



The 13th International Semantic Web Conference

19-23 October 2014

Proceedings of the ISWC 2014 Posters & Demonstrations Track

Editors:

Matthew Horridge, Marco Rospocher and Jacco van Ossenbruggen

Preface

The ISWC 2014 Poster and Demonstration track complements the Research Paper track of the conference and offers an opportunity for presenting late-breaking research results, ongoing research projects, and speculative or innovative work in progress. The informal setting of the track encourages presenters and participants to engage in discussions about the presented work. Such discussions can be a valuable input into the future work of the presenters, while offering participants an effective way to broaden their knowledge of the emerging research trends and to network with other researchers.

These proceedings contain the four-page abstracts of all accepted posters and demos presented at ISWC 2014. Posters range from technical contributions, reports on Semantic Web software systems, descriptions of completed work, and also work in progress. Demonstrations showcase innovative Semantic Web related implementations and technologies. This year we had 156 submissions, of which the program committee accept 71 posters and 49 demos. We would like to take this opportunity to thank all of the authors for their contributions to the ISWC 2014 programme!

We would also like to thank the members of the program committee and the additional reviewers for their time and efforts. A special thanks for respecting our deadlines, we know these fell in the middle of the summer holidays for many of you! All abstracts included here have been revised and improved based on your valuable feedback, and we feel the final result represents a wide variety of topics that will offer a vibrant and exciting session at the conference.

Finally, we would like to thank our local organisers Luciano Serafini and Chiara Ghidini for their invaluable help in sorting out the logistics of this track.

September 2014
Stanford, Trento, Amsterdam

Matthew Horridge
Marco Rospocher
Jacco van Ossenbruggen

Program Committee

Alessandro Adamou	Despoina Magka
Carlo Allocca	Sara Magliacane
Samantha Bail	James Malone
Pierpaolo Basile	Nicolas Matentzoglou
Eva Blomqvist	Georgios Meditskos
Victor de Boer	Alessandra Mileo
Stefano Bortoli	Kody Moodley
Loris Bozzato	Andrea Moro
Volha Bryl	Yuan Ni
Marut Buranarach	Andrea Giovanni Nuzzolese
Jim Burton	Alessandro Oltramari
Elena Cabrio	Jacco van Ossenbruggen
Annalina Caputo	Alessio Palmero Aproso
Vinay Chaudhri	Matteo Palmonari
Gong Cheng	Jeff Z. Pan
Sam Coppens	Guilin Qi
Oscar Corcho	José Luis Redondo-García
Francesco Corcoglioniti	Marco Rospocher
Claudia D'Amato	Tuukka Ruotsalo
Chiara Del Vescovo	Manuel Salvadores
Aidan Delaney	Bernhard Schandl
Daniele Dell'Aglio	Christoph Schütz
Chiara Di Francescomarino	Patrice Seyed
Mauro Dragoni	Giorgos Stoilos
Marieke van Erp	Nenad Stojanovic
Antske Fokkens	Mari Carmen Suárez-Figueroa
Marco Gabriele Enrico Fossati	Hideaki Takeda
Anna Lisa Gentile	Myriam Traub
Aurora Gerber	Tania Tudorache
Jose Manuel Gomez-Perez	Anni-Yasmin Turhan
Tudor Groza	Victoria Uren
Gerd Gröner	Davy Van Deursen
Peter Haase	Willem Robert van Hage
Armin Haller	Maria Esther Vidal
Karl Hammar	Serena Villata
Michiel Hildebrand	Boris Villazón-Terrazas
Pascal Hitzler	Stefanos Vrochidis
Aidan Hogan	Martine de Vos
Matthew Horridge	Simon Walk
Hanmin Jung	Hai H. Wang
Simon Jupp	Haofen Wang
Haklae Kim	Kewen Wang
Pavel Klinov	Shenghui Wang
Patrick Lambrix	Jun Zhao
Paea Le Pendu	Yi Zhou
Florian Lemmerich	Amal Zouaq
Yuan-Fang Li	
Joanne Luciano	

Additional Reviewers

Muhammad Intizar Ali
David Carral
Marco Cremaschi
Brian Davis
Jangwon Gim
Hegde Vinod
Nazmul Hussain
Myungwon Hwang
Amit Joshi
Kim Taehong
Kim Young-Min
Fadi Maali
Theofilos Mailis
Nicolas Matentzoglou
Jim Mccusker
David Molik
Raghava Mutharaju
Alina Patelli
Thomas Ploeger
Riccardo Porrini
Anon Reviewera
Victor Saquicela
Ana Sasa Bastinos
Veronika Thost
Jung-Ho Um
Zhangquan Zhou

Contents

1	Life Stories as Event-based Linked Data: Case Semantic National Biography <i>Eero Hyvönen, Miika Alonen, Esko Ikkala and Eetu Mäkelä</i>	1
2	News Visualization based on Semantic Knowledge <i>Sebastian Arnold, Damian Burke, Tobias Dörsch, Bernd Löber and Andreas Lommatzsch</i>	5
3	Sherlock: a Semi-Automatic Quiz Generation System using Linked Data <i>Dong Liu and Chenghua Lin</i>	9
4	Low-Cost Queryable Linked Data through Triple Pattern Fragments <i>Ruben Verborgh, Olaf Hartig, Ben De Meester, Gerald Haesendonck, Laurens De Vocht, Miel Vander Sande, Richard Cyganiak, Pieter Colpaert, Erik Mannens and Rik Van de Walle</i>	13
5	call: A Nucleus for a Web of Open Functions <i>Maurizio Atzori</i>	17
6	Cross-lingual detection of world events from news articles <i>Gregor Leban, Blaž Fortuna, Janez Brank and Marko Grobelnik</i>	21
7	Multilingual Word Sense Disambiguation and Entity Linking for Everybody <i>Andrea Moro, Francesco Cecconi and Roberto Navigli</i>	25
8	Help me describe my data: A demonstration of the Open PHACTS VoID Editor <i>Carole Goble, Alasdair J. G. Gray and Eleftherios Tatakis</i>	29
9	OUSocial2 - A Platform for Gathering Students' Feedback from Social Media <i>Keerthi Thomas, Miriam Fernandez, Stuart Brown and Harith Alani</i>	33
10	Using an Ontology Learning System for Trend Analysis and Detection <i>Gerhard Wohlgenannt, Stefan Belk, Matyas Karacsonyi and Matthias Schett</i>	37
11	A Prototype Web Service for Benchmarking Power Consumption of Mobile Semantic Applications <i>Evan Patton and Deborah McGuinness</i>	41
12	SPARKLIS: a SPARQL Endpoint Explorer for Expressive Question Answering <i>Sébastien Ferré</i>	45
13	Reconciling Information in DBpedia through a Question Answering System <i>Elena Cabrio, Alessio Palmero Aprosio and Serena Villata</i>	49
14	Open Mashup Platform - A Smart Data Exploration Environment <i>Tuan-Dat Trinh, Ba-Lam Do, Peter Wetz, Amin Anjomshoaa, Elmar Kiesling and Amin Tjoa</i>	53

15	CIMBA - Client-Integrated MicroBlogging Architecture <i>Andrei Sambra, Sandro Hawke, Timothy Berners-Lee, Lalana Kagal and Ashraf Aboulnaga</i>	57
16	The Organiser - A Semantic Desktop Agent based on NEPOMUK <i>Sebastian Faubel and Moritz Eberl</i>	61
17	HDTourist: Exploring Urban Data on Android <i>Elena Hervalejo, Miguel A. Martinez-Prieto, Javier D. Fernández and Oscar Corcho</i>	65
18	Integrating NLP and SW with the KnowledgeStore <i>Marco Rospocher, Francesco Corcoglioniti, Roldano Cattoni, Bernardo Magnini and Luciano Serafini</i>	69
19	Graphical Representation of OWL 2 Ontologies through Graphol <i>Marco Console, Domenico Lembo, Valerio Santarelli and Domenico Fabio Savo</i>	73
20	LIVE: a Tool for Checking Licenses Compatibility between Vocabularies and Data <i>Guido Governatori, Ho-Pun Lam, Antonino Rotolo, Serena Villata, Ghislain Auguste Atemezing and Fabien Gandon</i>	77
21	The Map Generator Tool <i>Valeria Fionda, Giuseppe Pirrò and Claudio Gutierrez</i>	81
22	Named Entity Recognition using FOX <i>René Speck and Axel-Cyrille Ngonga Ngomo</i>	85
23	A Linked Data Platform adapter for the Bugzilla issue tracker <i>Nandana Mihindukulasooriya, Miguel Esteban-Gutierrez and Raúl García-Castro</i>	89
24	LED: curated and crowdsourced Linked Data on Music Listening Experiences <i>Alessandro Adamou, Mathieu D'Aquin, Helen Barlow and Simon Brown</i>	93
25	WhatTheySaid: Enriching UK Parliament Debates with Semantic Web <i>Yunjia Li, Chaohai Ding and Mike Wald</i>	97
26	Multilingual Disambiguation of Named Entities Using Linked Data <i>Ricardo Usbeck, Axel-Cyrille Ngonga Ngomo, Wencan Luo and Lars Wesemann</i>	101
27	The Wikipedia Bitaxonomy Explorer <i>Tiziano Flati and Roberto Navigli</i>	105
28	Enhancing Web intelligence with the content of online video fragments <i>Lyndon Nixon, Matthias Bauer and Arno Scharl</i>	109
29	EMBench: Generating Entity-Related Benchmark Data <i>Ekaterini Ioannou and Yannis Velegarakis</i>	113

30	Demonstration of multi-perspective exploratory search with the Discovery Hub web application <i>Nicolas Marie and Fabien Gandon</i>	117
31	Modeling and Monitoring Processes exploiting Semantic Reasoning <i>Mauro Dragoni, Piergiorgio Bertoli, Chiara Di Francescomarino, Chiara Ghidini, Michele Nori, Marco Pistore, Roberto Tiella and Francesco Corcoglioniti</i>	121
32	WikipEvent: Temporal Event Data for the Semantic Web <i>Ujwal Gadiraju, Kaweh Djafari Naini, Andrea Ceroni, Mihai Georgescu, Dang Duc Pham and Marco Fisichella</i>	125
33	Towards a DBpedia of Tourism: the case of Tourpedia <i>Stefano Cresci, Andrea D’Errico, Davide Gazzè, Angelica Lo Duca, Andrea Marchetti and Maurizio Tesconi</i>	129
34	Using Semantics for Interactive Visual Analysis of Linked Open Data <i>Gerwald Tschinkel, Eduardo Veas, Belgin Muthu and Vedran Sabol</i>	133
35	Exploiting Linked Data Cubes with OpenCube Toolkit <i>Evangelos Kalampokis, Andriy Nikolov, Peter Haase, Richard Cyganak, Arkadiusz Stasiewicz, Areti Karamanou, Maria Zotou, Dimitris Zeginis, Efthimios Tambouris and Konstantinos Tarabanis</i>	137
36	Detecting Hot Spots in Web Videos <i>José Luis Redondo-García, Mariella Sabatino, Pasquale Lisena and Raphaël Troncy</i>	141
37	EUROSENTIMENT: Linked Data Sentiment Analysis <i>J. Fernando Sánchez-Rada, Gabriela Vulcu, Carlos A. Iglesias and Paul Buitelaar</i>	145
38	Property-based typing with LITEQ <i>Stefan Scheglmann, Martin Leinberger, Ralf Lämmel, Steffen Staab and Matthias Thimm</i>	149
39	From Tale to Speech: Ontology-based Emotion and Dialogue Annotation of Fairy Tales with a TTS Output <i>Christian Eisenreich, Jana Ott, Tonio Süßdorf, Christian Willms and Thierry Declerck</i>	153
40	BIOTEX: A system for Biomedical Terminology Extraction, Ranking, and Validation <i>Juan Antonio Lossio Ventura, Clement Jonquet, Mathieu Roche and Maguelonne Teisseire</i>	157
41	Visualizing and Animating Large-scale Spatiotemporal Data with ELBAR Explorer <i>Suvodeep Mazumdar and Tomi Kauppinen</i>	161
42	A Demonstration of Linked Data Source Discovery and Integration <i>Jason Slepicka, Chengye Yin, Pedro Szekely and Craig Knoblock</i>	165

43	Developing Mobile Linked Data Applications <i>Oshani Seneviratne, Evan Patton, Daniela Miao, Fuming Shih, Weihua Li, Lalana Kagal and Carlos Castillo</i>	169
44	A Visual Summary for Linked Open Data sources <i>Fabio Benedetti, Laura Po and Sonia Bergamaschi</i>	173
45	EasyESA: A Low-effort Infrastructure for Explicit Semantic Analysis <i>Danilo Carvalho, Cagatay Calli, Andre Freitas and Edward Curry</i>	177
46	LODHub - A Platform for Sharing and Analyzing large-scale Linked Open Data <i>Stefan Hagedorn and Kai-Uwe Sattler</i>	181
47	LOD4AR: Exploring Linked Open Data with a Mobile Augmented Reality Web Application <i>Silviu Vert, Bogdan Dragulescu and Radu Vasiu</i>	185
48	PLANET: Query Plan Visualizer for Shipping Policies against Single SPARQL Endpoints <i>Maribel Acosta, Maria Esther Vidal, Fabian Flöck, Simon Castillo and Andreas Harth</i>	189
49	High Performance Linked Data Processing for Virtual Reality Environments <i>Felix Leif Keppmann, Tobias Käfer, Steffen Stadtmüller, René Schubotz and Andreas Harth</i>	193
50	Analyzing Relative Incompleteness of Movie Descriptions in the Web of Data: A Case Study <i>Wancheng Yuan, Elena Demidova, Stefan Dietze and Xuan Zhou</i>	197
51	A Semantic Metadata Generator for Web Pages Based on Keyphrase Extraction <i>Dario De Nart, Carlo Tasso and Dante Degl’Innocenti</i>	201
52	A Multilingual SPARQL-Based Retrieval Interface for Cultural Heritage Objects <i>Dana Dannells, Ramona Enache and Mariana Damova</i>	205
53	Extending Tagging Ontologies with Domain Specific Knowledge <i>Frederic Font, Sergio Oramas, György Fazekas and Xavier Serra</i>	209
54	Disambiguating Web Tables using Partial Data <i>Ziqi Zhang</i>	213
55	On Linking Heterogeneous Dataset Collections <i>Mayank Kejriwal and Daniel Miranker</i>	217
56	Scientific data as RDF with Arrays: Tight integration of SciSPARQL queries into MATLAB <i>Andrej Andrejev, Xueming He and Tore Risch</i>	221
57	Measuring similarity in ontologies: a new family of measures <i>Tahani Alsubait, Bijan Parsia and Uli Sattler</i>	225

58	Towards Combining Machine Learning with Attribute Exploration for Ontology Refinement <i>Jedrzej Potoniec, Sebastian Rudolph and Agnieszka Lawrynowicz</i>	229
59	ASSG: Adaptive structural summary for RDF graph data <i>Haiwei Zhang, Yuanyuan Duan, Xiaojie Yuan and Ying Zhang</i>	233
60	Evaluation of String Normalisation Modules for String-based Biomedical Vocabularies Alignment with AnAGram <i>Anique van Berne and Veronique Malaise</i>	237
61	Keyword-Based Semantic Search Engine Koios++ <i>Björn Forcher, Andreas Giloj and Erich Weichselgartner</i>	241
62	Supporting SPARQL Update Queries in RDF-XML Integration <i>Nikos Bikakis, Chrisa Tsinaraki, Ioannis Stavrakantonakis and Stavros Christodoulakis</i>	245
63	CURIOS: Web-based Presentation and Management of Linked Datasets <i>Hai Nguyen, Stuart Taylor, Gemma Webster, Nophadol Jekjantuk, Chris Mellish, Jeff Z. Pan and Tristan Ap Rheinallt</i>	249
64	The uComp Protege Plugin for Crowdsourcing Ontology Validation <i>Florian Hanika, Gerhard Wohlgenannt and Marta Sabou</i>	253
65	Frame-Semantic Web: a Case Study for Korean <i>Jungyeul Park, Sejin Nam, Youngsik Kim, Younggyun Hahm, Dosam Hwang and Key-Sun Choi</i>	257
66	SparkRDF: Elastic Discreted RDF Graph Processing Engine With Distributed Memory <i>Xi Chen, HuaJun Chen, Ningyu Zhang and songyang Zhang</i>	261
67	LEAPS: A Semantic Web and Linked data framework for the Algal Biomass Domain <i>Monika Solanki</i>	265
68	Bridging the Semantic Gap between RDF and SPARQL using Completeness Statements <i>Fariz Darari, Simon Razniewski and Werner Nutt</i>	269
69	COLINA: A Method for Ranking SPARQL Query Results through Content and Link Analysis <i>Azam Feyznia, Mohsen Kahani and Fattane Zarrinkalam</i>	273
70	Licentia: a Tool for Supporting Users in Data Licensing on the Web of Data <i>Cristian Cardellino, Serena Villata, Fabien Gandon, Guido Governatori, Ho-Pun Lam and Antonino Rotolo</i>	277
71	Automatic Stopword Generation using Contextual Semantics for Sentiment Analysis of Twitter <i>Hassan Saif, Miriam Fernandez and Harith Alani</i>	281
72	The Manchester OWL Repository: System Description <i>Nicolas Matentzoglou, Daniel Tang, Bijan Parsia and Uli Sattler</i>	285

73	A Fully Parallel Framework for Analyzing RDF Data <i>Long Cheng, Spyros Kotoulas, Tomas Ward and Georgios Theodoropoulos</i>	289
74	Objects as results from graph queries using an ORM and generated semantic-relational binding <i>Marc-Antoine Parent</i>	293
75	Hedera: Scalable Indexing and Exploring Entities in Wikipedia Revision History <i>Tuan Tran and Tu Ngoc Nguyen</i>	297
76	Evaluating Ontology Alignment Systems in Query Answering Tasks <i>Alessandro Solimando, Ernesto Jimenez-Ruiz and Christoph Pinkel</i>	301
77	Using Fuzzy Logic For Multi-Domain Sentiment Analysis <i>Mauro Dragoni, Andrea Tettamanzi and Célia Da Costa Pereira</i>	305
78	AMSL — Creating a Linked Data Infrastructure for Managing Electronic Resources in Libraries <i>Natanael Arndt, Sebastian Nuck, Andreas Nareike, Norman Radtke, Leander Seige and Thomas Riechert</i>	309
79	Extending an ontology alignment system with BioPortal: a preliminary analysis <i>Xi Chen, Weiguo Xia, Ernesto Jimenez-Ruiz and Valerie Cross</i>	313
80	How much navigable is the Web of Linked Data? <i>Valeria Fionda and Enrico Malizia</i>	317
81	A Framework for Incremental Maintenance of RDF Views of Relational Data <i>Vânia Vidal, Marco Antonio Casanova, Jose Monteiro, Narciso Arruda, Diego Sá and Valeria Pequeno</i>	321
82	Document Relation System Based on Ontologies for the Security Domain <i>Janine Hellriegel, Hans Ziegler and Ulrich Meissen</i>	325
83	Representing Swedish Lexical Resources in RDF with lemon <i>Lars Borin, Dana Dannells, Markus Forsberg and John P. Mccrae</i>	329
84	QASM: a Q&A Social Media System Based on Social Semantic <i>Zide Meng, Fabien Gandon and Catherine Faron-Zucker</i>	333
85	A Semantic-Based Platform for Efficient Online Communication <i>Zaenal Akbar, José María García, Ioan Toma and Dieter Fensel</i>	337
86	SHAX: The Semantic Historical Archive eXplorer <i>Michael Feldman, Shen Gao, Marc Novel, Katerina Papaioannou and Abraham Bernstein</i>	341
87	SemanTex: Semantic Text Exploration Using Document Links Implied by Conceptual Networks Extracted from the Texts <i>Suad Aldarra, Emir Muñoz, Pierre-Yves Vandenbussche and Vit Novacek</i>	345

88	Towards a Top-K SPARQL Query Benchmark <i>Shima Zahmatkesh, Emanuele Della Valle, Daniele Dell'aglio and Alessandro Bozzon</i>	349
89	Exploring type-specific topic profiles of datasets: a demo for educational linked data <i>Davide Taibi, Stefan Dietze, Besnik Fetahu and Giovanni Fulantelli</i>	353
90	TEX-OWL: a Latex-Style Syntax for authoring OWL 2 ontologies <i>Matteo Matassoni, Marco Rospocher, Mauro Dragoni and Paolo Bouquet</i>	357
91	Supporting Integrated Tourism Services with Semantic Technologies and Machine Learning <i>Francesca Alessandra Lisi and Floriana Esposito</i>	361
92	Towards a Semantically Enriched Online Newspaper <i>Ricardo Kawase, Eelco Herder and Patrick Siehndel</i>	365
93	Identifying Topic-Related Hyperlinks on Twitter <i>Patrick Siehndel, Ricardo Kawase, Eelco Herder and Thomas Risse</i>	369
94	Capturing and Linking Human Sensor Observations with YouSense <i>Tomi Kauppinen, Evgenia Litvinova and Jan Kallenbach</i>	373
95	An update strategy for the WaterFowl RDF data store <i>Olivier Curé and Guillaume Blin</i>	377
96	Linking Historical Data on the Web <i>Valeria Fionda and Giovanni Grasso</i>	381
97	User driven Information Extraction with LODIE <i>Anna Lisa Gentile and Suvodeep Mazdumar</i>	385
98	QALM: a Benchmark for Question Answering over Linked Merchant Websites Data <i>Amine Hallili, Elena Cabrio and Catherine Faron Zucker</i>	389
99	GeoTriples: a Tool for Publishing Geospatial Data as RDF Graphs Using R2RML Mappings <i>Kostis Kyzirakos, Ioannis Vlachopoulos, Dimitrianos Savva, Stefan Mane-gold and Manolis Koubarakis</i>	393
100	New Directions in Linked Data Fusion <i>Jan Michelfeit and Jindich Mynarz</i>	397
101	Bio2RDF Release 3: A larger, more connected network of Linked Data for the Life Sciences <i>Michel Dumontier, Alison Callahan, Jose Cruz-Toledo, Peter Ansell, Vincent Emonet, François Belleau and Arnaud Droit</i>	401
102	Infoboxer: Using Statistical and Semantic Knowledge to Help Create Wikipedia Infoboxes <i>Roberto Yus, Varish Mulwad, Tim Finin and Eduardo Mena</i>	405

103	The Topics they are a-Changing — Characterising Topics with Time-Stamped Semantic Graphs <i>Amparo E. Cano, Yulan He and Harith Alani</i>	409
104	Linked Data and facets to explore text corpora in the Humanities: a case study <i>Christian Morbidoni</i>	413
105	Dexter 2.0 - an Open Source Tool for Semantically Enriching Data <i>Diego Ceccarelli, Claudio Lucchese, Salvatore Orlando, Raffaele Perego and Salvatore Trani</i>	417
106	A Hybrid Approach to Learn Description Logic Ontology from Texts <i>Yue Ma and Alifah Syamsiyah</i>	421
107	Identifying First Responder Communities Using Social Network Analysis <i>John Erickson, Katherine Chastain, Zachary Fry, Jim Mccusker, Rui Yan, Evan Patton and Deborah McGuinness</i>	425
108	Exploiting Semantic Annotations for Entity-based Information Retrieval <i>Lei Zhang, Michael Färber, Thanh Tran and Achim Rettinger</i>	429
109	Crawl Me Maybe: Iterative Linked Dataset Preservation <i>Besnik Fetahu, Ujwal Gadiraju and Stefan Dietze</i>	433
110	A Semantics-Oriented Storage Model for Big Heterogeneous RDF Data <i>Hyeongsik Kim, Padmashree Ravindra and Kemafor Anyanwu</i>	437
111	Approximating Inference-enabled Federated SPARQL Queries on Multiple Endpoints <i>Yuji Yamagata and Naoki Fukuta</i>	441
112	VKGBuilder – A Tool of Building and Exploring Vertical Knowledge Graphs <i>Tong Ruan, Haofen Wang and Fanghuai Hu</i>	445
113	Using the semantic web for author disambiguation - are we there yet? <i>Cornelia Hedeler, Bijan Parsia and Brigitte Mathiak</i>	449
114	SHEPHERD: A Shipping-Based Query Processor to Enhance SPARQL Endpoint Performance <i>Maribel Acosta, Maria Esther Vidal, Fabian Flöck, Simon Castillo, Carlos Buil Aranda and Andreas Harth</i>	453
115	AgreementMakerLight 2.0: Towards Efficient Large-Scale Ontology Matching <i>Daniel Faria, Catia Pesquita, Emanuel Santos, Isabel F. Cruz and Francisco Couto</i>	457
116	Extracting Architectural Patterns from Web data <i>Ujwal Gadiraju, Ricardo Kawase and Stefan Dietze</i>	461
117	Xodx — A node for the Distributed Semantic Social Network <i>Natanael Arndt and Sebastian Tramp</i>	465

118	An Ontology Explorer for Biomimetics Database <i>Kouji Kozaki and Riichiro Mizoguchi</i>	469
119	Semi-Automated Semantic Annotation of the Biomedical Literature <i>Fabio Rinaldi</i>	473
120	Live SPARQL Auto-Completion <i>Stephane Campinas</i>	477

Life Stories as Event-based Linked Data: Case Semantic National Biography

Eero Hyvönen, Miika Alonen, Esko Ikkala, and Eetu Mäkelä

Semantic Computing Research Group (SeCo), Aalto University
<http://www.seco.tkk.fi/>, firstname.lastname@aalto.fi

Abstract. This paper argues, by presenting a case study and a demonstration on the web, that biographies make a promising application case of Linked Data: the reading experience can be enhanced by enriching the biographies with additional life time events, by providing the user with a spatio-temporal context for reading, and by linking the text to additional contents in related datasets.

1 Introduction

This paper addresses the research question: *How can the reading experience of biographies be enhanced using web technologies?* Our research hypotheses is to apply the Linked Data (LD) approach to this, with the idea of providing the reader with a richer reading context than the biography document alone. The focus of research is on: 1) *Data linking*. Biographies can be linked with additional contextual data, such as links to the literal works of the person. 2) *Data enriching*. Data from different sources can be used for enriching the life story with additional events and data, e.g., with metadata about a historical event that the person participated in. 3) *Visualization*. LD can be visualized in useful ways. The life story can, e.g., be shown on maps and timelines. We tested the hypotheses in a case study¹ where the Finnish National Biography² (NB), a collection of 6,381 short biographies, is published as LD in a SPARQL endpoint with a demonstrational application based on its standard API.

2 Representing Biographies as Linked Data

To enrich and link biographical data with related datasets the data must be made semantically interoperable, either by data alignments (using, e.g., Dublin Core and the dumb down principle) or by data transformations into a harmonized form [3]. In our case study we selected the data harmonization approach and the event-centric CIDOC CRM³ ISO standard as the ontological basis, since biographies are based on life events. NB biographies are modeled as collections of CIDOC CRM events, where each event is characterized by the 1) actors involved, 2) place, 3) time, and 4) the event type.

¹ Our work was funded by Tekes, Finnish Cultural Foundation, and the Linked Data Finland consortium of 20 organizations.

² <http://www.kansallisbiografia.fi/english/?p=2>

³ <http://www.cidoc-crm.org/>

A simple custom event extractor was created for transforming biographies into this model represented in RDF. The extractor first lemmatizes a biography and then analyzes its major parts: a textual story followed by systematically titled sections listing major achievements of the person, such as “works”, “awards”, and “memberships” as snippets. A snippet represents an event and typically contains mentions of years and places. For example, the biography of architect Alvar Aalto tells “WORKS: ...; Church of Muurame 1926-1929;...” indicating an artistic creation event. The named entity recognition tool of the Machinese⁴ NLP library is used for finding place names in the snippets, and Geonames is used for geocoding. Timespans of snippet events are found easily as numeric years or their intervals, and an actor of the events is the subject person of the biography. The result of processing a biography is a list of spatio-temporal CIDOC CRM events with short titles (snippet texts) related to the corresponding person. At the moment, the extractor uses only the snippets for event creation—more generic event extraction from the free biography narrative remains a topic of further research.

For a domain ontology, we reused the Finnish History Ontology HISTO by transforming it into CIDOC CRM. The new HISTO version contains 1,173 major historical events (E5_Event in CIDOC CRM) covering over 1000 years of Finnish history, and includes 80,085 activities (E7_Activity) of different kinds, such as armistice, election etc. Linked to these are 7,302 persons (E21_Person) and a few hundred organizations and groups, 3,290 places (E53_Place), and 11,141 time spans (E52_Time-span). The data originates from the Agricola timeline⁵ created by Finnish historians.

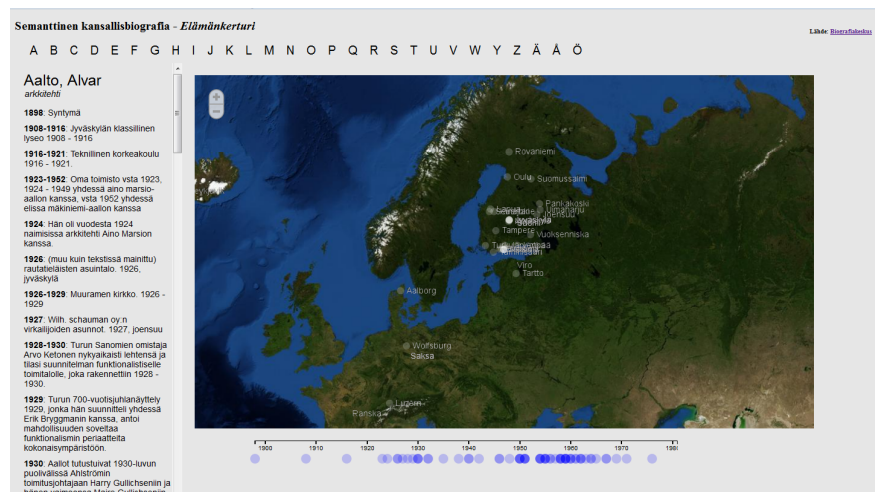


Fig. 1. Spatio-temporal visualization of Alvar Aalto's life with external links.

⁴ <http://www.connexor.com/nlplib/>

⁵ <http://agricola.utu.fi/>

The extracted events were then enriched with events from external datasets as follows: 1) Persons in SNB and HISTO were mapped onto each other based on their names. This worked well without further semantic disambiguation since few different persons had similar names. NB and HISTO shared 921 persons p , and the biography of each p could therefore be enriched with all HISTO events that p was involved in. 2) There were 361 artistic creation events (e.g., publishing a book) of NB persons that could be extracted from Europeana Linked Open Data⁶ using the person as the creator. Related biographies could therefore be enriched with events pointing to Europeana contents. 3) The NB persons were involved in 263 instances of publications of the Project Gutenberg data⁷. Corresponding events could therefore be added into the biographies, and links to the original digitized publications be provided. 4) The NB persons were also linked to Wikipedia for additional information; again simple string matching produced good results. These examples demonstrate how linked events can be extracted from other datasets and be used for enriching other biographical events. In the experiment, 116,278 spatio-temporal events were finally extracted for the NB biography records.

3 Biographies Enriched in a Spatio-temporal Context

Based on the enriched and linked biography data, a demonstrator was created proving the end user with a spatio-temporal context for reading NB biographical data as well as links to additional content from related sources. Fig. 1 depicts the user interface online⁸ with architect Alvar Aalto’s biography selected; the other 6,400 celebrities can be selected from the alphabetical list above. On the left column, temporal events extracted from the biography and related datasets are presented (in Finnish), such as “1898 Birth”, and “1908-1916 Jyväskylä Classical Lyceum”. The event “1930–1939: Alvar Aalto created his famous functionalist works (*Histo*)” shows an external link to HISTO for additional information. The events are also seen as bubbles on a timeline at the bottom. The map in the middle shows the end-user the places related to the biography events. By hovering the mouse over an event or its bubble the related event is high-lighted and the map zoomed and centered around the place related to the event. In this way the user can quickly get an overview about the spatio-temporal context of Alvar Aalto’s life, and get links to additional sources of information. The actual biography text can be read by clicking a link lower in the interface (not visible in the figure). The user interface also performs dynamic SPARQL querying for additional external links. In our demonstration, the BookSampo dataset and SPARQL endpoint [6] is used for enriching literature-related biographies with additional publication and literature award events.

The user interface for spatio-temporal lifeline visualization was implemented using AngularJS⁹ and D3¹⁰ on top of the Linked Data Finland (LDF) data service¹¹.

⁶ <http://pro.europeana.eu/linked-open-data>

⁷ <http://datahub.io/dataset/gutenberg>

⁸ <http://www.ldf.fi/dataset/history/map.html>

⁹ <http://angularjs.org>

¹⁰ <http://d3js.org>

¹¹ Cf. <http://www.ldf.fi/dataset/history/> for dataset documentation and SPARQL endpoint

4 Discussion, Related Work, and Future Research

Our case study suggests that biography publication is a promising application case for LD. The event-based modeling approach was deemed useful and handy, after learning basics of the fairly complex CIDOC CRM model. The snippet events could be extracted and aligned with related places, times, and actors fairly accurately using simple string-based techniques. However, the results of event extraction and entity linking have not been evaluated formally, and it is obvious that problems grow with larger datasets and when analysing free text—these issues are a topic of future research.

Biographical data has been studied by genealogists (e.g., (Event) GEDCOM¹²), CH organizations (e.g., the Getty ULAN¹³), and semantic web researchers (e.g., BIO ontology¹⁴). Semantic web event models include, e.g., Event Ontology [8], LOD ontology¹⁵, SEM [1], and Event-Model-F¹⁶ [9]. A history ontology with map visualizations is presented in [7], and an ontology of historical events in [4]. Visualization using historical timelines is discussed, e.g., in [5], and event extraction reviewed in [2].

References

1. van Hage, W.R., Malaisé, V., Segers, R., Hollink, L., Schreiber, G.: Design and use of the simple event model (SEM). *Web Semantics: Science, Services and Agents on the World Wide Web* 9(2), 128–136 (2011)
2. Hogenboom, F., Frasinca, F., Kaymak, U., de Jong, F.: An overview of event extraction from text. In: *DeRiVE 2011, Detection, Representation, and Exploitation of Events in the Semantic Web* (2011), <http://ceur-ws.org/Vol-779/>
3. Hyvönen, E.: *Publishing and using cultural heritage linked data on the semantic web*. Morgan & Claypool, Palo Alto, CA (2012)
4. Hyvönen, E., Alm, O., Kuittinen, H.: Using an ontology of historical events in semantic portals for cultural heritage. In: *Proceedings of the Cultural Heritage on the Semantic Web Workshop at the 6th International Semantic Web Conference (ISWC 2007)* (2007), <http://www.cs.vu.nl/~laroyo/CH-SW.html>
5. Jensen, M.: *Vizualising complex semantic timelines*. NewsBlip Research Papers, Report NBTR2003-001 (2003), <http://www.newsblip.com/tr/>
6. Mäkelä, E., Ruotsalo, T., Hyvönen: How to deal with massively heterogeneous cultural heritage data—lessons learned in CultureSampo. *Semantic Web – Interoperability, Usability, Applicability* 3(1) (2012)
7. Nagypal, G., Deswarte, R., Oosthoek, J.: Applying the semantic web: The VICODI experience in creating visual contextualization for history. *Lit Linguist Computing* 20(3), 327–349 (2005), <http://dx.doi.org/10.1093/lit/fqi037>
8. Raimond, Y., Abdallah, S.: *The event ontology* (2007), <http://motools.sourceforge.net/event/event.html>
9. Scherp, A., Saathoff, C., Franz, T.: *Event-Model-F* (2010), <http://www.uni-koblenz-landau.de/koblenz/fb4/AGStaab/Research/ontologies/events>

¹² <http://en.wikipedia.org/wiki/GEDCOM>

¹³ <http://www.getty.edu/research/tools/vocabularies/ulan/>

¹⁴ <http://vocab.org/bio/0.1/.html>

¹⁵ <http://linkedevents.org/ontology/>

¹⁶ <http://www.uni-koblenz-landau.de/koblenz/fb4/AGStaab/Research/ontologies/events>

News Visualization Based on Semantic Knowledge

Sebastian Arnold, Damian Burke, Tobias Dörsch,
Bernd Loeber, and Andreas Lommatzsch

Technische Universität Berlin
Ernst-Reuter-Platz 7, D-10587 Berlin, Germany
{sarnold, damian.burke, tobias.m.doersch, bernd.loeber,
andreas.lommatzsch}@mailbox.tu-berlin.de

Abstract. Due to the overwhelming amount of news articles from a growing number of sources, it has become nearly impossible for humans to select and read all articles that are relevant to get deep insights and form conclusions. This leads to a need for an easy way to aggregate and analyze news articles efficiently and visualize the garnered knowledge as a base for further cognitive processing. The presented application provides a tool to satisfy said need. In our approach we use semantic techniques to extract named entities, relations and locations from news sources in different languages. This knowledge is used as the base for data aggregation and visualization operators. The data operators include filtering of entities, types and date range, detection of correlated news topics for a set of selected entities and geospatial analysis based on locations. Our visualization provides a time-based graphical representation of news occurrences according to the given filters as well as an interactive map which displays news within a perimeter for the different locations mentioned in the news articles. In every step of the user process, we offer a tag cloud that highlights popular results and provide links to the original sources including highlighted snippets. Using the graphical interface, the user is able to analyze and explore vast amounts of fresh news articles, find possible relations and perform trend analysis in an intuitive way.

1 Introduction

Comprehensive news analysis is a common task for a broad range of recipients. To overlook the overwhelming amount of articles that are published in the Web every hour, new technologies are needed that help to classify, search and explore topics in real time. Current approaches focus on automated classification of documents into expert-defined categories, such as politics, business or sports. The results need to be tagged manually with meta-information about locations, people and current news topics. The simple model of categories and tags, however, is not detailed enough to suit temporal or regional relationships and it cannot bridge the semantic gap that the small subset of tagged information opens. The challenge for machine-driven news analysis consists of two parts. First, an extractor needs to be able to identify the key concepts and entities mentioned in the documents and to find the most important relationships between them. Second, an intuitive way for browsing the results with support for explorative discovery of relevant topics and emerging trends needs to be developed.

We present a semantic approach that abstracts from multi-lingual representation of facts and enriches extracted information with background knowledge. Our implementation utilizes natural language processing tools for the extraction of named entities, relations and semantic context. Open APIs are used to augment further knowledge (e.g. geo-coordinates) to the results. Our application visualizes the gained knowledge and provides time-based, location-based and relationship-based exploration operators. The relationship between original news documents and aggregated search results is maintained throughout the whole user process.

In Section 2, we give an overview on existing projects of similar focus. Our knowledge-based approach and the implementation is introduced in Section 3. The user interaction and visualization operators are discussed in Section 4. We conclude in Section 5.

2 Related Work

We start with an overview on existing projects in the field of semantic news visualization. The following projects are related to our approach on a conceptual or visual level.

MAPLANDIA¹ visualizes news for a specific date beginning in 2005 on a map. The system uses the BBC news feed as its only source to deliver markers and outlines for the countries that were mentioned in the news on a specified date. Additionally, it offers a list of the news for the day. However, by using only one marker the application is unable to visualize news on a more detailed and fine-grained level. MAPLANDIA also does not offer any possibility to limit the displayed visualizations to a certain region of interest. The application offers news in only one language and source for a specific day.

The idea behind the SPIGA-SYSTEM [3] is to provide businesses with a multilingual press review of news from national and international sources. Using the Apache UIMA framework, the system crawls a few thousand sources regardless of the language used. After a fitting business profile has been created, the system clusters information and visualizes current trends.

3 Implementation of Knowledge Extraction

In this section, we explain our semantic approach to news aggregation. In contrast to classical word- or tag-based indexing, we focus on semantic features that we extract from daily news documents. To handle the linguistic complexity of this problem, we utilize information extraction techniques for natural language [2]. The knowledge extraction pipeline is shown in Fig. 1. It consists of a periodic RSS feed crawler as source for news documents² and the language components for sentence splitting, part-of-speech (POS) tagging, named entity recognition (NER) and coreference resolution. We utilize a Stanford CoreNLP named entity recognition pipeline [1] for the languages English and German. The pipeline periodically builds histograms over the frequency of named entity occurrences in all documents. Using a 3-class entity typification (*Person*, *Organization*, *Location*) we apply special treatment to each of the entity types.

¹ <http://maplandia.com/news>

² In our demonstrator, it is configured to use feeds from <http://www.theguardian.com>

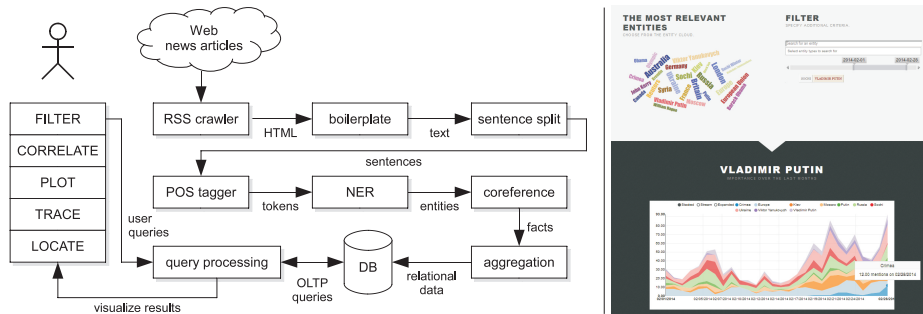


Fig. 1: The figure shows the architecture and a screenshot of our system. The system architecture is divided in user operators FILTER, CORRELATE, PLOT, TRACE and LOCATE (shown left) and the knowledge extraction pipeline (shown right). The screenshot of the Web application visualizes news entities related to *Vladimir Putin* utilizing the operators CORRELATE, FILTER and PLOT.

Person and *Organization* names are normalized to obtain representative identifiers from different spellings. *Location* names are resolved to geo-location coordinates using Google Maps API.³ The results are processed and aggregated into relations of the form $MentionedIn(ENTITY(type), DOCUMENT(date), frequency)$. To get a comprehensive view on the relevant information, we create histograms over these relations and store the distributions in a relational database. The data is processed using relational statements to fit the type of user query that is requested from the frontend. To increase processing performance of the information in Web-scale, we partly precompute these statements. The results are then visualized to allow easy recognition of trends and relationships.

4 Demonstrator and Benefits to the Audience

In this section, we present our user interface which has a specific focus on a simple workflow.⁴ Our aim is to relieve the user from having much work with sorting and filtering tables and instead allow exploratory search [4] inside the data set. We present the results in a clean web interface that establishes an interaction flow from top to bottom. Our system offers several operators to explore the temporal, geographical and relational distribution in the news corpus.

A user session starts with the FILTER operator that is visualized as a tag cloud showing the most mentioned entities in a given time range in a size proportional to their frequency (e.g. location *Russia*). The selection can be influenced by further filter settings, such as type and date restrictions. Clicking on an entity within the tag cloud will trigger the CORRELATE operator, which offers related entities to the selected one in the tag cloud (e.g. location *Ukraine*, person *Vladimir Putin*). This is done by intersecting the most relevant documents for a given entity and time range and picking the top mentioned named entities in these articles. Selecting different items will further narrow down the

³ <https://developers.google.com/maps/>

⁴ An online demo is available at <http://irml-lehre.aot.tu-berlin.de>

results. Both the selected and the correlated entities are then displayed in a time-based PLOT with the time range on the x-axis and the frequency of occurrences on the y-axis. To instantly reveal the relationships and importance of co-occurrent entities, one can modify the display style (e.g. stacked, expanded or streamed). To get more detailed information about specific data points, the user can hover the cursor above them to trigger a TRACE operation. Then, more details about the selected tuple (ENTITY, date) are revealed: counts and snippets of the news articles that mention the selected entity and links to trace back the original documents.

The LOCATE operator focuses on geographic locating of selected entities and their relations. The operator works by computing a set of bounding coordinates which are used to query the database for possible locations. Using the same FILTER interface, a location of interest can be selected from the tag cloud or by entering its name in the text field. By utilizing a slider to set a search perimeter, the user is able to further focus on the regions around the selected location. After selection, a world map will display locations mentioned in the matching articles. By clicking the markers on the map, a balloon listing shows up to ten headlines and links to the respective articles. This allows the user to gain an overview of connections and associations of different countries and locations.

5 Conclusion and Future Work

The application allows the user to quickly visualize and analyze vast amounts of news articles. By the use of interactive elements such as graphs and a world map the user is able to check hypotheses and draw conclusions in an explorative and playful manner. This greatly reduces the cognitive load for the user as he or she is able to find the relevant facts fast and browse the underlying news articles to get further information from the original source. In our conducted user studies we observed that a streamlined interface with the options at the top and the results below was most appealing to users, and fewer options led to a more intuitive experience. The presented application is realized as a prototype and will be expanded by further development. A broader range of information can be achieved by including more news sources, implementing extended language support (e.g. multi-lingual knowledge extraction) and expanding the features of the FILTER operator (e.g. including sentiment selection). A deeper enrichment of knowledge can be achieved by linking the detected entities to additional knowledge sources (e.g. DBpedia or freebase) and using context information to extract more language features (e.g. sentiments, quotations, relations).

References

1. D. Cer, M.-C. de Marneffe, D. Jurafsky, and C. D. Manning. Parsing to stanford dependencies: Trade-offs between speed and accuracy. In *7th Intl. Conf. LREC 2010*, 2010.
2. R. Grishman. Information extraction: Techniques and challenges. In *Information Extraction A Multidisciplinary Approach to an Emerging Information Technology*. Springer, 1997.
3. L. Hennig, D. Ploch, D. Prawdzik, B. Armbruster, H. Düwiger, E. W. De Luca, and S. Albayrak. Spiga - a multilingual news aggregator. In *Proceedings of GSCL 2011*, 2011.
4. G. Marchionini. Exploratory search: from finding to understanding. *Communications of the ACM*, 49(4):41–46, 2006.

Sherlock: a Semi-Automatic Quiz Generation System using Linked Data

Dong Liu¹ and Chenghua Lin²

¹ BBC Future Media & Technology - Knowledge & Learning, Salford M50 2QH, UK,
Dong.Liu@bbc.co.uk

² Department of Computing Science, University of Aberdeen, AB24 3UE, UK
chenghua.lin@abdn.ac.uk

Abstract. This paper presents Sherlock, a semi-automatic quiz generation system for educational purposes. By exploiting semantic and machine learning technologies, Sherlock not only offers a generic framework for domain independent quiz generation, but also provides a mechanism for automatically controlling the difficulty level of the generated quizzes. We evaluate the effectiveness of the system based on three real-world datasets.

Keywords: Quiz Generation, Linked Data, RDF, Educational Games

1 Introduction

Interactive games are effective ways of helping knowledge being transferred between humans and machines. For instance, efforts have been made to unleash the potential of using Linked Data to generate educational quizzes. However, it is observed that the existing approaches [1, 2] share some common limitations that they are either based on domain specific templates or the creation of quiz templates heavily relies on ontologist and Linked Data experts. There is no mechanism provided to end-users to engage with customised quiz authoring.

Moreover, a system that can generate quizzes with different difficulty levels will better serve users' needs. However, such an important feature is rarely offered by the existing systems, where most of the practices simply select the distractors (i.e., the wrong candidate answers) at random from an answer pool (e.g., obtained by querying the Linked Data repositories). Some work has attempted to determine the difficulty of a quiz but still it is simply based on assessing the popularity of a RDF resource, without considering the fact that the difficulty level of a quiz is directly affected by semantic relatedness between the correct answer and the distractors [3].

In this paper, we present a novel semi-automatic quiz generation system (Sherlock) empowered by semantic and machine learning technologies. Sherlock is distinguished from existing systems in a few aspects: (1) it offers a generic framework for generating quizzes of multiple domains with minimum human effort; (2) a mechanism is introduced for controlling the difficulty level of the generated quizzes; and (3) an intuitive interface is provided for engaging users

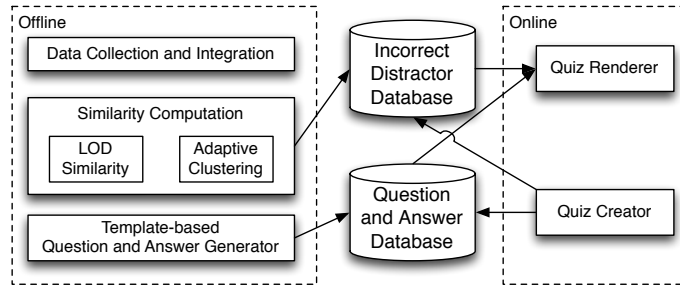


Fig. 1. Overall architecture of Sherlock.

in creating customised quizzes. The live Sherlock system can be accessed from <http://sherlock.pilots.bbconnectedstudio.co.uk/>¹.

2 System Architecture

Fig. 1 depicts an overview of the Sherlock framework, in which the components are logically divided into two groups: online and offline. These components can interact with each other via shared databases which contain the information of the questions, correct answers and distractors (i.e., incorrect answers).

Data Collection and Integration: We collected RDF data published by DBpedia and the BBC. These data play two main roles, i.e., serving as the knowledge base for quiz generation and used for calculating the similarity scores between objects/entities (i.e., answers and distractors).

Similarity Computation: The similarity computation module is the core for controlling the difficulty level of quiz generation. It first accesses the RDF store, and then calculates the similarity scores between each object/entity pairs. In the second step, the module performs K -means clustering to partition the distractors into different difficulty levels according to their Linked Data Semantic Distance (LSD) [4] scores with the correct answer of a quiz. In the preliminary experiment, we empirically set $K=3$ corresponding to three difficulty levels, i.e. “easy”, “medium” and “difficult”.

Template-based Question and Answer Generator: This module automates the process of generating questions and the correct answers. Fig. 2(a) demonstrates the instantiation of an example template: “Which of the following animals is $\{?animal_name\}$?”, where the variable is replaced with `rdfs:label` of the animal. The generated questions and answers will be saved in the database.

Quiz Renderer: The rendering module first retrieves the question and the correct answer from the database, and then selects suitable distractors from the entities returned by the similarity computation module. Fig. 2(a) shows the module’s intuitive gaming interface, as well as a distinctive feature for tuning up or down the quiz difficulty level dynamically, making Sherlock able to better serve the needs of different user groups (e.g., users of different age and with different

¹ For the best experiences, please use Safari or Opera to access the demo.

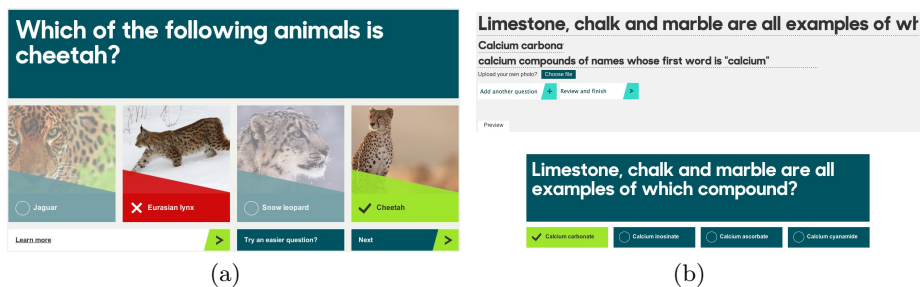


Fig. 2. (a) User interface for playing a quiz; (b) User interface for creating a quiz.

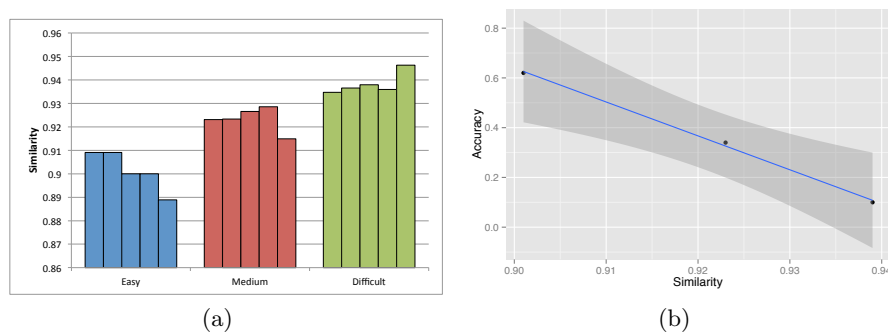


Fig. 3. (a) Averaged similarity of the testing quizzes (Wildlife domain); (b) Correlation measure of the Wildlife domain ($r = -0.97, p < 0.1$).

educational background). Furthermore, to enhance a user’s learning experience, the “*learn more*” link on the bottom left of the interface points to a Web page containing detailed information about the correct answer (e.g., Cheetah).

Quiz Creator: Fig. 2(b) depicts the quiz creator module, which complements the automatic quiz generation by allowing users to create customised quizzes with more diverse topics and to share with others. Quiz authoring involves three simple steps: 1) write a question; 2) set the correct answer (distractors are suggested by the Sherlock system automatically); and 3) preview and submit. For instance, one can take a picture of several ingredients and let people guess what dish one is going to cook. The quiz creator interface can be accessed from <http://sherlock.pilots.bbconnectedstudio.co.uk/#/quiz/create>.

3 Empirical Evaluation

This demo aims to show how Sherlock can effectively generate quizzes of different domains and how well a standard similarity measure can be used to suggest quiz difficulty level that matches human’s perception. The hypothesis is that if some objects/entities have higher degree of semantic relatedness, their differences would be subtle and hence more difficult to be disambiguated, and vice versa.

We investigated the correlation between the difficulty level captured by the similarity measure and that perceived by human. To test our hypothesis, a group of 10 human evaluators were presented with 45 testing quizzes generated by Sherlock based on the BBC Wildlife domain data, i.e., 15 quizzes per difficulty level. Next the averaged pairwise similarity between the correct answer and distractors of each testing quiz were computed, as shown in Fig. 3(a). Fig. 3(b) demonstrates that the quiz test accuracy of human evaluation indeed shows a negative correlation ($r = -0.97$, $p < 0.1$) with the average similarity of the quiz answer choices (i.e., each datapoint is the averaged value over 15 quizzes per difficulty level). This suggests that LDS is an appropriate similarity measure for indicating quiz difficulty level, which inlines with our hypothesis.

In another set of experiments, we evaluated Sherlock as a generic framework for quiz generation, in which the system was tested on structural RDF datasets from three different domains, namely, BBC Wildlife, BBC Food and BBC Your-Paintings², with 321, 991 and 2,315 quizzes automatically generated by the system for each domain respectively. Benefiting from the domain-independent similarity measure (LDS), Sherlock can be easily adapted to generate quizzes of new domains with minimum human efforts, i.e., no need to manually define rules or rewrite SPARQL queries.

4 Conclusion

In this paper, we presented a novel generic framework (Sherlock) for generating educational quizzes using linked data. Compared to existing systems, Sherlock offers a few distinctive features, i.e., it not only provides a generic framework for generating quizzes of multiple domains with minimum human effort, but also introduces a mechanism for controlling the difficulty level of the generated quizzes based on a semantic similarity measure.

Acknowledgements

The research described here is supported by the BBC Connected Studio programme and the award made by the RCUK Digital Economy theme to the dot.rural Digital Economy Hub; award reference EP/G066051/1. The authors would like to thank Ryan Hussey, Tom Cass, James Ruston, Herm Baskerville and Nava Tintarev for their valuable contribution.

References

- [1] Damljanovic, D., Miller, D., O’Sullivan, D.: Learning from quizzes using intelligent learning companions. In: WWW (Companion Volume). (2013) 435–438
- [2] Álvaro, G., Alvaro, J.: A linked data movie quiz: the answers are out there, and so are the questions [blog post]. <http://bit.ly/linkedmovies> (2010)
- [3] Waitelonis, J., Ludwig, N., Knuth, M., Sack, H.: WhoKnows? - evaluating linked data heuristics with a quiz that cleans up dbpedia. International Journal of Interactive Technology and Smart Education (ITSE) **8** (2011) 236–248
- [4] Passant, A.: Measuring semantic distance on linking data and using it for resources recommendations. In: AAAI Symposium: Linked Data Meets AI. (2010)

² <http://www.bbc.co.uk/nature/wildlife>, <http://www.bbc.co.uk/food> and <http://www.bbc.co.uk/arts/yourpaintings>

Low-Cost Queryable Linked Data through Triple Pattern Fragments

Ruben Verborgh¹, Olaf Hartig², Ben De Meester¹, Gerald Haesendonck¹,
Laurens De Vocht¹, Miel Vander Sande¹, Richard Cyganiak³, Pieter Colpaert¹,
Erik Mannens¹, and Rik Van de Walle¹

¹ Ghent University – iMinds, Belgium
{firstname.lastname}@ugent.be

² University of Waterloo, Canada
ohartig@uwaterloo.ca

³ Digital Enterprise Research Institute, NUI Galway, Ireland
richard@cyganiak.de

Abstract. For publishers of Linked Open Data, providing queryable access to their dataset is costly. Those that offer a public SPARQL endpoint often have to sacrifice high availability; others merely provide non-queryable means of access such as data dumps. We have developed a client-side query execution approach for which servers only need to provide a lightweight triple-pattern-based interface, enabling queryable access at low cost. This paper describes the implementation of a client that can evaluate SPARQL queries over such triple pattern fragments of a Linked Data dataset. Graph patterns of SPARQL queries can be solved efficiently by using metadata in server responses. The demonstration consists of SPARQL client for triple pattern fragments that can run as a standalone application, browser application, or library.

Keywords: Linked Data, Linked Data Fragments, querying, availability, scalability, SPARQL

1 Introduction

An ever increasing amount of Linked Data is published on the Web, a large part of which is freely and publicly available. The true value of these datasets becomes apparent when users can execute arbitrary queries over them, to retrieve precisely those facts they are interested in. The SPARQL query language [3] allows to specify highly precise selections, but it is very costly for servers to offer a public SPARQL endpoint over a large dataset [6]. As a result, current public SPARQL endpoints, often hosted by institutions that cannot afford an expensive server setup, suffer from low availability rates [1]. An alternative for these institutions is to provide their data in a non-queryable form, for instance, by allowing consumers to download a data dump which they can use to set up their own private SPARQL endpoint. However, this prohibits live querying of the data, and is in turn rather expensive on the client side.

In this demo, we will show a low-cost server interface that offers access to a dataset through all of its triple patterns, together with a client that performs efficient execution of complex queries through this interface. This enables publishers to provide Linked Data in a queryable way at low cost. The demo complements our paper at the ISWC2014 Research Track [6], which profoundly explains the principles behind the technology and experimentally verifies its scalability. The present paper details the implementation and introduces the supporting prototype implementation of our SPARQL client of triple pattern fragments.

2 Related Work

We contrast our approach with the three categories of current HTTP interfaces to RDF, each of which comes with its own trade-offs regarding performance, bandwidth, and client/server processor usage and availability.

Public SPARQL endpoints The current de-facto way for providing queryable access to triples on the Web is the SPARQL protocol, which is supported by many triple stores such as Virtuoso, AllegroGraph, Sesame, and Jena TDB. Even though current SPARQL interfaces offer high performance, individual queries can consume a significant amount of server processor time and memory. Because each client requests unique, highly specific queries, regular HTTP caching is ineffective, since this can only optimize repeated identical requests. These factors contribute to the low availability of public SPARQL endpoints, which has been documented extensively [1]. This makes providing reliable public SPARQL endpoints an exceptionally difficult challenge, incomparable to hosting regular public HTTP servers.

Linked Data servers Perhaps the most well-known alternative interface to triples is described by the Linked Data principles. The principles require servers to publish documents with triples about specific entities, which the client can access through their entity-specific URI, a process which is called *dereferencing*. Each of these Linked Data documents should contain data that mention URIs of other entities, which can be dereferenced in turn. Several Linked Data querying techniques [4] use dereferencing to solve queries over the Web of Data. This process happens client-side, so the availability of servers is not impacted. However, execution times are high, and many queries cannot be solved (efficiently) [6].

Other HTTP interfaces for triples Additionally, several other HTTP interfaces for triples have been designed. Strictly speaking, the most trivial HTTP interface is a data dump, which is a single-file representation of a dataset. The Linked Data Platform [5] is a read/write HTTP interface for Linked Data, scheduled to become a W3C recommendation. It details several concepts that extend beyond the Linked Data principles, such as containers and write access. However, the API has been designed primarily for consistent read/write access to Linked Data resources, not to enable reliable and/or efficient query execution. The interface we will discuss next offers low-cost publishing and client-side querying.

3 Linked Data Fragments and Triple Pattern Fragments

Linked Data Fragments [6] enable a uniform view on all possible HTTP interfaces for triples, and allow to define new interfaces with different trade-offs.

Definition 1. A *Linked Data Fragment* (LDF) of a dataset is a resource consisting of those triples of this dataset that match a specific selector, together with their metadata and hypermedia controls to retrieve other Linked Data Fragments.

We define a specific type of LDFs that require minimal effort to generate by a server, while still enabling efficient querying on the client side:

Definition 2. A *triple pattern fragment* is a Linked Data Fragment with a triple pattern as selector, count metadata, and the controls to retrieve any other triple pattern fragment of the dataset. Each **page** of a triple pattern fragment contains a subset of the matching triples, together with all metadata and controls.

Triple pattern fragments can be generated easily, as triple-pattern selection is an indexed operation in the majority of triple stores. Furthermore, specialized formats such as the compressed RDF HDT (Header – Dictionary – Triples [2]) natively support fast triple-pattern extraction. This ensures low-cost servers.

Clients can then efficiently evaluate SPARQL queries over the remote dataset because each page contains an estimate of the total number of matching triples. This allows efficient asymmetric joins by first binding those triple patterns with the lowest number of matches. For basic graph patterns (BGPs), which are the main building blocks of SPARQL queries, the algorithm works as follows:

1. For each triple pattern tp_i in the BGP $B = \{tp_1, \dots, tp_n\}$, fetch the first page ϕ_1^i of the triple pattern fragment f_i for tp_i , which contains an estimate cnt_i of the total number of matches for tp_i . Choose ϵ such that $cnt_\epsilon = \min(\{cnt_1, \dots, cnt_n\})$. f_ϵ is then the optimal fragment to start with.
2. Fetch all remaining pages of the triple pattern fragment f_ϵ . For each triple t in the LDF, generate the solution mapping μ_t such that $\mu_t(tp_\epsilon) = t$. Compose the subpattern $B_t = \{tp \mid tp = \mu_t(tp_j) \wedge tp_j \in B\} \setminus \{t\}$. If $B_t \neq \emptyset$, find mappings Ω_{B_t} by recursively calling the algorithm for B_t . Else, $\Omega_{B_t} = \{\mu_\emptyset\}$ with μ_\emptyset the empty mapping.
3. Return all solution mappings $\mu \in \{\mu_t \cup \mu' \mid \mu' \in \Omega_{B_t}\}$.

4 Demo of Client-side Querying

The above recursive algorithm has been implemented by a dynamic pipeline of iterators [6]. At the deepest level, a client uses `TriplePatternIterators` to retrieve pages of triple pattern fragments from the server, turning the triples on those pages into bindings. A basic graph pattern of a SPARQL query is evaluated by a `GraphPatternIterator`, which first discovers the triple pattern in this graph with the lowest number of matches by fetching the first page of the corresponding triple pattern fragment. Then, `TriplePatternIterators` are recursively chained together in the optimal order, which is chosen dynamically based on the number of matches for each binding. More specific iterators enable other SPARQL features.

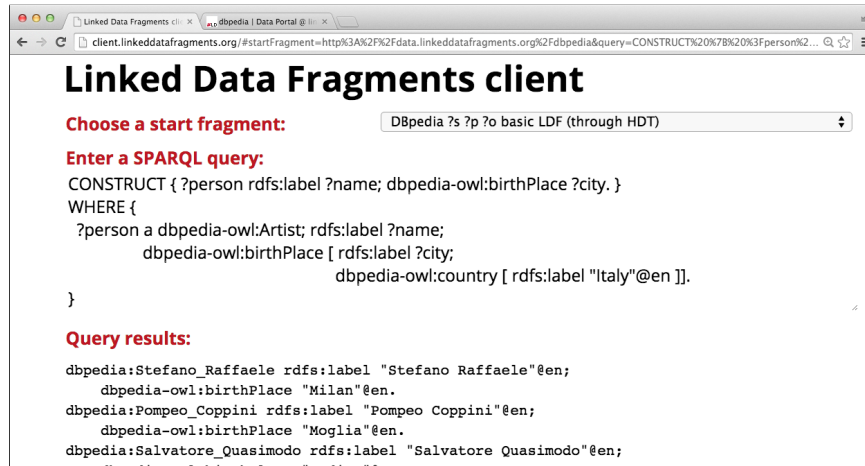


Fig. 1. The demo shows how Linked Data Fragments clients, such as Web browsers, evaluate SPARQL queries over datasets offered as inexpensive triple pattern fragments. In the above example, a user searches for artists born in Italian cities.

This iterator-based approach has been implemented as a JavaScript application (Fig. 1), to allow its usage on different platforms (standalone, library, browser application). The source code of the client, and also of triple pattern fragment servers, is freely available at <https://github.com/LinkedDataFragments/>. The versatility and efficiency of client-side querying is demonstrated through the Web application <http://client.linkeddatafragments.org>, which allows users to execute arbitrary SPARQL queries over triple pattern fragments. That way, participants experience first-hand how low-cost Linked Data publishing solutions can still enable efficient, realtime query execution over datasets on the Web.

References

1. Buil-Aranda, C., Hogan, A., Umbrich, J., Vandenbussche, P.Y.: SPARQL Web-querying infrastructure: Ready for action? In: Proceedings of the 12th International Semantic Web Conference (Nov 2013)
2. Fernández, J.D., Martínez-Prieto, M.A., Gutiérrez, C., Polleres, A., Arias, M.: Binary RDF representation for publication and exchange (HDT). *Journal of Web Semantics* 19, 22–41 (Mar 2013)
3. Harris, S., Seaborne, A.: SPARQL 1.1 query language. Recommendation, w3C (Mar 2013), <http://www.w3.org/TR/sparql11-query/>
4. Hartig, O.: An overview on execution strategies for Linked Data queries. *Datenbank-Spektrum* 13(2), 89–99 (2013)
5. Speicher, S., Arwe, J., Malhotra, A.: Linked Data Platform 1.0. Working draft, w3C (Mar 2014), <http://www.w3.org/TR/2014/WD-ldp-20140311/>
6. Verborgh, R., Hartig, O., De Meester, B., Haesendonck, G., De Vocht, L., Vander Sande, M., Cyganiak, R., Colpaert, P., Mannens, E., Van de Walle, R.: Querying datasets on the Web with high availability. In: Proceedings of the 13th International Semantic Web Conference (Oct 2014)

call: A Nucleus for a Web of Open Functions

Maurizio Atzori

Math/CS Department
University of Cagliari
Via Ospedale 72
09124 Cagliari (CA), Italy
atzori@unica.it

Abstract. In our recent work we envisioned a Web where functions, like Linked Data, can be openly published and available to all the users of any remote SPARQL endpoint. The resulting *Web of Functions* can be realized by introducing a `call` SPARQL extension that can invoke any remote function (custom third-party extensions) by only knowing its corresponding URI, while the implementation and the computational resources are made available by the function publisher. In this paper we demo our framework with a set of functions showing (1) advanced use of its higher-order expressivity power featuring, e.g., function composition of third-party functions, and (2) a possible bridge between hundreds of standard Web APIs and the Web of Functions. In our view these functions found an initial nucleus to which anyone can contribute within the decentralized Web of Functions, made available through `call`.

1 Introduction

While extending the language with user-defined custom functions (sometimes called extension functions) represented by URIs is a native feature of the SPARQL language, the mechanism only works on the single endpoint featuring that specific function. In our recent work [1], we investigated interoperability, computational power and expressivity of functions that can be used within a SPARQL query, envisioning a Web where also functions can be openly published, making them available to all the users of any other endpoint. In [1] we define a `wfn:call` function with three possible architectures to deploy it in a backward compatible manner. It is the basis needed in order to realize a *Web of Open Functions*, meaning that users can call a function by only knowing its corresponding URI, as it is the case for entities and properties in the Web of Open Data¹, while the implementation and the computational resources are made available by the function publisher, as it happens with usual Web APIs.

Practically, supposing that Alice wants to use a Bob's SPARQL extension (only defined in Bob's endpoint) from her own endpoint, she will write the following:

```
PREFIX wfn: <http://webofcode.org/wfn/>
```

¹ We titled this paper after DBpedia milestone work in [2].

```

PREFIX bob: <http://bob-server.org/fn/>
SELECT *
WHERE {
    # within Alice data, find useful values for ?arg1, ?arg2
    ...
    # now use Bob's function
    FILTER(wfn:call(bob:complexFunction, ?arg1, ?arg2) )
}

```

Therefore, the function `wfn:call` takes care of finding the right endpoint (see [1, 3, 4]), i.e., Bob's, and then remotely call Bob's `complexFunction`. We believe this may be the first step toward a novel view of the Web as a place holding code and functions, not only data as the Linked Data is greatly doing. The Semantic Web already shifted URIs from pages to conceptual entities, primarily structured data. We believe that among these concepts there should be computable functions.

In this paper we demo our open source implementation for the `wfn:call` function, realized as an Apache Jena's custom function extension, and available with other resources (including a link to our endpoint that publishes it) at <http://atzori.webofcode.org/projects/wfn/>. In particular, we devise an initial nucleus of practical functions that may empower SPARQL queries with computations that exploit higher-order expressivity and hundreds of existing Web APIs.

2 Fully Higher-Order Functions in SPARQL

Higher-order functions (HOF) are functions that take functions as either input and/or output. Languages that allow functions to be used as any other kind of data, are said to feature first-class functions. Here we show that these advanced expressivity, typical of functional languages, can be used within SPARQL by means of `wfn:call`. In the following we exemplify it by describing the use of three important HOF: reduce, compose and memoize.

Reduce. In functional languages “reduce” (also called “fold”, “inject” or “aggregate”) is a function that takes a binary function (e.g., the + operator) and a list (e.g., a list of integers), producing a result by recursively applying the function to the remaining list (providing, e.g., the sum of all the elements). Thus, it represents a general-purpose aggregation mechanism potentially useful in SPARQL queries. In the following we show how it can be used to apply the binary max function provided by Jena to a list of 4 numbers:

```

PREFIX call: <http://webofcode.org/wfn/call>
PREFIX afn: <http://jena.hpl.hp.com/ARQ/function#>.

SELECT ?max {
    BIND( call:(wfn:reduce, afn:max, 5, 7, -1, 3) AS ?max)
}

```

resulting in `?max = 7`.

Compose. Another important HOF is the composition function. Given two functions g and f , it returns a third function that behaves as the application of f followed by the application of g , i.e., $g(f(\cdot))$. The following query excerpt:

```

BIND(call:(wfn:compose, fn:upper-case, afn:namespace)
AS ?uppercase_ns).
BIND(call:(?uppercase_ns, <http://something.org/myentity>) AS ?result)

```

returns the uppercased namespace, that is, `HTTP://SOMETHING.ORG/`. In particular, variable `?uppercase_ns` is binded to a dynamically generated SPARQL function that, whenever invoked, applies `afn:namespace` followed by `fn:upper-case`.

Memoize. Many SPARQL queries may iterate over intermediate results, requiring the execution of the same function multiple times, possibly with the same parameters. In order to speed up the execution of potentially time-consuming functions, we implemented a memoization function that keeps the results of function calls and then returns the cached result when the same inputs occur again. This part of the query:

```

BIND(call:(wfn:memoize, ?slow_function) AS ?fast_function).
BIND(call:(?fast_function, 1) AS ?result1). #1st time at normal speed
BIND(call:(?fast_function, 1) AS ?result2). #2nd time is faster

```

dynamically generates a `?fast_function` that is the memoization of function in `?slow_function`. Please notice that this kind of useful features are possible only in higher-order environments, such as the one resulting by the use of our `call` function [1].

3 Bridging Web APIs and the Web of Functions

In order to develop a useful Web of Functions, the small set of auxiliary functions presented in the previous section are clearly not enough. While some other powerful user-defined functions are already online (e.g., `runSPARQL` [5] computes recursive SPARQL functions), we need a larger nucleus of functions allowing any sort of computation from within SPARQL queries. In this section we propose the exploitation of a well-known Web API hub, namely *Mashape*², by using a simple bridge function that allows to call any mashape-featured API. This function, that we called `wfn:api-bridge`, pushes hundreds of Web APIs within the Web of Functions, ranging from weather forecast to face detection, from language translation to flight information lookup. For instance, we can iterate over DBpedia cities in Tuscany finding those with a close airport, cheap flight and good weather during the week after a given arrival day. In the following we find large Tuscany cities sorted by current weather temperature:

```

SELECT * {
  ?city dbpedia-owl:region dbpedia:Tuscany;
        dbpedia-owl:populationTotal ?population;

```

² Freely available at <http://www.mashape.com/>

```

    geo:lat ?lat; geo:long ?long.

FILTER(?population > 80000).
BIND(CONCAT("lat=",?lat,"&lon=",?long) AS ?parameters)

BIND( call:(wfn:api-bridge, "community-open-weather-map", ?parameters,
    "main.temp") as ?temperature).
} ORDER BY ?temperature

```

The `wfn:api-bridge` function calls the Mashape Web API corresponding to the first argument, with parameters specified in the second argument, then returning the JSON field selected in the third argument. Different APIs necessary to answer the query can be combined together with `compose`, and the resulting function may be memoized for better performance if needed. Advanced uses may exploit Linked Data information to search through existing Web API repositories [6].

4 Conclusions and Demo Showcase

We presented a set of SPARQL extensions containing higher-order manipulation functions, allowing for instance function composition, together with a bridge function that allows the use of hundreds of existing Web APIs from any SPARQL endpoint featuring the `wfn:call` function, that we developed and opensourced for Apache Jena. This set, forming an initial nucleus for the *Web of Functions*, enables a wide spectrum of much powerful SPARQL queries w.r.t. the ones we are currently used to, with a number of practical examples that will be showcased during the demo and made available at our website.

Acknowledgments. This work was supported in part by the RAS Project CRP-17615 *DENIS: Dataspaces Enhancing Next Internet in Sardinia* and by MIUR PRIN 2010-11 project *Security Horizons*.

References

1. Atzori, M.: Toward the Web of Functions: Interoperable Higher-Order Functions in SPARQL. In: 13th International Semantic Web Conference (Research Track). (2014)
2. Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z.G.: DBpedia: A Nucleus for a Web of Open Data. In: The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference (ISWC/ASWC). (2007) 722–735
3. Paulheim, H., Hertling, S.: Discoverability of SPARQL Endpoints in Linked Open Data. In: International Semantic Web Conference (Posters & Demos). (2013)
4. Alexander, K., Cyganiak, R., Hausenblas, M., Zhao, J.: Describing Linked Datasets with the VoID Vocabulary (December 2010)
5. Atzori, M.: Computing Recursive SPARQL Queries. In: 8th IEEE International Conference on Semantic Computing. (2014)
6. Bianchini, D., Antonellis, V.D., Melchiori, M.: A Linked Data Perspective for Effective Exploration of Web APIs Repositories. In: ICWE 2013. (2013) 506–509

Cross-lingual detection of world events from news articles

Gregor Leban, Blaž Fortuna, Janez Brank, and Marko Grobelnik

Jožef Stefan Institute
Ljubljana, Slovenia
{firstname.surname}@ijs.si

Abstract. In this demo we describe a system called Event Registry (<http://eventregistry.org>) that can identify world events from news articles. Events can be detected in different languages. For each event, the system can extract core event information and store it in a structured form that allows advanced search options. Numerous visualizations are provided for visualizing search results.

Keywords: information extraction; cross-linguality; events; named entity detection; clustering; news

1 Introduction

Each day there are thousands of small, medium and large events that are happening in the world. These events range from insignificant meetings, conferences and sport events to important natural disasters and decisions made by world leaders. The way we learn about these events is from different media channels. The unstructured nature of content provided on these channels is suitable for humans, but hard to understand by computers.

Identifying events described in the news and extracting main information about these events is the main goal of the system presented in this paper. Event Registry [2] is able to process news articles published in different languages worldwide. By analyzing the articles it can identify the mentioned events and extract main event information. Extracted event information is stored in a structured way that provides unique features such as searching for events by date, location, entity, topic and other event properties. Beside finding events of interest, Event Registry also provides user interface with numerous visualizations showing date, location, concept and topic aggregates of events that match the search criteria.

The rest of the paper is organized as follows. We start by describing the high level architecture of the system. Next, we describe in more details the process of identification of events from individual news articles. We continue by providing information about how event information is extracted from a group of articles mentioning the event. In the end we also describe the main features of the frontend interface of the Event Registry.

2 System architecture

Event Registry consists of a pipeline of components, that each provide unique and relevant features for the system. In order to identify events we first need data from which we can extract the information. In Event Registry we use news articles as the data source. The news articles are collected using the News-Feed [5] service which collects news articles from more than 75.000 worldwide news sources. Collected articles are in more than 40 languages, where articles in English, Spanish, German and Chinese languages amount to about 70% of all articles. These languages are also the only ones we use in the Event Registry.

Each collected article in the mentioned languages is then analyzed in order to extract relevant information. One of the key tasks is identification and disambiguation of named entities and topics mentioned in the article. We perform this task using a semantic enrichment service developed in the XLike project. We also detect date mentions in the text, since they frequently specify the date when the event occurred. By analyzing the articles we noticed that different news sources frequently publish almost identical news articles. These duplicated articles don't bring any new information, which is why we identify them and ignore them in the rest of the pipeline.

An important feature of the Event Registry is cross-linguality. To support it, we identify for each article a set of most similar articles in other languages. To identify these articles we use canonical correlation analysis[4] which maps articles from different languages into a common semantic space. The common space can then be used to identify most similar articles in all other languages. In order to train the mapping to the common space we used the aligned corpus of Wikipedia articles.

After extracting relevant features from each individual article we start with the main task of identifying events from groups of articles. Since this is the main contribution of the paper we will describe the details of the process in the next three sections.

3 Identification of events

An assumption that we make in identifying events is that any relevant event should be reported at least by a few different news publishers. In order to identify events we therefore apply an online clustering algorithm based on [1] on articles as they are added into the system. Each article is first transformed into a TF-IDF weighted feature vector. The features in the vector are the words in the document as well as the identified named entities and topics. If the cosine similarity of the closest centroid is above a threshold, the article is added to the closest cluster and the cluster properties are updated. Otherwise, a new cluster is formed that contains only the new article.

Each identified cluster of articles is considered to describe an event if it contains at least a minimum number of articles (the minimum value used in our system is 5 articles). Once a cluster reaches this limit, we treat it as an

event and the information about it's articles are sent to the next components in the pipeline. Those components are responsible for extracting event information from the articles and will be described in the next sections.

Since clusters are being constantly updated with new articles we want to reevaluate each cluster after a few updates in order to determine if it should be split into two clusters or merged with another cluster. In order to decide if the cluster should be split we apply a bisecting k-means algorithm (with $k = 2$) on the cluster. We then use a variant of the Bayesian Information Criterion to decide whether to accept the new split or not. Periodically we also identify pairs of clusters with high similarity and decide if they should be merged or not. The decision is made using the Lughofer's ellipsoid criterion [3]. We assume that the clusters that have not been modified for a few days mention past events and we therefore remove them from the list of maintained clusters.

3.1 Cross-lingual merging of clusters

Each identified cluster of articles contains only articles in a single language. Since articles in different languages can describe the same events we want to identify clusters describing the same event in different languages and represent them as a single event. In order to determine which cluster pairs to merge we represent the task as a binary classification problem. Given a cluster pair c_1 and c_2 in languages l_1 and l_2 we want to extract a set of features that would discriminate between cluster pairs that describe the same event and those that don't. A very important learning feature that we can extract for each cluster pair is computed by inspecting individual articles in each cluster. Using canonical correlation analysis we are able to obtain for each article in c_1 a list of 10 most similar articles in language l_2 . Using this information we can check how many of these articles are in c_2 . We can repeat the same computation for articles in c_2 . By normalizing the results by the size of the clusters we can obtain a score that should by intuition correlate with similarity of the two clusters across the two languages. Some of the other features that we extract include the time difference between the average article date of each cluster, cosine similarity of the annotated concepts and topics, and category similarity.

In order to build a classification model we collected 85 learning examples. Some cluster pairs were selected randomly and some were selected by users based on their similarity. The features for the selected learning examples were extracted automatically, while the correct class was assigned manually. A linear SVM was then trained on the data which achieved 87% accuracy using 10-fold cross validation.

4 Event information extraction

Once we obtain one or more new clusters that are believed to describe a single event, we assign them a new id in the Event Registry. From the associated articles we then try to extract the relevant information about the event. We try to

determine event date by seeing if there is a common date reference frequently mentioned in the articles. If no date is mentioned frequently enough we use the average article’s published date as the event date. The location of the event is determined by locating frequently mentioned named entities that represent locations. To determine what the event is about we aggregate the named entities and topics identified in the articles. Each event is also categorized using a DMoz taxonomy. All extracted information is stored in the Event Registry in a structured form that provides rich search and visualization capabilities.

5 Event search, visualization options and data accessibility

Event Registry is available at <http://eventregistry.org> and currently contains 15.000.000 articles from which we identified about 1 million events. Available search options include search by relevant named entities, keywords, publishers, event location, date and category. The resulting events that match the criteria can then be seen as a list or using one of numerous visualizations. Main visualizations of search results include location and time aggregates, list of top named entities and topics, graph of related entities, concept trends, concept matrix, date mentions, clusters of events and event categories. For each individual event we can provide the list of articles describing it as well as visualizations of concepts, article timeline, date mentions, article sources and other similar events. Examples of these visualizations are (due to space limit) available on <http://eventregistry.org/screens>. All Event Registry data is also stored using the Storyline ontology and is available through a SPARQL endpoint available at <http://eventregistry.org/sparql>.

6 Acknowledgments

This work was supported by the Slovenian Research Agency and X-Like (ICT-288342-STREP).

References

1. C. C. Aggarwal and P. Yu. A framework for clustering massive text and categorical data streams. In *Proceedings of the sixth SIAM international conference on data mining*, volume 124, pages 479–483, 2006.
2. G. Leban, B. Fortuna, J. Brank, and M. Grobelnik. Event registry – learning about world events from news. In *WWW 2014 Proceedings*, pages 107–110. ACM, 2014.
3. E. Lughofer. A dynamic split-and-merge approach for evolving cluster models. *Evolving Systems*, 3(3):135–151, 2012.
4. J. Rupnik, A. Muhic, and P. Skraba. Cross-lingual document retrieval through hub languages. In *NIPS*, 2012.
5. M. Trampus and B. Novak. Internals of an aggregated web news feed. In *Proceedings of 15th Multiconference on Information Society 2012 (IS-2012)*, 2012.

Multilingual Word Sense Disambiguation and Entity Linking for Everybody

Andrea Moro, Francesco Cecconi, and Roberto Navigli

Sapienza University of Rome, Viale Regina Elena 295, 00198, Italy
{moro,cecconi,navigli}@di.uniroma1.it

Abstract. In this paper we present a Web interface and a RESTful API for our state-of-the-art multilingual word sense disambiguation and entity linking system. The Web interface has been developed, on the one hand, to be user-friendly for non-specialized users, who can thus easily obtain a first grasp on complex linguistic problems such as the ambiguity of words and entity mentions and, on the other hand, to provide a showcase for researchers from other fields interested in the multilingual disambiguation task. Moreover, our RESTful API enables an easy integration, within a Java framework, of state-of-the-art language technologies. Both the Web interface and the RESTful API are available at <http://babelfy.org>

Keywords: Multilinguality, Word Sense Disambiguation, Entity Linking, Web interface, RESTful API

1 Introduction

The tasks of Word Sense Disambiguation (WSD) and Entity Linking (EL) are well-known in the computational linguistics community. WSD [9, 10] is a historical task aimed at assigning meanings to single-word and multi-word occurrences within text, while the aim of EL [3, 12] is to discover mentions of entities within a text and to link them to the most suitable entry in the considered knowledge base. These two tasks are key to many problems in Artificial Intelligence and especially to Machine Reading (MR) [6], i.e., the problem of automatic, unsupervised understanding of text. Moreover, the recent upsurge of interest in the use of semi-structured resources to create novel repositories of knowledge [5] has opened up new opportunities for wide-coverage, general-purpose Natural Language Understanding techniques. The next logical step, from the point of view of Machine Reading, is to link natural language text to the aforementioned resources.

In this paper, we present a Web interface and a Java RESTful API for our state-of-the-art approach to WSD and EL in arbitrary languages: Babelfy [8]. Babelfy is the first approach which explicitly aims at performing both multilingual WSD and EL at the same time. The approach is knowledge-based and exploits semantic relations between word meanings and named entities from BabelNet [11], a multilingual semantic network which provides lexicalizations and glosses for more than 9 million concepts and named entities in 50 languages.

2 BabelNet

In our work we use the BabelNet 2.5¹ semantic network [11] since it is the largest available multilingual knowledge base and is obtained from the automatic seamless integration of Wikipedia², WikiData³, OmegaWiki⁴, WordNet [4], Open Multilingual WordNet [1] and Wiktionary⁵. It is available in different formats, such as via its Java API, a SPARQL endpoint and a linked data interface [2]. It contains more than 9 million concepts and named entities, 50 million lexicalizations and around 250 million semantic relations (see <http://babelnet.org/stats> for more detailed statistics). Moreover, by using this resource we can leverage the multilingual lexicalizations of the concepts and entities it contains to perform disambiguation in any of the 50 languages covered in BabelNet.

3 The Babelfy System

Our state-of-the-art approach, Babelfy [8], is based on a loose identification of candidate meanings (substring matching instead of exact matching) coupled with a densest subgraph heuristic which selects high-coherence semantic interpretations. Here we briefly describe its three main steps:

1. Each vertex, i.e., either concept or named entity, is automatically associated with a semantic signature, that is, a set of related vertices by means of random walks with restart on the BabelNet network.
2. Then, given an input text, all the linkable fragments, i.e., pieces of text being equal to or substring of at least one lexicalization contained in BabelNet, are selected and, for each of them, the possible meanings are listed according to the semantic network.
3. A graph-based semantic interpretation of the whole text is produced by linking the candidate meanings of the selected fragments using the previously-computed semantic signatures. Then a densest subgraph heuristic is used to extract the most coherent interpretation and finally the fragments are disambiguated by using a centrality measure within this graph.

A detailed description and evaluations of the approach are given in [7, 8].

4 Web Interface and RESTful API

We developed a Web interface and a RESTful API by following the KISS principle, i.e., “keep it simple, stupid”. As can be seen from the screenshot in Figure

¹ <http://babelnet.org>

² <http://www.wikipedia.org>

³ <http://wikidata.org>

⁴ <http://omegawiki.org>

⁵ <http://wiktionary.org>



Fig. 1. A screenshot of the Babelify Web interface.

1, the Web interface asks for the input text, its language and whether the partial matching heuristic should be used instead of the exact string matching one. After clicking on “Babelify!” the user is presented with the annotated text where we denote with green circles the concepts and with yellow circles the named entities. As for the Java RESTful API, users can exploit our approach by writing less than 10 lines of code. Here we show a complete example:

```
// get an instance of the Babelify RESTful API manager
Babelify bfy = Babelify.getInstance(AccessType.ONLINE);
// the string to be disambiguated
String inputText = "hello world, I'm a computer scientist";
// the actual disambiguation call
Annotation annotations = bfy.babelify("", inputText,
    Matching.EXACT, Language.EN);
// printing the result
for(BabelSynsetAnchor annotation : annotations.getAnnotations())
    System.out.println(annotation.getAnchorText()+"\t"+
        annotation.getBabelSynset().getId()+"\t"+
        annotation.getBabelSynset());
```

4.1 Documentation for the RESTful API

```
Annotation babelify(String key, String inputText,
    Matching candidateSelectionMode, Language language)
```

The first parameter is the access key. A random or empty key will grant 100 requests per day (but a less restrictive key can be requested). The second parameter is a string representing the input text (sentences or whole documents can be input up to a maximum of 3500 characters). The third parameter is an enum with two possible values: EXACT or PARTIAL, to enable, respectively, the exact or partial matching heuristic for the selection of fragment candidates found in the input text. The fourth parameter is the language of the input text (among 50 languages denoted with their ISO 639-1 uppercase code).

Annotation is the object that contains the output of our system. A user can access the POS-tagged input text with `getText()` which returns a list of `WordLemmaTag` objects with the respective getters. With `getAnnotations()` a user will get a list of `BabelSynsetAnchor` objects, i.e., the actual annotations. A user can use `getAnchorText()` to get the disambiguated fragment of text and with `getBabelSynset()` get the selected Babel synset. Moreover, if a user wants to anchor the disambiguated entry to the input text, the start and end indices of the tagged text can be gotten with `getStart()` and `getEnd()`.

5 Conclusion

In this paper, we presented and described the typical use of the Web interface and Java RESTful API of our state-of-the-art system for multilingual Word Sense Disambiguation and Entity Linking, i.e., Babelfy, available at <http://babelfy.org>

Acknowledgments



The authors gratefully acknowledge the support of the ERC Starting Grant MultiJEDI No. 259234.



References

1. Bond, F., Foster, R.: Linking and extending an open multilingual wordnet. In: Proc. of ACL. pp. 1352–1362 (2013)
2. Ehrmann, M., Cecconi, F., Vannella, D., Mccrae, J.P., Cimiano, P., Navigli, R.: Representing Multilingual Data as Linked Data: the Case of BabelNet 2.0. In: Proc. of LREC. pp. 401–408 (2014)
3. Erbs, N., Zesch, T., Gurevych, I.: Link Discovery: A Comprehensive Analysis. In: Proc. of ICSC. pp. 83–86 (2011)
4. Fellbaum, C.: WordNet: An Electronic Lexical Database. MIT Press (1998)
5. Hovy, E.H., Navigli, R., Ponzetto, S.P.: Collaboratively built semi-structured content and Artificial Intelligence: The story so far. *Artificial Intelligence* 194, 2–27 (2013)
6. Mitchell, T.M.: Reading the Web: A Breakthrough Goal for AI. *AI Magazine* (2005)
7. Moro, A., Navigli, R., Tucci, F.M., Passonneau, R.J.: Annotating the MASC Corpus with BabelNet. Proc. of LREC pp. 4214–4219 (2014)
8. Moro, A., Raganato, A., Navigli, R.: Entity Linking meets Word Sense Disambiguation: A Unified Approach. *TACL* 2, 231–244 (2014)
9. Navigli, R.: Word sense disambiguation: A survey. *ACM Comput. Surv.* 41(2), 1–69 (2009)
10. Navigli, R.: A Quick Tour of Word Sense Disambiguation, Induction and Related Approaches. In: Proc. of SOFSEM. pp. 115–129 (2012)
11. Navigli, R., Ponzetto, S.P.: BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial Intelligence* 193, 217–250 (2012)
12. Rao, D., McNamee, P., Dredze, M.: Entity linking: Finding extracted entities in a knowledge base. In: Multi-source, Multilingual Information Extraction and Summarization, pp. 93–115 (2013)

Help me describe my data: A demonstration of the Open PHACTS VoID Editor

Carole Goble¹, Alasdair J G Gray², and Eleftherios Tatakis¹

¹ School of Computer Science, University of Manchester, Manchester, UK

² Department of Computer Science, Heriot-Watt University, Edinburgh, UK

Abstract. The Open PHACTS VoID Editor helps non-Semantic Web experts to create machine interpretable descriptions for their datasets. The web app guides the user, an expert in the domain of the data, through a series of questions to capture details of their dataset and then generates a VoID dataset description. The generated dataset description conforms to the Open PHACTS dataset description guidelines that ensure suitable provenance information is available about the dataset to enable its discovery and reuse.

The VoID Editor is available at <http://voideditor.cs.man.ac.uk>.

The source code can be found at

<https://github.com/openphacts/Void-Editor2>.

Keywords: Dataset descriptions, VoID, Provenance, Metadata

1 Motivating Problem

Users of systems such as the Open PHACTS Discovery Platform³ [1,2] need to know which datasets have been integrated. In the scientific domain they particularly need to know which version of a dataset is loaded in order to correctly interpret the results returned by the platform. To satisfy this need, the provenance of the datasets loaded into the Open PHACTS Discovery Platform are needed. This provenance information is then available for any data returned by the platform's API. Within the Open PHACTS project we have identified a minimal set of metadata that should be provided to aid understanding and reuse of the data [3]. Additionally, we recommend that the metadata is provided using the VoID vocabulary [4] so that the data is self-describing and machine processable.

Open PHACTS does not publish its own datasets; it integrates existing publicly available domain data. Typically the publishers of these scientific data sets are experts in their scientific domain, viz. chemistry or biology, but not in the semantic web. They need to be supported in the creation of VoID descriptions of their datasets which may have been published in a database and converted into RDF. A tool which hides the underlying details of the semantic web but enables the creation of descriptions understandable to a domain expert is thus needed.

³ <https://dev.openphacts.org/> accessed July 2014

Fig. 1: Screenshot of the VoID Editor

2 VoID Editor

The aim of the VoID Editor (see screenshot in Figure 1) is to allow a data publisher to create validated dataset descriptions within 30 minutes. In particular, the data publisher does not need to read and understand the Open PHACTS dataset descriptions guidelines [3] which provide a checklist of the RDF properties that MUST and SHOULD be provided. There is also no need for the data publisher to understand RDF or the VoID vocabulary.

The VoID Editor is a web application that guides the data provider through a series of questions to acquire the required metadata properties. The user is first asked for details about themselves and other individuals involved in the authoring of the data. Core publishing metadata such as the publishing organisation and the license are then gathered. The user is then asked for versioning information and the expected update frequency of the data. The Sources tab helps the user to provide details of source datasets from which their data is derived. They can either select from the datasets already known to the Open PHACTS Discovery Platform or enter the details manually. The list of known datasets is populated by a call to the Open PHACTS API. The Distribution Formats tab allows the user to describe the distributions in which the data is provided, e.g. RDF, database dump, or CSV. The final screen allows the user to export the RDF of their dataset description as well as providing a summary of any validation errors, e.g. not supplying a license which is a required field, such errors

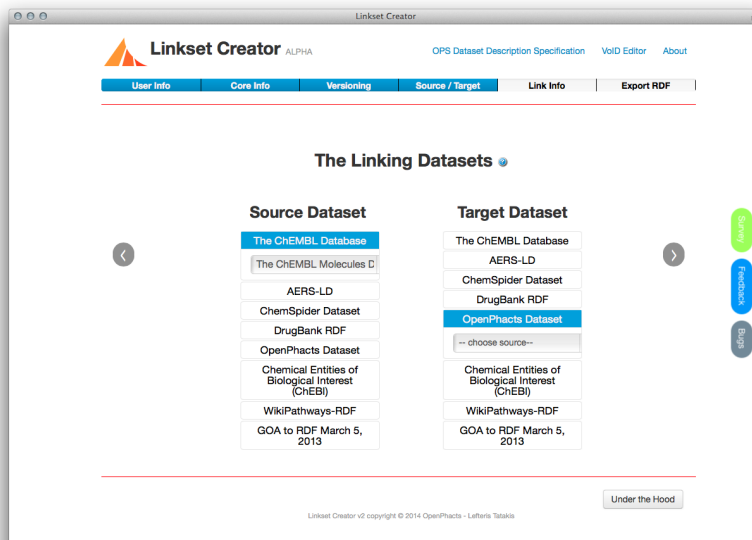


Fig. 2: Screenshots of the Linkset Editor

will already have been indicated by a red bar at the top of the screen containing an error message. Note that the ‘Export RDF’ button is only activated when a valid dataset description can be created, i.e. all required fields have been filled in.

At any stage, the generated RDF dataset description may be inspected by clicking the ‘Under the Hood’ button. This button can also be used to save a partially generated description that can later be imported into the editor through the ‘Import VoID’ button. The ‘Under the Hood’ feature is also useful for semantic web experts to see what is being generated at any stage.

3 Linkset Editor

In companion with the VoID Editor, a Linkset Editor (see screenshot in Figure 2) has been developed. The Linkset Editor allows for the creation of descriptions of the links between two datasets. The same interface design and framework is used.

The Linkset Editor reuses the first three tabs of the VoID Editor to capture details of the authors, core publishing information, and details about versioning. The Source/Target tab allows the user to select the pair of datasets that are connected by the linkset. Again, the list of possible datasets is generated by a call to the Open PHACTS API. The Link Info tab asks the user to declare

the link predicate used in the linkset and provide some justification to capture the nature of the equality relationship encoded in the links. (For details about linkset justifications, please see Section 5 of [3].)

4 Implementation

The VoID and Linkset Editors have been implemented using AngularJS as a Javascript framework for the web client with a server implementation using Jena libraries. A user-centric approach was followed for the design and development of the VoID Editor. A small number of data providers were consulted about the type of tool they required with regular interviews and feedback on prototype versions. A larger number of potential users were involved in an evaluation of the VoID Editor. Full details can be found in [5].

In the future we plan to investigate how the VoID Editor can generate template descriptions that can be populated as part of the data publishing pipeline. We also plan to look at how the editor could be adapted to other dataset description guidelines, e.g. DCAT⁴ or the W3C HCLS community profile⁵. However, this is not a straightforward process since considerable care and attention is paid to the phrasing and grouping of questions to ensure a pleasant user experience.

Acknowledgements

The research has received support from the Innovative Medicines Initiative Joint Undertaking under grant agreement number 115191, resources of which are composed of financial contribution from the European Union's Seventh Framework Programme (FP7/2007- 2013) and EFPIA companies in kind contribution.

References

1. Gray, A.J.G., Groth, P., Loizou, A., Askjaer, S., Breninkmeijer, C.Y.A., Burger, K., Chichester, C., Evelo, C.T., Goble, C.A., Harland, L., Pettifer, S., Thompson, M., Waagmeester, A., Williams, A.J.: Applying linked data approaches to pharmacology: Architectural decisions and implementation. *Semantic Web* **5**(2) (2014) 101–113 doi:10.3233/SW-2012-0088.
2. Groth, P., Loizou, A., Gray, A.J.G., Goble, C., Harland, L., Pettifer, S.: API-centric Linked Data Integration: The Open PHACTS Discovery Platform Case Study. *Journal of Web Semantics* (2014) In press. doi:10.1016/j.websem.2014.03.003.
3. Gray, A.J.G.: Dataset descriptions for the Open Pharmacological Space. Working draft, Open PHACTS (September 2013)
4. Alexander, K., Cyganiak, R., Hausenblas, M., Zhao, J.: Describing Linked Datasets with the VoID Vocabulary. Note, W3C (March 2011)
5. Tatakis, E.: VoID Editor v2. Undergraduate dissertation, School of Computer Science, University of Manchester, Manchester, UK (April 2014)

⁴ <http://www.w3.org/TR/vocab-dcat/> accessed July 2014

⁵ <http://www.w3.org/2001/sw/hcls/notes/hclsdataset/> access July 2014

OUSocial2 - A Platform for Gathering Students' Feedback from Social Media

Keerthi Thomas, Miriam Fernandez, Stuart Brown, and Harith Alani

Open University, UK

`first.last@open.ac.uk`, `h.alani@open.ac.uk`

Abstract. Universities strive to collect feedback from students to improve their courses and tutorship. Such feedback is often collected at the end of a course via survey forms. However, such methods in collecting feedback are too controlled, slow, and passive. With the rise of social media, many students are finding online venues to group and share their experiences and seek peers' support. OUSocial2 is a platform that monitors behaviour, sentiment, and topics, in open social media groups set up by, and for, Open University students. It captures anonymous feedback from students towards their courses, and tracks the evolution of engagement behaviour and sentiment within those groups.

Keywords: social media, behaviour analysis, sentiment analysis

1 Introduction

The Open University (OU) has around 250 thousand students, rendering it the largest university in the United Kingdom and one of the leading distance teaching institutions. Although the university provide students with several websites and applications where they can discuss their courses with their tutors and peers, many seem to be more engaged in such discussions on open social media platforms, such as Facebook groups.

Social media has become a rich source for student feedback, which could be collected and investigated, to capture any issues and problems in real time, as well as to monitor the engagement of students with their courses and peers. Students retention is especially challenging in distance learning, and close monitoring of students' activities and involvement can greatly help to predict churn of students, and thus giving their tutors an opportunity to intervene and support disengaging or struggling students [4].

OUSocial2 is a prototypical platform for collecting and analysing content from relevant and public Facebook groups, set up by OU students. These open groups have been set up to bring together other students who enrolled in particular OU courses or modules. OUSocial2 extends its predecessor which is described in [1], with a completely new interface, and lexicon-based sentiment tracking service.

More specifically, the objectives of the OUSocial2 project are:

1. Build a data collection service for gathering, storing, and integrating data from public Facebook groups related to OU
2. Develop and train a model for identifying the behaviour of individual users based on their activities and interactions in the Facebook online groups
3. Extract the topics that emerge in Facebook group discussions
4. Track the sentiment expressed about the specific topics by the group members

This paper describes the OUSocial2 platform’s architecture, analysis components, and data enrichment with the OUs linked data portal.

Demo: A fully working OUSocial2 platform will be demoed at the conference, running over 44 groups from Facebook, with a total of 172,695 posts from 19,759 users. Audience will be able to see how the various analyses components described below can be used to assess and monitor engagement of students in course groups, their evolving sentiment, and topics. For privacy reasons, the live demo is not publicly available yet. A video recording of the demo is available at:

<https://dl.dropboxusercontent.com/u/17906712/ousocial2-demo.mp4>

<https://dl.dropboxusercontent.com/u/17906712/ousocial2-demo.avi>

2 OUSocial2

In this section we describe the three main OUSocial2 analyses components and how their output is visualised in the demo. Facebook API is used to collect all posts and interactions from public groups about OU courses. 44 of such groups are identified by matching their titles to official OU course codes (e.g., T224). Collected data includes group ID, posts’ content, owner, time of posting, whether the post is a reply to another post, users, etc. Data collection is reactivated every 24 hours to update the database.

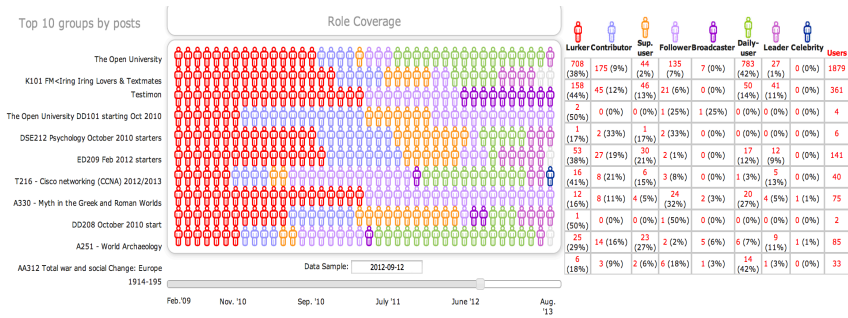


Fig. 1. Distribution of behaviour roles over time for several selected groups

2.1 Behaviour Analyser

This component applies our behaviour analyses service (see [3]) which uses machine learning and SPIN (spinrdf.org/) rules to identify the roles of users. Understanding the behaviour composition of a group, and the evolution of behavioural roles of individuals (micro) and groups (macro) is useful for assessing user engagement and future prospects [3, 2].

This component identifies eight types of roles; *Lurker*, *Follower*, *Daily User Contributor*, *Broadcaster*, *Leader*, *Celebrity* and *Super User*. Figure 1 is the OUSocial2 display of the role compositions of the top 10 active groups. The slide bar at the bottom is to view the roles at different points in time. The number and percentage of each type of role in a group is displayed on the right hand side. Engagement of particular group members can also be studied (see Figure 2).

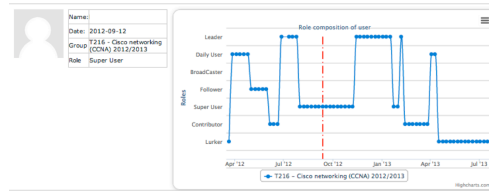


Fig. 2. User behaviour over time

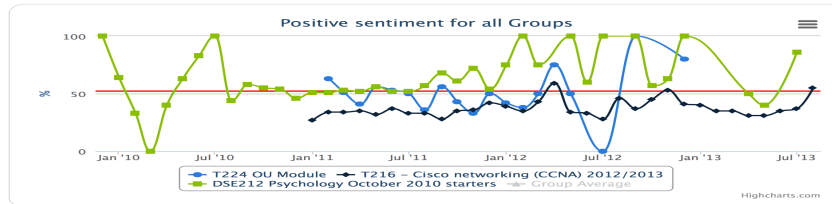


Fig. 3. Evolution of group sentiment over time. Red line is average sentiment across all groups.

2.2 Sentiment Analyser

The sentiment analysis component calculates the sentiment for each post. We use SentiStrength;¹ a lexicon-based sentiment analyser, to estimate the strength of positive and negative sentiments in our posts. We calculate sentiment at the community and member levels. OUSocial2 users can visualise and compare the evolution of sentiment in selected groups (Figure 3). Users can also see the sentiment distribution of a given group over time⁴, and upon clicking on a specific time point, the list of top positive, negative, and neutral posts are listed (Fig 5).

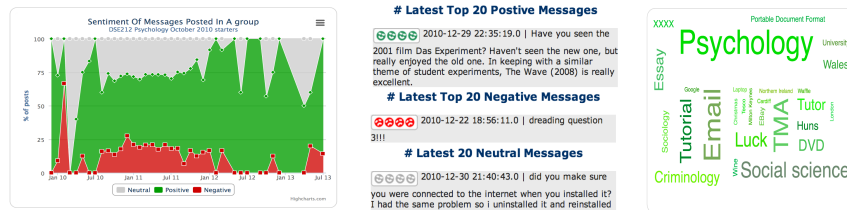


Fig. 4. Overall positive and negative sentiment levels in a group

Fig. 5. Topics appearing in posts with positive sentiment

Fig. 6. Topics in positive posts

2.3 Topic Analyser

Several named entity recognition systems have emerged recently, such as Textwise, Zemanta, DBpedia Spotlight, OpenCalais, Alchemy API, and TextRazor. OUSocial2 uses TextRazor since it seems to provide the best accuracy in our context. TextRazor

¹ <http://sentistrength.wlv.ac.uk/>

(textazor.com/) identifies entities from our posts, and returns the relevant URIs from DBpedia and Freebase, with confidence scores. Users of OUSocial2 can view the topics that appear in posts, in tag clouds of positive or negative entities (Fig. 6, to help them spot any issues or concerns raised by the members of these groups.

2.4 Data Enrichment

OU official information about all courses already exist as linked data from `data.open.ac.uk`. Course information can be SPARQLed to retrieve course titles, descriptions, topic categories, relevant courses, etc.

3 Feedback and Future Work

OUSocial2 was demonstrated to the university's executive board and strategy office, and was generally very well received as a tool that could enhance our collection of feedback, and speeding up our reaction to any concerns or challenges raised by students. Privacy was raised as an important issue, and further steps are planned to abstract any information that could lead to the identification of students. It was suggested that sentiment and engagement results could be compared to actual students' performance on the courses in question, as well as to their end-of-year feedback forms. Other requests include the implementation of alerts on abnormal activities in chosen groups (e.g., drop in engagement, rise in negative sentiment), and a comparison between groups on the same course but on different years.

Sentiment analysis was done using SentiStrength; a general-purpose lexicon-based tool. However, results showed that many posts on a course about World Wars were being incorrectly flagged as negative, whereas they were simply mentioning various course topic words (e.g., war, deaths, holocaust), rather than expressing a negative opinion about the topic or course itself. We plan to investigate using course descriptions to further tune our sentiment analyses.

4 Conclusions

In this document we described the main components of OUSocial2; a web based tool for assessing and monitoring students' engagement and sentiment in public social media groups about their courses. The tool enables course leaders and the university to become aware of potential concerns and factors that could lead to students to quit their courses, which is a known problem with online learning and MOOCs.

References

1. M. Fernandez, H. Alani, and S. Brown. Ou social: reaching students in social media. In *Proc. 12th International Semantic Web Conference (ISWC 2013) - Demo*, Sydney, Australia, 2013.
2. M. Rowe and H. Alani. What makes communities tick? community health analysis using role compositions. In *4th IEEE Int. Conf. Social Computing (SocialCom)*, Amsterdam, 2012.
3. M. Rowe, M. Fernandez, S. Angeletou, and H. Alani. Community analysis through semantic rules and role composition derivation. *Journal of Web Semantics (JWS)*, 18(1):31–47, 2013.
4. A. Wolff, Z. Zdrahal, A. Nikolov, and M. Pantucek. Improving retention: predicting at-risk students by analysing clicking behaviour in a virtual learning environment. In *Third Conference on Learning Analytics and Knowledge (LAK 2013)*, Leuven, Belgium, 2013.

Using an Ontology Learning System for Trend Analysis and Detection

Gerhard Wohlgenannt and Stefan Belk and Matyas Karacsonyi and Matthias Schett

Vienna Univ. of Economics and Business, Welthandelsplatz 1, 1200 Wien, Austria
{gerhard.wohlgenannt, stefan.belk, matyas.karacsonyi}@wu.ac.at
<http://www.wu.ac.at>

Abstract. The aim of ontology learning is to generate domain models (semi-) automatically. We apply an ontology learning system to create domain ontologies from scratch in a monthly interval and use the resulting data to detect and analyze trends in the domain. In contrast to traditional trend analysis on the level of single terms, the application of semantic technologies allows for a more abstract and integrated view of the domain. A Web frontend displays the resulting ontologies, and a number of analyses are performed on the data collected. This frontend can be used to detect trends and evolution in a domain, and dissect them on an aggregated, as well as a fine-grained-level.

Keywords: trend detection, ontology evolution, semantic technologies, ontology learning

1 Introduction

Ontologies are a cornerstone technology of the Semantic Web. As the manual construction of ontologies is expensive, there have been a number of efforts to (semi-)automatic ontology learning (OL). The demo application builds upon an existing OL system, but extends the system to apply it as a Web intelligence, resp. a trend detection, tool.

As the system generates lightweight domain ontologies from scratch in regular intervals (ie. monthly), the starting point is always the same. This allows meaningful comparisons between ontologies, allowing to trace ontology evolution and general trends in the domain. The system captures an abundance of data about the ontologies in a relational database, from high-level to low-level (see below), which helps to analyze and visualize trends. The OL system generates ontologies from 32 heterogeneous evidence sources, which contain domain data from the respective period of time, so we can not only analyze the resulting ontologies but trace which sources support which ontological elements.

In summary, we use Semantic Web technologies as a Web intelligence tool by extending the system with visual and analytic components for trend detection. Trend detection is a major issue in a world that is changing rapidly. Timely detection of trends (and reaction to them) is important in many areas, eg. for success in business [2].

2 The Underlying Ontology Learning System

This section gives a brief introduction to the ontology learning (OL) system, as well as the sources of evidence used. We try to be as brief as possible, and include only information crucial to understand the trend detection application (for more details see the related work section and the referenced literature).

All trend detection analyses described in the following are based on a specific system for OL and ontology evolution. The system learns lightweight ontologies, more precisely taxonomies plus unlabeled non-taxonomic relations, from heterogeneous input sources. At the moment we use “climate change” as our test domain, and generate ontologies in monthly intervals. As the framework learns from scratch, it starts with a small seed ontology (two static concepts). For this seed ontology, we collect evidence from the evidence sources, and integrate the data (typically a few thousand terms including their relation to the seed concepts) into a spreading activation network. The spreading activation algorithm selects the 25 (current setting) most important new domain concept candidates. The only step which needs human assessment is a relevance check for the concept candidates done with crowdsourcing. A positioning step integrates the candidates into the existing seed ontology. This concludes the first “stage” of OL. We then use the extended ontology as new seed ontology, and start over. The system halts after three rounds of extension.

As already mentioned, the learning process relies on 32 heterogeneous evidence sources. Most of these sources are very dynamic and therefore well-fit for trend detection. The text-based sources include domain-specific corpora extracted from news media articles (segregated by country of origin), Web sites of NGOs and Fortune 1000 companies, domain-filtered postings from Facebook, Youtube, etc. We use keyword extraction and Hearst-style patterns to collect evidence, i.e. terms and relations. Furthermore, the system queries Social Web APIs (Twitter, Flickr) to get related and terms. We also use a few rather static sources, such as WordNet and DBpedia to help with taxonomy building.

3 Trend Detection and Analysis on Different Levels

Our demo system contains three main areas, namely (i) the ontologies, ie. the monthly snapshots of the domain model, (ii) high-level evolution, which include aggregated analyses on the characteristics of the evidence sources and ontologies, and (iii) low-level evolution, which trace the dynamics of concepts and single evidence on a fine-grained level. The demo portal can be found at <http://hugo.ai.wu.ac.at:5050>, a screencast presentation of the portal is available at <http://ai.wu.ac.at/~wohlg/iswc-demo.mp4>.

3.1 Ontologies

The *Ontologies* menu lists all ontologies computed per computation setting. The computation setting is simply a distinct system configuration. By clicking on an

ontology, the system displays detailed information. This includes representations in OWL/Turtle syntax and as graph of the resulting ontology, as well as of intermediary results. A user also finds performance data and the list of concepts by extension level. For a more detailed analysis, one can take a look at all evidence collected and used in the learning process. Multiple viewpoints (by concept, by evidence source, ...) allow investigating the underlying data.

In a nutshell, the *Ontologies* menu facilitates the analysis of trends in the domain both on the level of ontologies and the underlying evidence data.

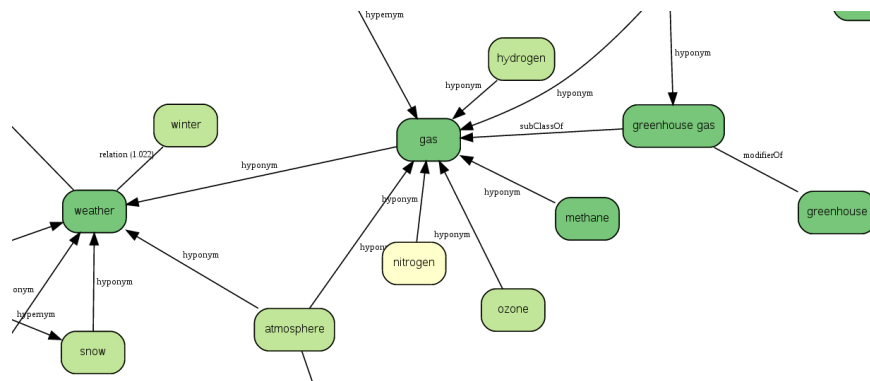


Fig. 1. Example snippet from an ontology (as graph) generated.

3.2 Low-Level Evolution

The *Concept History* shows which concepts have been added and removed from the ontology over time – for a specific system setting. For example, due to media coverage on hurricanes in October 2013 (see also Google trends), the concept hurricane was added to the ontology in November 2013 (in most settings). Entering “hurricane” as concept candidate in the *ECM* analysis presents the fine-grained development of evidence of the concept. Figure 2 shows which sources (US news media, UK news media, etc.) support the concept to what extend.

3.3 High-Level Evolution

The *High-Level Evolution* menu includes tools and visualizations to trace the evolution of evidence sources and the quality of the OL algorithms. For example, the source impact vector (SIV) graph shows the impact of the evidence sources on the system, which is computed according to the observed quality of suggestions from these sources. *Source evolution* displays the evolution of quality of concept candidates suggested by the source.

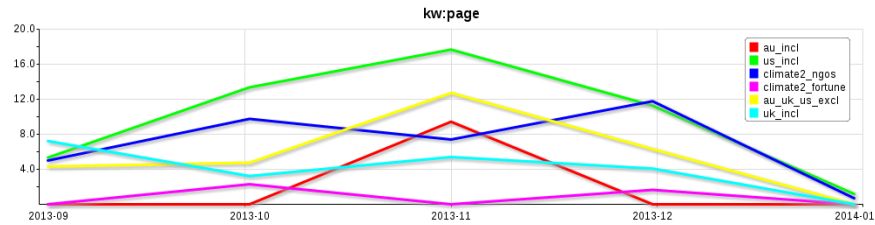


Fig. 2. (Keyword-generated) evidence for concept *hurricane* in various Web corpora (News Media, NGOs Websites, etc.)

4 Related Work

More information about the OL system used as foundation for the trend detection experiments and visualizations can be found in Weichselbraun et al. [3] and Wohlgenannt et al.[4]. A number of approaches have been proposed for trend detection from text data. For example, Bolelli et al. [1] first divide documents into time segments, then detected topics with a latent Dirichlet allocation model, and finally trace the evolution of the topics. In the realm of social media, TwitterMonitor [2] identifies trends on Twitter in real time.

5 Conclusions

The demo application uses Semantic Web (ontology learning) technologies to facilitate trend analysis and detection in a given domain. Users can trace change on different levels, (i) on the level of ontologies themselves, (ii) the aggregated level of quality of the system and impact of evidence sources, and (iii) the fine-grained level on concepts and single evidence. The fine-grained level is especially helpful to determine the reasons for trends in the sources of evidence. Future work will include the implementation of additional analyses and visualizations and the application of the tool in other domains, for example finance and politics.

References

1. Bolelli, L., Ertekin, S., Giles, C.L.: Topic and trend detection in text collections using latent dirichlet allocation. In: Proc. 31th European Conf. on IR Research. pp. 776–780. ECIR '09, Springer-Verlag, Berlin, Heidelberg (2009)
2. Mathioudakis, M., Koudas, N.: Twittermonitor: Trend detection over the twitter stream. In: Proc. of the 2010 ACM SIGMOD Int. Conference on Management of Data. pp. 1155–1158. SIGMOD '10, ACM, New York, NY, USA (2010)
3. Weichselbraun, A., Wohlgenannt, G., Scharl, A.: Refining non-taxonomic relation labels with external structured data to support ontology learning. *Data & Knowledge Engineering* 69(8), 763–778 (2010)
4. Wohlgenannt, G., Weichselbraun, A., Scharl, A., Sabou, M.: Dynamic integration of multiple evidence sources for ontology learning. *Journal of Information and Data Management (JIDM)* 3(3), 243–254 (2012)

A Prototype Service for Benchmarking Power Consumption of Mobile Semantic Applications

Evan W. Patton and Deborah L. McGuinness

Rensselaer Polytechnic Institute
110 8th Street, Troy NY 12180 USA
{pattoe, dlm}@cs.rpi.edu
<http://tw.rpi.edu/>

Abstract. We present a prototype web service that enables researchers to evaluate the performance per watt of semantic web tools. The web service provides access to a hardware platform for collecting power consumption data for a mobile device. Experiments are specified using RDF to define the conditions of the experiment, the operations that compose those conditions, and how they are combined into individual execution plans. Further, experimental descriptions and their provenance are published as linked data, allowing others to easily repeat experiments. We will demonstrate how we have used the system to date, how others can use it, and discuss its potential to revolutionize design and development of semantically enabled mobile applications.

Keywords: reasoning, mobile, power, performance, resource-constrained

1 Introduction

One challenge that semantic technologies face when deployed on mobile platforms like smartphones is the amount of energy available for the device to compute and communicate with other agents. For example, the Google Nexus One, one of the first Android smartphones, had a single core processor operating at 1 GHz and 512 MB of RAM. Samsung's latest offering, the Galaxy S5, has a quad core, 2.5 GHz processor and 2 GB of RAM, more than a 8-fold increase in processing power and 4-fold increase in memory in 5 years. However, the battery capacity of the two phones are 1400 mAh and 2800 mAh, resp., which indicates that battery technology is progressing more slowly than processing technology. Further, application complexity has also increased. Tools are needed to help developers understand how semantic tools consume power so as to identify when they can use local reasoning on mobile platforms or when off-device computation is more practical.

We introduced a broadly reusable methodology [3] motivated by these concerns to evaluate the performance of reasoners relative to the amount of energy consumed during operation. Ultimately, these metrics will provide developers deeper insight into power consumption and enable next-generation applications of semantic technologies for power constrained devices. We present a prototype

Table 1. A data sample for query 14 (Listing 1.1) from LUBM executed on the Samsung Galaxy S4. Times are in milliseconds, memory is in kilobytes, and power is in milliwatts.

Reasoner	Init.	Ont. Load	Data Load	Query Plan	Answer	Memory	Power
Jena	0.122	372.6	7076	2.594	233.2	35023	944
Pellet	0.152	355.7	8872	1.984	12350	59418	1024
HermiT	0.427	407.8	17442	0.092	21205	58720	995

ontology-driven web service for researchers to use our reference hardware setup to perform analysis of semantic web tools’ power consumption.

2 Web Service for Power-Performance Evaluation

Our power benchmarking methodology [3] bypasses the removable battery in a Samsung Galaxy S4 to collect power data during reasoning and query answering tasks using three reasoning engines. Because our methodology requires a hardware setup, we are developing and will demonstrate a web service to execute experiments using our existing infrastructure. The web service is based on the Semantic Automated Discovery and Integration (SADI) Framework¹ and accepts jobs described using RDF and the ontology we will discuss in Section 3. On completion, it provides a ZIP file containing runtime information, raw and processed power measurements, power and energy consumption statistics, and provenance capturing information about the process. Table 1 shows a sample data point for each of three different reasoners on the Lehigh University Benchmark [2], query 14 (shown in Listing 1.1).

Listing 1.1. Lehigh University Benchmark query 14

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX ub: <http://swat.cse.lehigh.edu/onto/univ-bench.owl#>
SELECT ?X WHERE {?X rdf:type ub:UndergraduateStudent}
```

3 Toward an Ontology for Experiment Descriptions

We will demonstrate our experimental ontology for declaratively describing the operational constraints of an experiment, which is then executed on the target device. The experiment description is published as linked data, along with metadata about the experiment output, and provenance modeled using the PROV-O ontology [1]. These metadata are published in a triple store to enable meta-analysis, recombination, and extension of power experiments.

¹ <http://sadiframework.org/content/>

Experiment. *Experiment* provides the root of an experiment description. Listing 1.2 shows an example experiment. Core to an experiment description are conditions, which are grouped together based on some common dimension. If an experiment defines more than one condition group, the engine performing the experiment can generate specific conditions through the use of a *condition-FillStrategy*. We are currently investigating two different strategies, *CrossJoin* and *Paired*, that evaluate the cross product and paired conditions across groups, respectively. To provide control over what data are returned, the author of the experiment can declare which variables are of interest.

Listing 1.2. An example description of an experiment

```
[ ] a exp:Experiment ;
    exp:name 'LUBM on Android' ;
    exp:version '1.0' ;
    dc:creator <http://www.evanpatton.com/evan.rdf#me> ;
    exp:trials 30 ;
    exp:conditions :ReasonerConditionGroup,
        OntologyConditionGroup ;
    exp:conditionFillStrategy exp:CrossJoin ;
    exp:dependentVariable exp:ExecutionInterval,
        exp:AveragePowerConsumption, exp:MaxPowerConsumption .
```

Conditions and Condition Groups. *Conditions* are the highest unit of test in our experiment ontology. They are composed of collections of operations that specify a sequence of actions to take on the device. Listing 1.3 shows an example of an ontology condition group that specifies two different ontology operations. Currently, we only support nominal values, but future versions of the ontology will also support ordinal, scalar, and ratio level inputs.

Listing 1.3. An example of a condition group with conditions

```
:OntologyConditionGroup a exp:ConditionGroup ;
    exp:name 'Ontology Condition' ;
    exp:varies exp:OntologyDatasetQueryOperation ;
    exp:nominalValues ( :SchemaOrgOperations :LUBMOperations )
```

Operations. *Operations* encapsulate actions to be performed on the experimental device. In Listing 1.4, we show an example of how an operation would define tests for the LUBM query set. The *measurePowerDuringDownload* property can be used to evaluate the performance of communication channels while retrieving the content required for performing the experiment. In addition to loading ontologies, datasets, and executing queries, our ontology supports modeling reasoners, parallel and sequential operations, and randomization of operations.²

² Due to space constraints, we cannot elaborate on the details of modeling each operation type. For more information and examples, please see <http://power.tw.rpi.edu>

Listing 1.4. An example of an combined operation on an ontology, dataset, and queries

```
:LUMBOperations a exp:OntologyDatasetQueryOperation ;
  exp:name 'LUBM' ;
  exp:measurePowerDuringDownload false ;
  exp:ontology lubm:univ-bench.owl ;
  exp:dataset lubm:lubm-100k.ttl ;
  exp:query lubm:query1.rq , lubm:query2.rq , ... .
```

4 Discussion and Conclusions

As semantic technologies become more prevalent, we need to ensure that tools are available to assist in their deployment on a variety of devices including mobile platforms, which are often power constrained. While our initial work provides direction for this effort, we recognize that more widespread adoption requires lower barriers to entry. We described a web service under active development to provide access to a reference implementation of the hardware described in [3] where we found that, while compute time accounts for most energy consumption, significant memory consumption may affect power consumption during reasoning. With this investigation, we are working to enable semantic web researchers and implementers to obtain insight into the power requirements for semantic technology stacks. We will demonstrate a variety of example experiments and discuss broader usage with attendees. In future work we intend to use expand this web service to provide a means of easily repeating experiments as well as further support for modeling the execution of experiments using We also intend to provide example code that present an analysis of more reasoners, e.g. by utilizing the work in [4].

Acknowledgements

Mr. Patton was funded by an NSF Graduate Research Fellowship. RPIs Tetherless World Constellation is supported in part by Fujitsu, Lockheed Martin, LGS, Microsoft Research, Qualcomm, in addition to sponsored research from DARPA, IARPA, NASA, NIST, NSF, and USGS.

References

1. Belhajjame, K., Cheney, J., Corsar, D., Garijo, D., Soiland-Reyes, S., Zednik, S., Zhao, J.: PROV-O: The PROV ontology. Tech. rep., W3C (2013), <http://www.w3.org/TR/prov-o/>
2. Guo, Y., Pan, Z., Heflin, J.: LUBM: A benchmarking for OWL knowledge base systems. *Web Semantics* 3(2), 158–182 (2005)
3. Patton, E.W., McGuinness, D.L.: A power consumption benchmark for reasoners on mobile devices. In: *Proceedings of the 13th International Semantic Web Conference: Replication, Benchmark, Data & Software Track* (2014)
4. Yus, R., Bobed, C., Esteban, G., Bobillo, F., Mena, E.: Android goes semantic: DL reasoners on smartphones. In: *OWL Reasoner Evaluation Workshop 2013* (2013)

SPARKLIS: a SPARQL Endpoint Explorer for Expressive Question Answering

Sébastien Ferré

IRISA, Université de Rennes 1
Campus de Beaulieu, 35042 Rennes cedex, France
Email: ferre@irisa.fr

Abstract. SPARKLIS is a Semantic Web tool that helps users explore SPARQL endpoints by guiding them in the interactive building of questions and answers, from simple ones to complex ones. It combines the fine-grained guidance of faceted search, most of the expressivity of SPARQL, and the readability of (controlled) natural languages. No endpoint-specific configuration is necessary, and no knowledge of SPARQL and the data schema is required from users. This demonstration paper is a companion to the research paper [2].

1 Motivation

A wealth of semantic data is accessible through SPARQL endpoints. DBpedia alone contains several billions of triples covering all sorts of topics (e.g., people, places, buildings, species, films, books). Although different endpoints may use different vocabularies and ontologies, they all share a common interface to access and retrieve semantic data: the SPARQL query language. In addition to being a widely-adopted W3C standard, the advantages of SPARQL are its *expressivity*, especially since version 1.1, and its *scalability* for large RDF stores thanks to highly optimized SPARQL engines (e.g., Virtuoso, Jena TDB). Its main drawback is that writing SPARQL queries is a tedious and error-prone task, and is largely inaccessible to most potential users of semantic data.

Our motivation in developing SPARKLIS¹, shared by many other developers of Semantic Web tools and applications, is to unleash access to semantic data by making it easier to define and send SPARQL queries to endpoints. The novelty of SPARKLIS is to combine in an integrated fashion different search paradigms: Faceted Search (FS), Query Builders (QB), and Natural Language Interfaces (NLI). That integration is the key to reconcile properties for which there is generally a trade-off in existing systems: user guidance, expressivity, readability of queries, scalability, and portability to different endpoints [2].

2 Principles

SPARKLIS re-uses and generalizes the interaction model of Faceted Search (FS) [8], where users are *guided step-by-step* in the selection of items. At each

¹ Online at <http://www.irisa.fr/LIS/ferre/sparklis/osparklis.html>

step, the system gives a set of suggestions to refine the current selection, and users only have to pick a suggestion according to their preferences. The suggestions are specific to the selection, and therefore support exploratory search [7] by providing overview and feedback during the search process.

To overcome *expressivity* limitations of FS and existing extensions for the Semantic Web (e.g., gFacet [4], VisiNav [3], SemFacet [1]), we have generalized it to Query-based Faceted Search (QFS), where the selection of items is replaced by a structured query. The latter is built step-by-step through the successive choices of the user. This makes SPARKLIS a kind of Query Builder (QB), like SemanticCrystal [5]. QBs have the advantage to allow for a high expressivity while assisting users about syntax, e.g. avoiding syntax errors, listing eligible constructs. However, the FS-based guidance of SPARKLIS is more fine-grained than in QBs. SPARKLIS avoids vocabulary errors by retrieving the URIs and literals right from the SPARQL endpoint. It needs not be configured for a particular dataset, and dynamically discovers the data schema. In fact, SPARKLIS only allows the building of queries that *do* return results, preventing users to fall on empty results. That is because system suggestions are computed for the individual results, not for their common class. In fact, SPARKLIS is as much about building answers as about building questions.

To overcome the lack of *readability* of SPARQL queries for most users, SPARKLIS queries and suggestions are verbalized in natural language so that SPARQL queries never need to be shown to users. This makes SPARKLIS a kind of Natural Language Interface (NLI), like PowerAqua [6]. The important difference is that questions are built through successive user choices in SPARKLIS instead of being freely input in NLIs. SPARKLIS interaction makes question formulation more constrained, slower, and less spontaneous, but it provides guidance and safeness with intermediate answers and suggestions at each step. Moreover, it avoids the hard problem of NL understanding: i.e., ambiguities, out-of-scope questions. A few NLI systems, like Ginseng [5], are based on a controlled NL and auto-completion to suggest the next words in a question. However, their suggestions are not fine-grained like with FS, and less flexible because they only apply to the end of the question. In SPARKLIS, questions form complete sentences at any step of the search; and suggestions are not words but meaningful phrases (e.g., **that has a director**), and can be inserted at any position in the current question.

3 User Interface and Interaction

Figure 1 is a SPARKLIS screenshot taken during an exploration of book writers in DBpedia. From top to bottom, the user interface contains (1) navigation buttons and the endpoint URL, (2) the current question and the current *focus* as a subphrase (highlighted in green), (3) three lists of suggestions for insertion at the focus, and (4) the table of answers. The shown question and answer have been built in 10 steps (8 insertions and 2 focus moves): **a Writer/that has a birthDate/after 1800/focus on a Writer/that is the author of something/a Book/a number of/the**

The screenshot shows the SPARKLIS interface. At the top, there are navigation arrows and a SPARQL endpoint field with the URL `http://dbpedia.org/sparql`. Below this is a 'Your query' section with a 'permlink' button. The query text is: 'Give me a **Writer** whose **birthDate** is after 1800 and that is the **author** of the **highest-to-lowest** number of **Book** and that has a **nationality** **X**'. Below the query is a section for 'Sparklis suggestions to refine your query' containing three panels: '40 entities' (listing nationalities like American, British, etc.), '202 concepts' (listing properties like `sameAs`, `type`, etc.), and '10 modifiers' (listing logical connectives like `and`, `or`, etc.). At the bottom, the 'Results of your query' section shows a table with 4 results, with the 'nationality' column highlighted in green.

	the Writer	the Writer's birthDate	the Writer's nationality	the number of Book
1	L. Sprague de Camp	1907-11-26+02:00 (date)	American (en)	128 (integer)
2	Agatha Christie	1890-09-14+02:00 (date)	British (en)	103 (integer)
3	Isaac Asimov	1920-01-01+02:00 (date)	American (en)	75 (integer)
4	Philip K. Dick	1928-12-15+02:00 (date)	American (en)	74 (integer)

Fig. 1. SPARKLIS screenshot: a list of writers with their birth date (after 1800), nationality, and (decreasing) number of written books. Current focus is on writer’s nationality.

highest-to-lowest/focus on a Writer/that has a nationality. Note that different insertion orderings are possible for a same question. Navigation buttons allow to move backward/forward in the construction history. A permalink to the current navigation state (endpoint+question) can be generated at any time. To switch to another SPARQL endpoint, it is enough to input its URL in the entry field. The query focus is moved simply by clicking on different parts of the question, or on different table column headers. Every suggestion in the three lists, as well as every table cell, can be inserted or applied to the current focus by clicking it. The first suggestion list contains entities (individuals and literals). The second list contains concepts (classes and properties). The third list contains logical connectives, sorting modifiers, and aggregation operators. Each suggestion list is equipped with an immediate-feedback filtering mechanism to quickly locate suggestions in long lists. With the first list, filters can be inserted into the query with different filter operators listed in a drop-down menu (e.g., **matches**, **higher** or **equal than**, **before**). Questions and suggestions use indentation to disambiguate different possible groupings and improve readability, and syntax coloring to distinguish between the different kinds of words.

4 Performances and Limitations

Portability. SPARKLIS conforms to the SPARQL standard, and requires no pre-processing or configuration to explore an endpoint. It entirely relies on the end-

point to discover data and its schema. The main limitation is that URIs are displayed through their local names, which is not always readable.

Expressivity. SPARKLIS covers many features of SPARQL: basic graph patterns (including cycles), basic filters, UNION, OPTIONAL, NOT EXISTS, SELECT, ORDER BY, multiple aggregations with GROUP BY. Almost all queries of the QALD² challenge can be answered. Uncovered features are expressions, named graphs, nested queries, queries returning RDF graphs, and updates.

Scalability. SPARKLIS is responsive on the largest well-known endpoint: DBpedia. Among the 100 QALD-3 questions, half can be answered in less than 30 seconds (wall-clock time including user interaction and system computations).

5 Demonstration

The demonstration has shown to participants how QALD questions over DBpedia can be answered in a step-by-step process. Those questions cover various retrieval tasks: basic facts (*Give me the homepage of Forbes*), entity lists (*Which rivers flow into a German lake?*), counts (*How many languages are spoken in Colombia?*), optimums (*Which of Tim Burton's films had the highest budget?*). More complex analytical question answering has also been demonstrated (*Give me the total runtime, from highest to lowest, of films per director and per country*). Participants were also given the opportunity to explore any SPARQL endpoint of their choice.

References

1. Arenas, M., Grau, B., Kharlamov, E., Š. Marciuška, Zheleznyakov, D., Jimenez-Ruiz, E.: SemFacet: Semantic faceted search over YAGO. In: World Wide Web Conf. Companion. pp. 123–126. WWW Steering Committee (2014)
2. Ferré, S.: Expressive and scalable query-based faceted search over SPARQL endpoints. In: Mika, P., Tudorache, T. (eds.) Int. Semantic Web Conf. Springer (2014)
3. Harth, A.: VisiNav: A system for visual search and navigation on web data. J. Web Semantics 8(4), 348–354 (2010)
4. Heim, P., Ertl, T., Ziegler, J.: Facet graphs: Complex semantic querying made easy. In: et al., L.A. (ed.) Extended Semantic Web Conference. pp. 288–302. LNCS 6088, Springer (2010)
5. Kaufmann, E., Bernstein, A.: Evaluating the usability of natural language query languages and interfaces to semantic web knowledge bases. J. Web Semantics 8(4), 377–393 (2010)
6. Lopez, V., Fernández, M., Motta, E., Stieler, N.: PowerAqua: Supporting users in querying and exploring the semantic web. Semantic Web 3(3), 249–265 (2012)
7. Marchionini, G.: Exploratory search: from finding to understanding. Communications of the ACM 49(4), 41–46 (2006)
8. Sacco, G.M., Tzitzikas, Y. (eds.): Dynamic taxonomies and faceted search. The information retrieval series, Springer (2009)

² <http://greententacle.techfak.uni-bielefeld.de/~cunger/qald/>

Reconciling Information in DBpedia through a Question Answering System

Elena Cabrio^{1,2}, Alessio Palmero Aprosio³, and Serena Villata¹

¹ INRIA Sophia Antipolis, France - `firstname.lastname@inria.fr`

² EURECOM, France

³ Fondazione Bruno Kessler, Trento, Italy - `aprosio@fbk.eu`

Abstract. Results obtained querying language-specific DBpedia chapters SPARQL endpoints for the same query can be related by several heterogeneous relations, or contain an inconsistent set of information about the same topic. To overcome this issue in question answering systems over language-specific DBpedia chapters, we propose the RADAR framework for information reconciliation. Starting from a categorization of the possible relations among the resulting instances, such framework: *(i)* classifies such relations, *(ii)* reconciles the obtained information using argumentation theory, *(iii)* ranks the alternative results depending on the confidence of the source in case of inconsistencies, and *(iv)* explains the reasons underlying the proposed ranking.

1 Introduction

In the Web of Data, it is possible to retrieve heterogeneous information items concerning a single real-world object coming from different data sources, e.g., the results of a single SPARQL query on different endpoints. These results may conflict with each other, or they may be linked by some other relation like a specification. The automated detection of the kind of relationship holding between different instances about a single object with the goal of reconciling them is an open problem for consuming in the Web of Data. In particular, this problem arises while querying the language-specific chapters of DBpedia, that may contain different information with respect to the English version. This issue becomes therefore particularly relevant in Question Answering (QA) systems exploiting DBpedia language-specific chapters as referential data set, since the user expects a unique (and possibly correct) answer to her factual natural language question.

In this demo, we propose the RADAR (ReconciliAtion of Dbpedia through ARgumentation) framework that: *i)* adopts a classification method to return the relation holding between two information items; *ii)* applies *abstract argumentation theory* [4] for reasoning about conflicting information and assessing the acceptability degree of the information items, depending on the kind of relation linking them; and *iii)* returns the graph of the results set, together with the acceptability degree of each information item, to motivate the resulting information ranking. We have integrated RADAR into the QA system QAKiS [1], that queries language-specific DBpedia chapters using a natural language interface.

2 RADAR: a Framework for Information Reconciliation

The RADAR framework (Fig. 1) takes as input a collection of results from the same SPARQL query raised against the language-specific DBpedia chapters SPARQL endpoints, and retrieves: (i) the sources proposing each particular element of the results set, and (ii) the elements of the results set themselves. The first module of RADAR (*Source confidence assignment score*, Fig. 1) takes each information source, and following two different heuristics, i.e., *Wikipedia page length* (the chapter of the longest language-specific Wikipedia page describing the queried entity is rewarded w.r.t. the others) and *entity geo-localization* (the chapter of the language spoken in the places linked to the page of the entity is rewarded with respect to the others), assigns a confidence degree to the source. Such metrics are summed, and normalized ($0 \leq \text{score} \leq 1$), where 0 is the less reliable chapter for a certain entity and 1 is the most reliable one. Such confidence degree will affect the reconciliation if inconsistencies arise: information proposed by the more reliable source will obtain a higher acceptability degree.

The second module (*Relation classification module*,

Fig. 1) starts from the results set, and it matches every element with all the other returned elements, detecting the kind of relation holding between this pair of elements, following the categorization of [3]. Such categories correspond to the linguistic phenomena (mainly discourse and lexical semantics relations) holding among heterogeneous

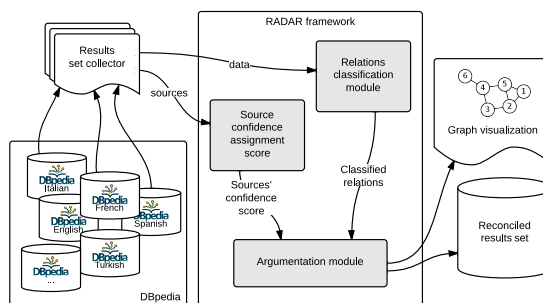


Fig. 1: RADAR framework architecture.

values obtained querying two DBpedia language-specific chapters, given a certain subject and a certain ontological property. RADAR clusters the relations of *identity*, *disambiguated entity* and *coreference* into a unique category, called *surface variants* of the entity, and automatically detects such relation among two entities applying one of the following strategies: cross-lingual links (using WikiData), text identity (i.e., string matching), Wikipedia redirection and disambiguation pages. Moreover, RADAR integrates into a unique category *geo-specification* and *renaming*, and classifies a relation of this category when in GeoNames one entity results as contained in the other one. We also consider the alternative names gazette included in GeoNames, and geographical information extracted from English Wikipedia infoboxes, such as **Infobox former country**. Finally, RADAR clusters *meronymy*, *hyponymy*, *metonymy* and *identity:stage name* into a unique category, called *inclusion*, and detects it exploiting a set of features extracted from heterogeneous resources: MusicBrainz, NCF Thesaurus, DBpe-

dia, WikiData and Wikipedia hierarchical information. Concerning inconsistent data in DBpedia language-specific chapters, RADAR labels a relation between entities/objects as negative, if every attempt to find one of the positive relations described above fails. The output consists in a graph composed by the elements of the results set connected with each other by the identified relations. Both the sources associated with a confidence score and the results set under the form of a graph are then provided to the third module of RADAR, the *Argumentation module* (Fig. 1). Its aim is to reconcile the results set: it considers all positive relations as a *support* relation and all negative relations as an *attack* relation, building a bipolar argumentation graph where each element of the results set is seen as an argument. Finally, adopting a bipolar fuzzy labeling algorithm [2] relying on the source’s confidence to decide the acceptability of the information, the module returns the acceptability degree of each argument, i.e., element of the results set. RADAR provides as output: *i*) the acceptable elements (a threshold is adopted), and *ii*) the graph of the results set, i.e., the explanation about the choice of the acceptable elements returned.

Integrating RADAR into QAKiS. QAKiS addresses the task of QA over structured knowledge-bases (e.g., DBpedia) [1], where the relevant information is expressed also in unstructured forms (e.g., Wikipedia pages). It implements a relation-based match for question interpretation, to convert the user question into a query language (e.g., SPARQL), making use of relational patterns (automatically extracted from Wikipedia and collected in the WikiFramework repository) that capture different ways to express a certain relation in a given language. In QAKiS, the SPARQL query created after the question interpretation phase is sent to a set of language-specific DBpedia chapters SPARQL endpoints for answer retrieval. The set of retrieved answers from each endpoint is then sent to RADAR for answers reconciliation¹. The user can select the DBpedia chapter she wants to query besides English (that must be selected as it is needed for Named Entity (NE) recognition), i.e., French or German. After writing a question or selecting it among the proposed examples, the user has to click on the tab *RADAR* where a graph with the answers provided by the different endpoints and the relations among them is shown. Each node has an associated confidence score, resulting from the fuzzy labeling algorithm. Moreover, each node is related to the others by a relation of support or attack, and a further specification of such relations according to the identified categories [3] is provided to the user as justification of the performed reconciliation and ranking.

To evaluate RADAR integration into QAKiS, we extract from QALD-2 data set² the questions currently addressed by QAKiS (i.e., questions containing a NE related to the answer through one single ontological property), corresponding to 58 questions (26 in the training, 32 in the test set). The discarded questions require either some forms of reasoning on data, aggregation from data sets other than DBpedia, involve n-relations, or are boolean questions. We submit

¹ A demo of RADAR integrated into QAKiS can be tested at <http://qakis.org>.

² <http://bit.ly/QALD2014>

such questions to QAKiS on the English, German and French DBpedia chapters. Since QALD-2 questions were created to query the English chapter only, it turned out that only in 25/58 cases at least two endpoints provide an answer (in all the other cases the answer is provided by the English chapter only, not useful for our purposes). For instance, given the question *Who developed Skype?* the English DBpedia provides *Skype Limited* as the answer, while the French one returns *Microsoft*. We evaluate the ability of RADAR to correctly classify the relations among the answers provided to the same query by the different language-specific endpoints, w.r.t. a manually annotated goldstandard (built according to [3]’s guidelines), carrying out two sets of experiments: *i*) we start from the answers provided by the different DBpedia endpoints to the 25 QALD questions, and we run RADAR on it; *ii*) we add QAKiS in the loop, meaning that the data we use as input for the argumentation module are directly provided by the system. We obtain the following results: RADAR achieves a precision/recall/f-measure of 1 in the classification of *surface form* and *inclusion* relations (overall positive: p/r/f=1); QAKiS+RADAR obtains p=1, r=0.60 and f=0.75 on *surface form*, p/r/f of 1 on *inclusion* (overall positive: p/r/f= 1/0.63/0.77). Since QALD-2 data was created to query the English chapter only, this small data set does not capture the variability of possibly inconsistent answers among DBpedia language-specific chapters. Only two categories of relations are present in this data, i.e., *surface forms*, and *inclusion*, and for this reason RADAR has outstanding performances when applied on the correct mapping between NL questions and SPARQL queries. When QAKiS is added into the loop, its mistakes in translating the NL question into the correct SPARQL query are propagated.

3 Future Perspectives

This demo improves the results of [2] as the categorization is more specific thus producing a more insightful explanation graph, and more performing techniques are applied to extract the relations. As future work, we will address a user evaluation to check whether QAKiS answer explanation suits data consumers’ needs, and we will explore the possibility to leave the data consumer herself to assign the confidence degree to the sources depending on searched information.

References

1. Cabrio, E., Cojan, J., Gandon, F.: Mind the cultural gap: bridging language specific dbpedia chapters for question answering. In: Cimiano, P., Buitelaar, P. (eds.) *Towards the Multilingual Semantic Web*. Springer Verlag (2014)
2. Cabrio, E., Cojan, J., Villata, S., Gandon, F.: Argumentation-based inconsistencies detection for question-answering over dbpedia. In: *NLP-DBPEDIA@ISWC* (2013)
3. Cabrio, E., Villata, S., Gandon, F.: Classifying inconsistencies in dbpedia language specific chapters. In: *LREC-2014* (2014)
4. Dung, P.: On the acceptability of arguments and its fundamental role in non-monotonic reasoning, logic programming and n-person games. *Artif. Intell.* 77(2), 321–358 (1995)

Open Mashup Platform – A Smart Data Exploration Environment

Tuan-Dat Trinh, Ba-Lam Do, Peter Wetz, Amin Anjomshoaa,
Elmar Kiesling, and A Min Tjoa

Vienna University of Technology, Vienna, Austria
{tuan.trinh, peter.wetz, ba.do, amin.anjomshoaa,
elmar.kiesling, a.tjoa}@tuwien.ac.at

Abstract. The number of applications designed around Linked Open Data (LOD) has expanded rapidly in recent years. However, these applications typically do not make use of the vast amounts of LOD datasets, but only provide access to predefined, domain-specific subsets. Exceptions that do allow for more flexible exploration of LOD are not targeted at end users, which excludes users who have limited experience with Semantic Web technologies from realizing the potential of the so-called LOD cloud. This paper introduces a Mashup Platform that models, manages, reuses, and interconnects LOD web applications, thereby encouraging initiative and creativity of potential users. Figuratively, our approach allows developers to implement building blocks whereas the platform provides the cement so that end users can build houses by themselves.

1 Introduction

More than ten years after the concept of LOD has been introduced, can end users really benefit from it? In spite of considerable effort made by researchers, the answer, unfortunately, is “just a little”. This issue is becoming more pressing as the number of LOD dataset increases; as of 2014¹, there are already more than 60 billion triples from 928 datasets.

The limited adoption of LOD by end users may be explained by the following observations: (i) There are currently no platforms that allow end users to manage and reuse LOD applications. An “*LOD App Store*” – by analogy with digital distribution platforms for mobile apps such as *Google Play Store* or *App Store* – would be an interesting concept to foster diffusion among end users. (ii) Most LOD providers publish their datasets without paying regard to how their data may be used effectively or how it may be combined with data provided by others. After all, specific use cases are typically the most effective way to illustrate that the data is useful for end users. To save their own times, LOD providers need tools that support them in implementing such applications as efficiently as possible. (iii) Most importantly, end users currently play a passive role, waiting for developers to deliver LOD applications rather than leveraging LOD according

¹ <http://stats.lod2.eu/>

to their individual needs. To make good use of LOD, users currently need to equip themselves with knowledge about Semantic Web technologies as well as the SPARQL query language. Since this cannot generally be expected, the key question we address here is: “Is there any way to utilize the users’ capabilities and creativity and allow them to explore LOD themselves without the need to acquire specialized knowledge?”

To address these issues, we propose an *Open Mashup Platform*. The key idea of this platform is to compose LOD applications from linkable *Web widgets* provided by data publishers and developers. These widgets are divided into three categories, i.e., *data*, *process*, and *visualization* widgets which perform the data retrieval, data processing/data integration, and data presentation tasks, respectively. The internal mechanics of these complicated tasks are not visible to end users because they are encapsulated inside the widgets. Widgets have inputs and outputs and can be easily linked to each other by end users. Being web applications, they can run on various platforms and be shared and reused. When a LOD provider publishes a new dataset, they can develop new widgets and add them to the platform so that users can dynamically, actively and creatively combine these widgets to compose LOD applications across multiple LOD datasets. This paper presents the most basic functionalities of the Platform – a smart data exploration environment for end users available at <http://linkedwidgets.org>.

2 Prototype System

2.1 Graph-based model and the Annotator tool

To communicate and transmit data between widgets, each of them implements its own well-defined model as well as interfaces to provide the features required by the Platform. A widget is similar to a service in that it has multiple inputs and a single output. To model them, however, instead of capturing the functional semantics and focusing on input and output parameters like SAWSDL [6], OWL-S [2], WSMO [1] etc., we use a graph-based model similar to [5]. This approach has a number of advantages and is a prerequisite for the semantic search and auto-composition algorithms described in this paper. For example, from the model for a *Film Merger* widget that requires an *Actor* and a *Director* as its two inputs and returns a list of *Films* starring the actor and being directed by the director (cf. Fig. 1a), the relation between input and output instances is immediately apparent.

To allow developers to create and annotate widgets correctly and efficiently, we provide a *Widget Annotator* tool. Developers simply drag, drop and then configure three components, i.e., *WidgetModel*, *Object*, and *Relation* to visually define their widget models. After that, the OWL description file for the model as well as the corresponding HTML widget file are generated automatically. The latter includes the injected Java Script code snippet served for the widget communication protocol and a sample *JSON-LD* input/output of the widget according to the defined model. Based on that, developers can implement the widget’s processing function which receives input from widgets and returns output to others. De-

velopers can also rewrite/improve widgets using server-side scripting languages. Finally, as soon as they have deployed their widgets, developers can submit their work to the platform where it is listed and can be reused with other available widgets; in particular, the widgets annotations are published into the LOD of widgets which can be accessed from the graph <http://linkedwidgets.org> of the <http://ogd.ifs.tuwien.ac.at/sparql> SPARQL endpoint.

2.2 Semantic Widget Search

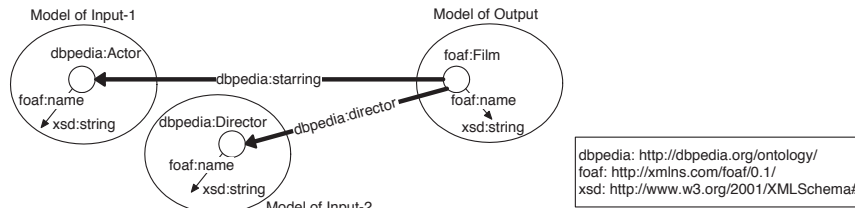
In line with the growth of the LOD cloud, the number of available widgets can be expected to grow rapidly. In this case, to ensure that users can find widgets on the platform, a *semantic search* will be provided in addition to conventional search methods by keywords, category, tags, etc. Because the widgets' RDF metadata is openly available via the SPARQL endpoint, other third parties, if necessary, can also develop their own widget-search tool. Our search tool is similar to the annotator tool, but it is much simpler and directed at end users. By defining the constraints for input/output, they, for example, can find widgets which return *Films* with particular properties for each *Film*, or even find widgets which consist of relationship between *Films* and *Actors*.

2.3 Mashup Panel

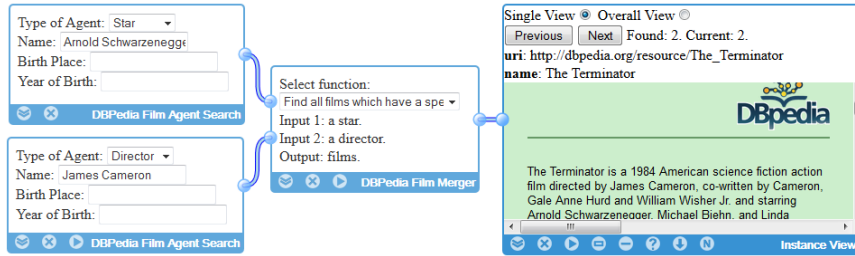
The *Mashup Panel* is the most crucial part of the platform; it allows users to compose, publish and share their applications, thereby enabling them to dynamically and actively explore LOD datasets without special skills or knowledge.

Widgets are grouped into *Widget Collections* to offer a group of scenarios after being combined to each other. Users can create their own collections or choose to work with existing collections from other users. The list of widgets that belong to the selected collection is placed at the left-hand side of the *Mashup Panel*. Users simply drag and drop widgets into the mashup area at the right-hand side. For each chosen widget, available operations are *resize*, *run*, *view/cache output data*, *get detailed information* about the widget based on its URI. In the next step, users can wire the input of a widget to the output of another one and thus build up a data-processing flow. The connected widgets, under the coordination of the platform, will communicate and transmit data to each other, from the very first data widgets to the visualization widgets. Finally, if the whole mashup is saved, parameters set in HTML form inputs from each widget will be automatically detected and stored so that users can publish the final result displayed inside the visualization widget onto their websites. Furthermore, the combined application are semantically annotated and can be shared between users via their URLs or URIs.

We implemented two algorithms to help users acquaint themselves with their new widgets: *auto-matching* and *auto-composition*. The *auto-matching* algorithm enables users to find – given input/output terminal *A* – all terminal *B* from all of other widgets such that connection between *A* and *B* is valid. The *auto-composition* algorithm is a more advanced approach in that it can automatically



(a) Model of the Film Merger Widget



(b) A complete sample LOD Application

Fig. 1: Sample widget model and widget combination

compose a complete application from a widget, or a complete branch that consumes/provides data for a specific output/input terminal. “Complete” in this context means that all terminals must be wired. This as well as the semantic search feature distinguish our platform from similar contributions, e.g., [4] or [3]. A sample application that collects all movies played by *Arnold Schwarzenegger* and directed by *James Cameron* is shown in Fig. 1b. Many other use cases can be found on the platform at <http://linkedwidgets.org>.

References

- de Bruijn, J., et al.: Web Service Modeling Ontology (WSMO) (2005), <http://www.w3.org/Submission/WSMO/>
- David, M., et al.: OWL-S : Semantic Markup for Web Services (2004), <http://www.w3.org/Submission/OWL-S/>
- Imran, M., et al.: ResEval mash: a mashup tool for advanced research evaluation. In: 21st international conference companion on World Wide Web. pp. 361–364 (2012)
- Le-Phuoc, D., et al.: Rapid prototyping of semantic mash-ups through semantic web pipes. In: 8th international conference on World Wide Web. p. 581. ACM Press, New York, New York, USA (2009)
- Taheriyani, M., Knoblock, C.: Rapidly integrating services into the linked data cloud. In: 11th International Semantic Web Conference. pp. 559–574 (2012)
- Wc, C.B., Ibm, J.F.: SAWSDL : Semantic Annotations for WSDL and XML Schema. IEEE Internet Computing 11(6), 60–67 (2007)

CIMBA - Client-Integrated MicroBlogging Architecture

Andrei Vlad Samba¹, Sandro Hawke¹, Tim Berners-Lee¹, Lalana Kagal¹, and Ashraf Abounaga²

¹ Decentralized Information Group,
MIT CSAIL

² Qatar Computing Research Institute
asambra@mit.edu, sandro@w3.org, timbl@w3.org, lkagal@csail.mit.edu,
aabounaga@qf.org.qa

Abstract. Personal data ownership and interoperability for decentralized social Web applications are currently two debated topics, especially when taking into consideration the aspects of privacy and access control. To increase data ownership, users should have the freedom to choose where their data resides and who is allowed access to it by decoupling data storage from the application that consumes it. Through CIMBA, we propose a decentralized architecture based on Web standards, which puts users back in control of their own data.

Keywords: decentralization, Linked Data, social Web, privacy, Web apps

1 Introduction

Recently, we have witnessed a dramatic increase in the number of social Web applications. These applications come in different forms and offer different services such as social networks, content management systems (CMS), bug trackers, blogging tools, or collaboration services in general.

A common practice, specific to most Web services, is to centralize user resources thus becoming so-called *data silos*. Often when adhering to online services people usually end up creating dedicated local accounts, which ties and limits users to particular services and/or resources. A solution to *data silos* can be achieved through decentralization, where users are free to host their data wherever they want, and then use *several* Web apps to consume and manage the data. In the following section we will discuss how our decentralized architecture plays an important role in achieving true data ownership for users.

2 Architecture

Today, more and more software is built around an application-specific back-end database. This makes switching applications problematic, since data are

structured according to each specific application and it only has meaning within the context of those applications. Moreover, this practice also forces a tight coupling between backends and applications consuming the data (cf. Fig.1 a).

The proposed architecture uses the Resource Description Framework (RDF) [1] to achieve greater interoperability between servers and applications as well as to ensure the data structure remains the same, regardless of the server on which the data are stored. In our case, CIMBA is a simple microblogging client that is completely decoupled from the backend, which in turn is a generic storage server (cf. Fig.1 b).

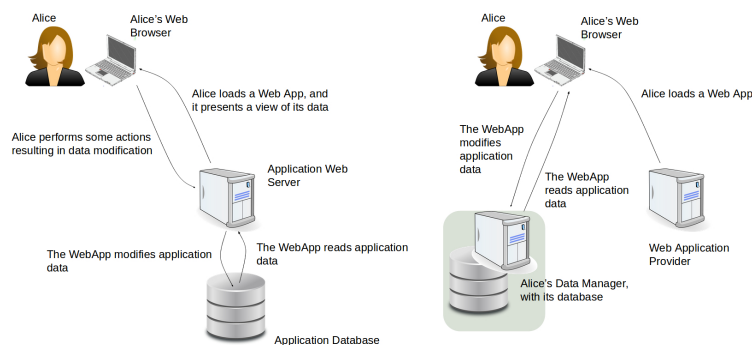


Fig. 1. a) Current architecture; b) Proposed decentralized architecture

By fully decoupling the server from the Web app, developers will be able to produce large scale Web apps without having to also manage the backend servers, making it very simple to switch from one backend to another, as well as from one Web app to another one without losing any data (cf. Fig.2 a).

Another advantage is that users are no longer locked into a *silo* because of their social connections (cf. Fig.2 b). Web apps reuse the user's social graph, which is also located on the data manager. The data manager is a generic Linked Data personal data server, which implements the Linked Data Platform specification [2] (currently on REC track at W3C³), as well as the Web Access Control [3] ontology (to enforce privacy policies).

Our architecture uses WebID [4] as the main mechanism to identify people at the Web scale, together with WebID-TLS [4] (to authenticate requests to restricted resources), a fully decentralized authentication scheme based on WebID.

³ <http://www.w3.org>

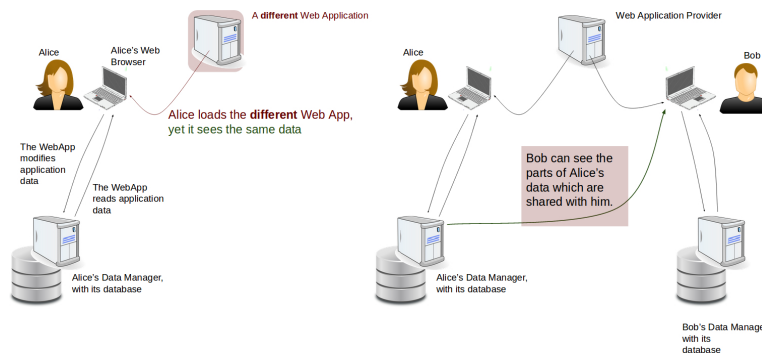


Fig. 2. a) users can easily switch software; b) social connections stay with the user

3 CIMBA

The name CIMBA stands for Client-Integrated MicroBlogging Application. It provides users with the power of having their own blog combined with the ease of using Twitter. CIMBA was written in Javascript, using the AngularJS⁴ framework. The source code is publicly available on GitHub⁵, and a running online demo can be accessed by visiting <http://cimba.co>.

Compared to Twitter, CIMBA users are not stuck with a single feed or timeline, but instead they can create multiple *Channels* and use them as categories for their posts (i.e. main, work, family, etc.). Access control can be set per channel as well as per post, though policies for posts will override those set per channel (i.e. having a private post in a public channel).

Figure 3 displays an overview of the architecture behind CIMBA. Users Alice and Bob each have their own personal data managers, which hold the posts data, configuration files as well as their personal WebID profiles.

Accessing CIMBA simply means loading all the necessary HTML, Javascript and CSS files from the application server into the user's browser. From that moment on, the application which now runs in the browser will communicate directly with the user's personal data manager. The location of the personal data manager is found by "faking" a WebID-TLS authentication process, with the purpose of finding the user's WebID and implicitly, the WebID profile. There is no actual need to authenticate the user to CIMBA, since all requests for data are authenticated by the user's personal data store.

Once the WebID profile is found, the app follows a series of links to discover useful information about the user, such as a generic Linked Data server from where the app can store and retrieve resources. CIMBA stores all the application data on the user's personal data manager, in a workspace dedicated to microblogging. Microblogging data are stored using the SIOC ontology.

⁴ <https://angularjs.org/>

⁵ <https://github.com/linkedata/cimba>

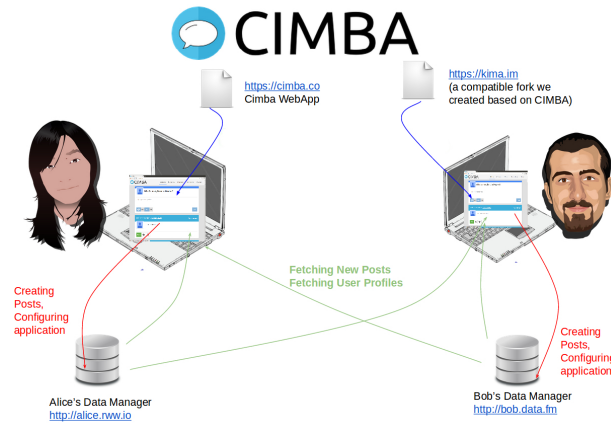


Fig. 3. Overview of the architecture for CIMBA

To read what other people write, users can subscribe to their channels. The list of subscriptions is also expressed using the SIOC vocabulary and it is stored on the user's server.

4 Conclusions and future work

Our proposed decentralized architecture offers significant benefits compared to current Web apps, both in terms of data ownership, privacy, as well as interoperability. Being fully decoupled from the backend, Web apps can be easily forked and improved by anyone with access to the source code, thus spurring innovation and creativity. At its current stage, CIMBA suffers from scalability issues, though we are very close to overcoming these issues.

References

1. Graham K., Carroll J., McBride B.: Resource description framework (RDF): Concepts and abstract syntax. In: W3C recommendation. (2004)
2. Speicher S., Arwe J., Malhotra A.: Linked Data Platform 1.0. <http://www.w3.org/TR/1dp/> (2014)
3. Hollenback J., Presbrey J., Berners-Lee T.: Using RDF metadata to enable access control on the social semantic web. In: Proceedings of the Workshop on Collaborative Construction, Management and Linking of Structured Knowledge (CK2009), vol. 514. (2009)
4. Sambra A., Henry S., Berners-Lee T.: WebID Specifications. <http://www.w3.org/2005/Incubator/webid/spec/> (2014)
5. Breslin J.G., Harth A., Bojars U., Decker, S.: Towards semantically-interlinked online communities. In: The Semantic Web: Research and Applications, pages 500-514. (2005)

The Organiser - A Semantic Desktop Agent based on NEPOMUK

Sebastian Faubel and Moritz Eberl

Semiodesk GbR, D-86159 Augsburg, Germany
{sebastian, moritz}@semiodesk.com

Abstract. In this paper we introduce our NEPOMUK-based Semantic Desktop for the Windows platform. It uniquely features an integrative user interface concept which allows a user to focus on personal information management while relying on the Property Projection agent for semi-automated file management.

Keywords: Semantic Desktop, Personal Information Management

1 Introduction

In recent years, mobile cloud computing [5] has created a paradigm shift in the use of electronic devices. It is now common for people to consume and produce content using multiple devices, online platforms and communication channels. However, productive and collaborative work is becoming increasingly fragmented across different mobile platforms, social networks and collaboration platforms [1].

Different devices and applications often come with their separate methods of organizing and storing information. Thus, considerable effort has to be made to represent a single piece of information in multiple systems. In our case, the ISWC 2014 conference is being represented in five different entities: a calendar event, a shared folder in the hierarchical file system, a notes list, a bookmarks folder and a task list.

There is a need for active support by computers in the creation and filtering process of personal and group information. It has to provide a consolidated view on data and blend the boundaries between workstations, mobile devices and web services. Semantic Web technologies, specifically the Semantic Desktop [6], offer a suitable platform for this purpose.

2 Our Solution

To solve this problem we have created the *Organiser*¹, a Semantic Desktop agent based on the NEPOMUK ontologies [4]. It integrates personal information such as contacts, events and notes from cloud services with the local file system and thus, provides a consolidated view.

¹ Demo video: <http://www.semiodesk.com/media/2014/0714-organiser-intro>

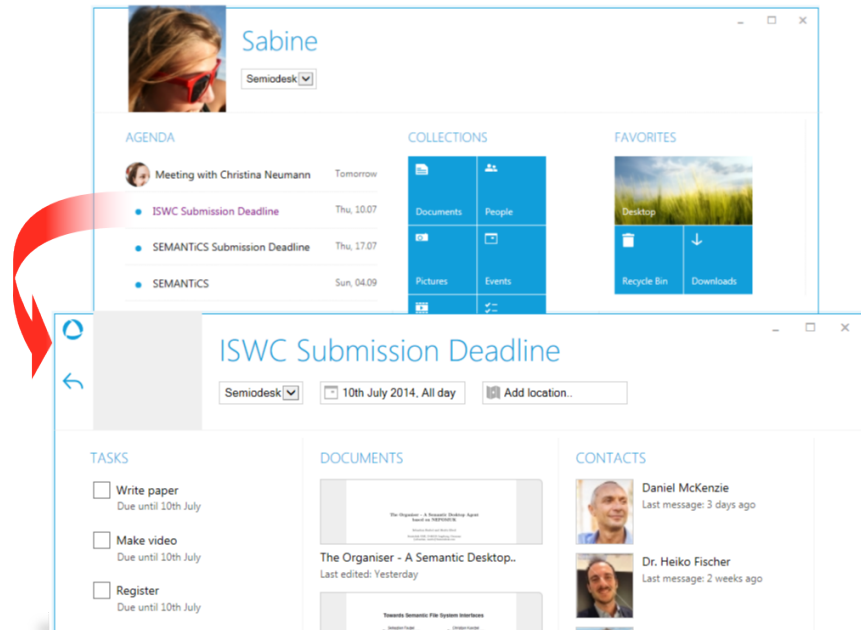


Fig. 1. Upcoming events are easily accessible from the Organiser's dashboard.

The application's dashboard is shown in figure 1. It serves as an entry point into exploring resources and provides quick access to resource collections (i.e. documents, pictures, events, etc.) and the local file system. Most prominently, it features an activity control (agenda and journal) which allows a user to plan into the future. Because future activities also serve as containers for files and related information, the dashboard provides quick access to all resources which are relevant to the user at a certain time.

Moreover, all resources can act as containers for relevant files and information. The displayed relations in the *resource view* can be hyperlinks to other resources, which enables browsing for interesting files and information. In order to assist a user in adding reasonable relations to a newly created resource, the Organiser actively analyzes the resource's properties and provides suggestions; such as to relate a collection of pictures to an event if they were taken at the time of the event. A user may accept or decline those suggestions.

Another feature of the resource view is, that it does not only consolidate already existing relations, but it also offers the ability to create new content. When adding new files, such as office documents, a file system path is being generated from the metadata of the file and the context of the resource it was created in.

2.1 File System Abstraction using Property Projection

The hierarchical file system is the de facto standard for organizing and sharing files in a productive computing environment. In order to support a soft transition away from the static file system as a primary means of organizing and browsing files, we have developed the *Property Projection* method: A Semantic Web agent that is capable of learning how resource metadata is being projected into the path component of a URI, either by analyzing existing file systems or through interactive user input (figure 2).

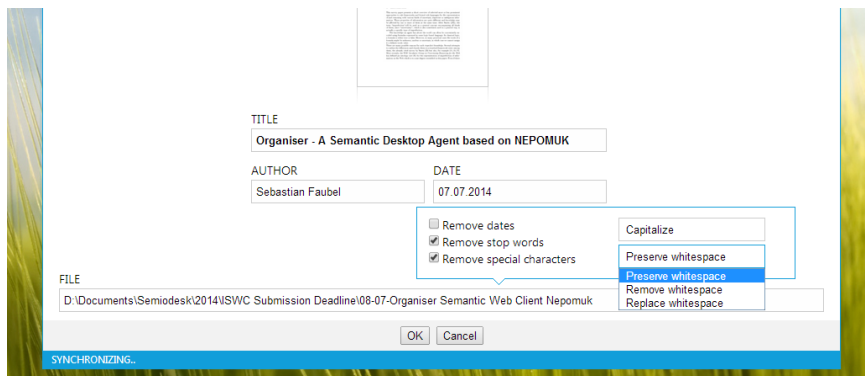


Fig. 2. Training the PropertyProjection agent interactively.

Once a projection schema has been learned, the agent can suggest storage locations for files being created in the context of a resource. This allows a user to shift from working with the static file system folders to working with resources such as contacts, events, notes or tasks that have temporal relevance to his or her activities.

The agent can formalize the schematic generation of URIs using the Property Projection Ontology. Based on the Path Projection Ontology [3], the revised ontology provides a vocabulary for the following features:

- Generating readable URIs from metadata (conforming to RFC 2396 [2])
- Generating names and titles for resources from metadata

URI schemes can be shared with other compatible agents on a global scale. This can improve overall team productivity, since a common file organization schema removes the burden to choose a storage location, and helps co-workers spend less time on looking for misfiled information.

2.2 Implementation

The Organiser is implemented using *Trinity*, our Semantic Web application development platform for .NET/Mono. It features a Semantic Object Mapping

mechanism that allows to define an object oriented abstraction layer on top of a RDF triple store. This layer promotes the use of common development methods, proven application design patterns, and significantly increases the compatibility to existing APIs.

Although the Organiser's user interface is currently implemented using WPF, the consequent use of the MVVM design pattern allows for reimplementations using other technologies such as HTML and JavaScript. The interface is laid out in such a way, that it can be used on touch screens and may be scaled down to the resolution of current smartphones.

Metadata from the file system and cloud services is gathered in the background by *Ubiquity*, a metadata extraction and synchronisation service. All changes to the extracted resources made in the Organiser are mirrored back to the metadata of the respective resource.

3 Conclusions / Future Work

The Organiser concept was refined over multiple iterations and the software is nearing completion. Only final usability tests and some connectivity and format extensions are missing.

A focus in the future is to implement the Organiser for mobile devices like Tablets and Smartphones. Because they accompany the user most of the time, we want them to act as a conduit to bring the seemingly virtual planning and organisation of the desktop into the everyday world of the user. Achieving a convergence of the user's data on mobile devices and the Desktop PC can lead to device independent productivity.

References

1. Bergman, O., Beyth-Marom, R., Nachmias, R.: The project fragmentation problem in personal information management. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. pp. 271–274. CHI '06, ACM, New York, NY, USA (2006)
2. Berners-Lee, T.: RFC 2396: Uniform Resource Identifiers (URI). Tech. rep., MIT (1998)
3. Faubel, S., Kuschel, C.: Towards semantic file system interfaces. In: Bizer, C., Joshi, A. (eds.) International Semantic Web Conference (Posters & Demos). CEUR Workshop Proceedings, vol. 401. CEUR-WS.org (2008)
4. Groza, T., Handschuh, S., Moeller, K., Grimnes, G., Sauermann, L., Minack, E., Mesnage, C., Jazayeri, M., Reif, G., Gudjnsdttir, R.: The nepomuk project - on the way to the social semantic desktop. In: Pellegrini, T., Schaffert, S. (eds.) Proceedings of I-Semantics' 07. pp. 201–211. JUCS (Sep 2007)
5. Liu, F., Shu, P., Jin, H., Ding, L., Yu, J., Niu, D., Li, B.: Gearing resource-poor mobile devices with powerful clouds: architectures, challenges, and applications. *IEEE Wireless Commun.* 20(3), 1–0 (2013)
6. Sauermann, L., Bernardi, A., Dengel, A.: Overview and outlook on the semantic desktop. In: Proceedings of the 1st Workshop on The Semantic Desktop at the ISWC 2005 Conference (2005)

HDTourist: Exploring Urban Data on Android

Elena Hervalejo¹, Miguel A. Martínez-Prieto¹, Javier D. Fernández^{1,2}, Oscar Corcho²

¹ DataWeb Research, Department of Computer Science, Univ. de Valladolid (Spain)
elena.hervalejo@gmail.com, {migumar2,jfergar}@infor.uva.es

² Ontology Engineering Group (OEG), Univ. Politécnica de Madrid (Spain)
{jdfernandez,ocorcho}@fi.upm.es

1 Introduction

The Web of Data currently comprises ≈ 62 billion triples from more than 2,000 different datasets covering many fields of knowledge³. This volume of structured Linked Data can be seen as a particular case of Big Data, referred to as *Big Semantic Data* [4]. Obviously, powerful computational configurations are traditionally required to deal with the scalability problems arising to Big Semantic Data. It is not surprising that this “data revolution” has competed in parallel with the growth of *mobile computing*. Smartphones and tablets are massively used at the expense of traditional computers but, to date, mobile devices have more limited computation resources.

Therefore, one question that we may ask ourselves would be: *can (potentially large) semantic datasets be consumed natively on mobile devices?* Currently, only a few mobile apps (e.g., [1, 9, 2, 8]) make use of semantic data that they store in the mobile devices, while many others access existing SPARQL endpoints or Linked Data directly. Two main reasons can be considered for this fact. On the one hand, in spite of some initial approaches [6, 3], there are no well-established triplestores for mobile devices. This is an important limitation because any potential app must assume both RDF storage and SPARQL resolution. On the other hand, the particular features of these devices (little storage space, less computational power or more limited bandwidths) limit the adoption of semantic data for different uses and purposes.

This paper introduces our *HDTourist* mobile application prototype. It consumes urban data from DBpedia⁴ to help tourists visiting a foreign city. Although it is a simple app, its functionality allows illustrating how semantic data can be stored and queried with limited resources. Our prototype is implemented for Android, but its foundations, explained in Section 2, can be deployed in any other platform. The app is described in Section 3, and Section 4 concludes about our current achievements and devises the future work.

2 Managing RDF in Mobile Devices

Our approach for managing RDF is inspired by the role played by SQLite⁵ in Android devices. SQLite is a self-contained SQL engine which is deployed as

³ Stats reported by LODStats: <http://stats.lod2.eu/>.

⁴ <http://dbpedia.org/>.

⁵ <http://www.sqlite.org/>.

an internal component of the application program. This way, the app itself can read and write data directly from the database files without requiring a separate process running as a DBMS (Database Management System).

Similarly, our only requirement is to hold properly serialized RDF files and a standardized interface to operate on them. Both responsibilities are provided by the RDF/HDT [5] format, which serializes RDF using up to 15 times less space than other syntaxes [4], while allowing basic SPARQL queries to be efficiently resolved on the serialized file [7]. Thus, including RDF/HDT as a library⁶ of the app, allows it to manage and query semantic data in compressed space.

3 HDTourist

HDTourist is a proof-of-concept app⁷ built on top of RDF/HDT. It is designed as a lightweight app to provide tourists with information when they are in a foreign place. In these cases, people are more reluctant to connect to Internet because of the potentially expensive costs of roaming. Thus, our mobile device will be useful to keep compressed semantic information and query it offline.

Use case. Let us suppose that we plan our trip to Riva del Garda to attend ISWC'2014, and our flight arrives to Verona. Fortunately, we have a day to visit the city and decide to use HDTourist. Before leaving home, or in a Wi-Fi hotspot (*i.e.* in the hotel), we use our Internet connection to download the RDF/HDT file with relevant information about Verona. Currently, these data are obtained by exploring different categories related to the DBpedia entity modeling the city: <http://dbpedia.org/page/Verona>. In addition to semantic data, we can download multimedia: images, maps of the region, etc. to improve the user experience. We download them and HDTourist is ready to be used in our visit.

Verona's HDT file has 18,208 triples, with a size of ≈ 850 KB, more than 4 times smaller than the original NTriples file (≈ 3.6 MB). Beyond the space savings, this HDT file is self-queryable in contrast to the flat NTriples serialization.

3.1 Retrieving Urban Data from DBpedia

DBpedia contains a lot of descriptive data about cities, which we filter as follows: given the URI u of a city (*e.g.* <http://dbpedia.org/page/Verona>), we run a CONSTRUCT query on DBpedia which retrieves: i) all triples describing the city, *i.e.*, all triples comprising u as subject, and ii) all landmarks related to the city, *i.e.*, all resources (and their descriptions) linking to u . We restrict to resources of some kind of landmarks that we have manually identified, *e.g.* resources of type *Place* (<http://dbpedia.org/ontology/Place>), *Historical Buildings* (<http://dbpedia.org/ontology/HistoricPlace>), etc. Other types specifically related to the city are considered, for instance the *squares in Verona* (<http://dbpedia.org/class/yago/PiazzasInVerona>).

The RDF subgraph returned by this CONSTRUCT query is then converted to RDF/HDT and ready to be downloaded and queried by our mobile app.

⁶ We use the Java RDF/HDT library: <https://github.com/rdfhdt/hdt-java>.

⁷ Available at: <http://dataweb.infor.uva.es/project/hdtourist/?lang=en>.

3.2 Browsing Urban Data

HDTourist uses categories to organize and display data. The main menu comprises four categories: *description*, *demography and geography*, *attractions*, and *other interesting data*. Figure 1 (a) shows the *description* of Verona, which includes basic information about the city. The information showed in each category is defined as SPARQL templates in XML configuration files (one per category), such as the following one:

```
<?xml version="1.0" encoding="utf-8"?>
<category id="Attractions">
  <name>Attractions</name>
  <group>
    <name>Squares</name>
    <sparql>
      SELECT ?label
      WHERE
      { ?place rdf:type <http://dbpedia.org/class/yago/SquaresIn${CITY}> .
        ?place rdfs:label ?label .}
      UNION
      { ?place rdf:type <http://dbpedia.org/class/yago/PiazasIn${CITY}> .
        ?place rdfs:label ?label .}
      }
    </sparql>
  </group>
  <group>
    <name>Buildings</name>
    ....
  
```

This XML excerpt corresponds to Figure 1 (b) showing the category “Attractions”, which includes *Squares*, *Buildings*, etc. Each group retrieves the label of attractions with a SPARQL query which typically consists of a UNION of Basic Graph Patterns searching for certain types of resources, as shown in the excerpt. When parsing the XML, the template $\${CITY}$ is converted to the appropriated name, *e.g.* *Verona*. Each SPARQL query is then resolved making use of the query API of RDF/HDT, retrieving the label shown in the screen layout.

As shown in Figure 1 (c), each landmark can be expanded, obtaining further information. In this screenshot, we choose the “Piazza delle Erbe” (within “Squares”), and the app retrieves the triples describing it. The concrete information to be shown in the landmark description is also configured by means of an XML file containing one SPARQL template per category, again resolved against the local RDF/HDT. As shown in the screenshot, pictures can be downloaded and stored offline. Finally, HDTourist is able to show geolocated landmarks in interactive maps, as shown in Figure 1 (d) for “Piazza delle Erbe”. The app uses Google maps by default, but offline maps⁸ can be downloaded beforehand.

4 Conclusions and Future Work

The offline capacities and structured information consumption possibilities of mobile devices are still several order of magnitudes below traditional devices. With our demo we show that RDF/HDT can be used as a self-contained engine to retrieve RDF information in mobile devices. To date, we have explored a given set of cities and certain query templates to build the screen layout. We are now exploring a spreading activation mechanism to automatically retrieve interesting

⁸ In this prototype we use Nutiteq SDK Maps, available at <http://www.nutiteq.com/>.

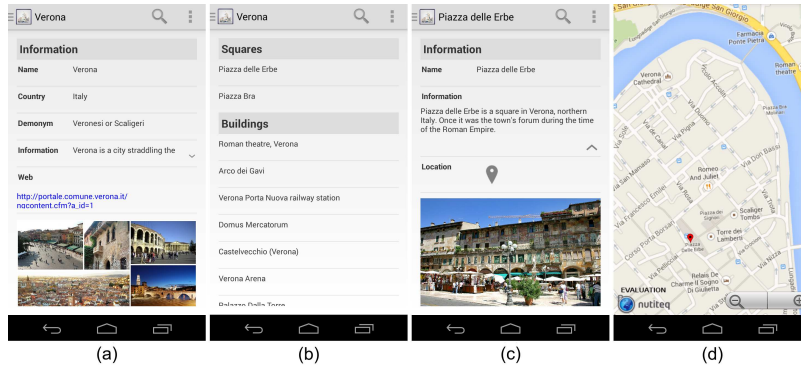


Fig. 1. Some screenshots of HDTourist.

features of a city which are then converted to HDT on the server side. This also takes into account other datasets besides DBpedia.

Acknowledgments

This work has been funded by the European Commission under the grant PlanetData (FP7-257641) and by the Spanish Ministry of Economy and Competitiveness (TIN2013-46238-C4-2-R).

References

1. C. Becker and C. Bizer. DBpedia Mobile: A Location-Enabled Linked Data Browser. In *Proc. of LDOW*, CEUR-WS 369, paper 14, 2008.
2. A.E. Cano, A.S. Dadzie, and M. Hartmann. Who's Who—A Linked Data Visualisation Tool for Mobile Environments. In *Proc. of ISWC*, pages 451–455, 2011.
3. J. David, J. Euzenat, and M. Rosoiu. Linked Data from your Pocket. In *Proc. of DownScale*, CEUR-WS 844, paper 2, pages 6–13, 2012.
4. J.D. Fernández, M. Arias, M.A. Martínez-Prieto, and C. Gutiérrez. Management of Big Semantic Data. In *Big Data Computing*, chapter 4. Taylor & Francis, 2013.
5. J.D. Fernández, M.A. Martínez-Prieto, C. Gutiérrez, and A. Polleres. *Binary RDF Representation for Publication and Exchange (HDT)*. W3C Member Submission, 2011. <http://www.w3.org/Submission/2011/03/>.
6. D. Le-Phuoc, J.X. Parreira, V. Reynolds, and M. Hauswirth. RDF On the Go: An RDF Storage and Query Processor for Mobile Devices. In *Proc. ISWC*, CEUR-WS 658, paper 19, 2010.
7. M.A. Martínez-Prieto, M. Arias, and J.D. Fernández. Exchange and Consumption of Huge RDF Data. In *Proc. of ESWC*, pages 437–452, 2012.
8. V.C. Ostuni, G. Gentile, T. Di Noia, R. Mirizzi, D. Romito, and E. Di Scascio. Mobile Movie Recommendations with Linked Data. In *Proc. of CD-ARES*, pages 400–415, 2013.
9. G. Parra, J. Klerkx, and E. Duval. More!: Mobile Interaction with Linked Data. In *Proc. of DCI*, CEUR-WS 817, paper 4, 2010.

Integrating NLP and SW with the KnowledgeStore

Marco Rospocher, Francesco Corcoglioniti, Roldano Cattoni,
Bernardo Magnini, and Luciano Serafini

Fondazione Bruno Kessler—IRST, Via Sommarive 18, Trento, I-38123, Italy
{rospocher, corcoglio, cattoni, magnini, serafini}@fbk.eu

Abstract. We showcase the KnowledgeStore (KS), a scalable, fault-tolerant, and Semantic Web grounded storage system for interlinking unstructured and structured contents. The KS contributes to bridge the unstructured (e.g., textual document, web pages) and structured (e.g., RDF, LOD) worlds, enabling to jointly store, manage, retrieve, and query, both typologies of contents.

1 Introduction: Motivations and Vision

Despite the widespread diffusion of structured data sources and the public acclaim of the Linked Open Data (LOD) initiative, a preponderant amount of information remains nowadays available only in unstructured form, both on the Web and within organizations. While different in form, structured and unstructured contents are often related in content, as they speak about the very same entities of the world (e.g., persons, organizations, locations, events), their properties, and relations among them. Despite the last decades achievements in Natural Language Processing (NLP), now supporting large scale extraction of knowledge about entities of the world from unstructured text, frameworks enabling the seamless integration and linking of knowledge coming both from structured and unstructured contents are still lacking.¹

In this demo we showcase the KnowledgeStore (KS), a scalable, fault-tolerant, and Semantic Web grounded storage system to jointly store, manage, retrieve, and query, both structured and unstructured data. Fig. 1a shows schematically how the KS manages unstructured and structured contents in its three *representation layers*. On the one hand (and similarly to a file system) the *resource layer* stores unstructured content in the form of resources (e.g., news articles), each having a textual representation and some descriptive metadata. On the other hand, the *entity layer* is the home of structured content, that, based on Knowledge Representation and Semantic Web best practices, consists of *axioms* (a set of ⟨subject, predicate, object⟩ triples), which describe the *entities* of the world (e.g., persons, locations, events), and for which additional metadata are kept to track their provenance and to denote the formal *contexts* where they hold (e.g., point of view, attribution). Between the aforementioned two layers there is the *mention layer*, which indexes *mentions*, i.e., snippets of resources (e.g., some characters in a text document) that denote something of interest, such as an entity or an axiom of the entity layer. Mentions can be automatically extracted by NLP tools, that can enrich them with additional attributes about how they denote their referent (e.g., with which name, qualifiers, “sentiment”). Far from being simple pointers, mentions present both unstructured

¹ See [1] for an overview of works related to the contribution presented in this demo.

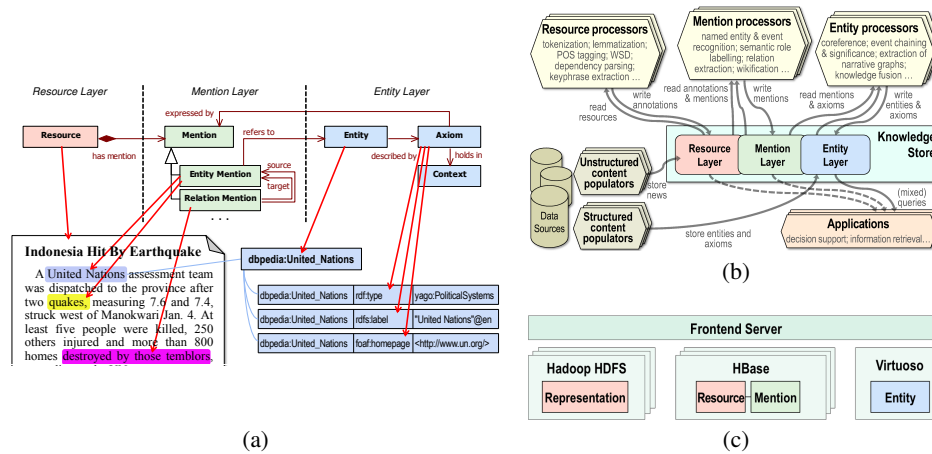


Fig. 1: (a) The three KS layers; (b) Interactions with external modules; (c) Components.

and structured facets (respectively snippet and attributes) not available in the resource and entity layers alone, and are thus a valuable source of information on their own.

Thanks to the explicit representation and alignment of information at different levels, from unstructured to structured knowledge, the KS supports a number of usage scenarios. It enables the development of enhanced applications, such as effective *decision support systems* that exploit the possibility to semantically query the content of the KS with requests combining structured and unstructured content, such as “retrieve all the documents mentioning that person Barack Obama participated to a sport event”. Then, it favours the design and empirical investigation of information processing tasks otherwise difficult to experiment with, such as cross-document *coreference resolution* (i.e., identifying that two mentions refer to the same entity of the world) exploiting the availability of interlinked structured knowledge. Finally, the joint storage of (i) extracted knowledge, (ii) the resources it derives from, and (iii) extracted metadata provides an ideal scenario for developing, training, and evaluating ontology population techniques.

2 An overview of the KnowledgeStore

In this section we briefly outline the main characteristics of the KS. For a more exhaustive presentation of the KS design, we point the reader to [1]. More documentation, as well as binaries and source code,² are all available on the KS web site [2].

Data Model The data model defines what information can be stored in the KS. It is organized in three layers (resource, mention and entity), with properties that relate objects across them. To favour the exposure of the KS content according to LOD principles, the data model is defined as an OWL 2 ontology (available on [2]). It contains the TBox definitions and restrictions for each model element and can be extended on a per-deployment basis, e.g., with domain-specific resource and linguistic metadata.

API The KS presents a number of interfaces through which external clients may access and manipulate stored data. Several aspects have been considered in defining them

² Released under the terms of the Apache License, Version 2.0.

(e.g., operation granularity, data validation). These interfaces are offered through two HTTP ReST endpoints. The *CRUD endpoint* provides the basic operations to access and manipulate (CRUD: create, retrieve, update, and delete) any object stored in any of the layers of the KS. Operations of the CRUD endpoint are all defined in terms of sets of objects, in order to enable bulk operations as well as operations on single objects. The *SPARQL endpoint* allows to query axioms in the entity layer using SPARQL. This endpoint provides a flexible and Semantic Web-compliant way to query for entity data, and leverages the grounding of the KS data model in Knowledge Representation and Semantic Web best practices. A Java client is also offered to ease the development of (Java) client applications.

Architecture At its core, the KS is a storage server whose services are utilized by external clients to store and retrieve the contents they process. From a functional point of view, we identify three main typologies of clients (see Fig. 1b): (i) *populators*, whose purpose is to feed the KS with basic contents needed by other applications (e.g., documents, background knowledge from LOD sources); (ii) *linguistic processors*, that read input data from the KS and write back the results of their computation; and, (iii) *applications*, that mainly read data from the KS (e.g., decision support systems). Internally, the KS consists of a number of software components (see Fig. 1c) distributed on a cluster of machines: (i) the *Hadoop HDFS* filesystem provides a reliable and scalable storage for the physical files holding the representations of resources (e.g., texts and linguistic annotations of news articles); (ii) the *HBase* column-oriented store builds on Hadoop to provide database services for storing and retrieving semi-structured information about resources and mentions; (iii) the *Virtuoso* triple-store stores axioms to provide services supporting reasoning and online SPARQL query answering; and, (iv) the *Frontend Server* has been specifically developed to implement the operations of the CRUD and SPARQL endpoints on top of the components listed above, handling global issues such as access control, data validation and operation transactionality.

User Interface (UI) The KS UI (see Fig. 2) enables human users to access and inspect the content of the KS via two core operations: (i) the *SPARQL query* operation, with which arbitrary SPARQL queries can be run against the KS SPARQL endpoint, obtaining the results directly in the browser or as a downloadable file (in various file formats, including the recently standardized JSON-LD); and, (ii) the *lookup* operation, which given the URI of an object (i.e., resource, mention, entity), retrieves all the KS content about that object. These two operations are seamlessly integrated in the UI, to offer a smooth browsing experience to the users.

3 Showcasing the KnowledgeStore and concluding remarks

During the Posters and Demos session, we will demonstrate live how to access the KS content via the UI (similarly to the detailed demo preview available at [3]), highlighting the possibilities offered by the KS to navigate back and forth from unstructured to structured content. For instance, we will show how to run arbitrary SPARQL queries, retrieving the mentions of entities and triples in the query result set, and the documents where they occur. Similarly, starting from a document URI, we will show how to access the mentions identified in the document, up to the entities and triples they refer to.

KnowledgeStore UI Lookup SPARQL query

ID example URI 1 mention found

Mention resource (excerpt): <./22251818>

Fifa launched its reform process almost two years ago amid fierce criticism after Mohamed bin Hammam, an election rival to president Sepp Blatter, was accused of bribery. Bin Hammam was later **banned** for life by Fifa, but he continues to deny any wrongdoing.

Mention data

ID	<./22251818#char=741,747&word=w131&term=t131>
ks:mentionOf	<./22251818>
nwr:eventClass	nwr:event_speech_cognitive
nwr:factualityConfidence	0.7226601421959593
nwr:framenetRef	framenet:Prohibiting
nwr:pos	nwr:pos_verb
nwr:pred	ban
nwr:propbankRef	<./ban.01>
nwr:verbnetRef	<./forbid-67>
nif:beginIndex	741
nif:endIndex	747
rdftype	ks:Mention nwr:EntityMention nwr:EventMention nwr:TimeOrEventMention

Mention referent (8 triples, max 1000): <./22251818#banEvent>

subject	predicate	object	graph
<./22251818#banEvent>	rdftype	<./communication>	<./instances>
<./22251818#banEvent>	rdftype	framenet:Prohibiting	<./instances>
<./22251818#banEvent>	rdftype	sem:Event	<./instances>
<./22251818#banEvent>	rdfs:label	ban	<./instances>
<./22251818#banEvent>	gsf:denotedBy	<./22251818#char=741,747&word=w131&term=t131>	<./instances>
<./22251818#banEvent>	sem:hasActor	dbpedia:FIFA	<./main>
<./22251818#banEvent>	sem:hasActor	dbpedia:Mohammed_bin_Hammam	<./main>
<./22251818#banEvent>	sem:hasTime	<./22251818#nafHeader_fileDesc_creationTime>	<./main>

Fig. 2: KS UI. Lookup of a mention. Note the three boxes (Mention resource, Mention Data, Mention Referent) corresponding to the three representation layers of the KS.

In the last few months, several running instances of the KS were set-up (on a cluster of 5 average specs servers) and populated using the NewsReader Processing Pipeline [4] with contents coming from various domains: to name a few, one on the global automotive industry [5] (64K resources, 9M mentions, 316M entity triples), and one related to the FIFA World Cup (212K resources, 75M mentions, 240M entity triples). The latter, which will be used for the demo, was exploited during a Hackathon event [6], where 38 web developers accessed the KS to build their applications (over 30K SPARQL queries were submitted – on average 1 query/s, with peaks of 25 queries/s).

Acknowledgements The research leading to this paper was supported by the European Union’s 7th Framework Programme via the NewsReader Project (ICT-316404).

References

1. Corcoglioniti, F., Rospoche, M., Cattoni, R., Magnini, B., Serafini, L.: Interlinking unstructured and structured knowledge in an integrated framework. In: 7th IEEE International Conference on Semantic Computing (ICSC), Irvine, CA, USA. (2013)
2. <http://knowledgestore.fbk.eu>
3. <http://youtu.be/if1PRwS115c>
4. <https://github.com/newsreader/>
5. <http://datahub.io/dataset/global-automotive-industry-news>
6. <http://www.newsreader-project.eu/come-hack-with-newsreader/>

Graphical Representation of OWL 2 Ontologies through Graphol

Marco Console, Domenico Lembo, Valerio Santarelli, and Domenico Fabio Savo

Dipartimento di Ingegneria Informatica, Automatica e Gestionale “Antonio Ruberti”
SAPIENZA Università di Roma
{console, lembo, santarelli, savo}@dis.uniroma1.it

Abstract. We present Graphol, a novel language for the diagrammatic representation of ontologies. Graphol is designed to offer a completely visual representation to the users, thus helping the understanding of people not skilled in logic. At the same time, it provides designers with simple mechanisms for ontology editing, which free them from having to write down complex textual syntax. Through Graphol we can specify $SRQLQ(D)$ ontologies, thus our language essentially captures the OWL 2 standard. In this respect, we developed a basic software tool to translate Graphol ontologies realized with the yEd graph editor into OWL 2 functional syntax specifications.

1 Introduction

Ontologies have become popular in recent years in several contexts, such as biomedicine, life sciences, e-commerce, enterprise applications [9]. Obviously, it is very likely that people operating in such contexts are not experts in logic and generally do not possess the necessary skills to interpret formulas through which ontologies are typically expressed. This turns out to be a serious problem also in the development of an ontology. Indeed, ontologists usually work together with domain experts, the former providing their knowledge about ontology modelling and languages, the latter providing their expertise on the domain of interest. During this phase, communication between these actors is fundamental to produce a correct specification.

The use of a graphical representation for ontologies is widely recognized as a means to mitigate this communication problem. At the same time, the possibility of specifying ontologies in a graphical way might bring software analysts and experts in conceptual modelling to approach ontology modelling, since they would be provided with mechanisms that are close in spirit to those they usually adopt for software design.

Various proposals in this direction exist in the literature, but to date graphical languages for ontology have not become very popular, especially for the editing task. Among various reasons, we single out the following points: (i) many languages for graphical representation of ontologies do not capture the current standard OWL 2, and their extension to it is not straightforward (see, e.g., [3,8,7,6,1]); (ii) other proposals require the use of formulas mixed with the graphical representation (see, e.g., [4,1]); (iii) popular ontology management tools, such as Protégé¹ or TopBraid Composer², offer visualization functionalities, but do not support a completely graphical editing.

¹ <http://protege.stanford.edu>

² <http://www.topquadrant.com/tools>

To meet the main disadvantages mentioned above, in this paper we present our proposal for graphical specification and visualization of ontologies, and introduce the novel *Graphol* language, whose main characteristics can be summarized as follows:

- Graphol is completely graphical (no formulae need to be used in our diagrams) and adopts a limited number of symbols. In Graphol, an ontology is a graph, whose nodes represent either predicates from the ontology alphabet or constructors used to build complex expressions from named predicates. Then, two kinds of edges are adopted: input edges, used to specify arguments of constructors, and inclusion edges, used to denote inclusion axioms between (complex) expressions.
- Graphol has a precise syntax and semantics, which is given through a natural encoding in Description Logics.
- Such encoding shows that Graphol subsumes $\mathcal{SROIQ}(D)$, the logical underpinning of OWL 2.
- Graphol is rooted in a standard language for conceptual modeling: the basic components of Graphol are taken from the Entity-Relationship (ER) model. Notably, simple ontologies that correspond to classical ER diagrams (e.g., some OWL 2 QL ontologies) have in Graphol a representation that is isomorphic to the ER one.
- Graphol comes with some basic tools that support both the graphical editing and the automatic translation of the diagrams into a corresponding OWL 2 specification, to foster the interoperation with standard OWL reasoners and development environments.

We have adopted Graphol in various industrial projects, where we have produced large ontologies with hundreds of predicates and axioms. In such projects we could verify the effectiveness of the language for communicating with domain experts. At the same time, we exploited Graphol in the editing phase: all ontologies realized in these projects have been indeed completely specified in our graphical language, whereas an OWL functional syntax encoding thereof has been obtained automatically through the use of our translator tool. One of these experiences is described in [2], where the impact of the use of Graphol on the quality of the realized ontology is widely discussed.

We also conducted some user evaluation tests, where both designers skilled in conceptual modelling (but with no or limited experience in ontology modelling) and users without specific logic background were involved. From these tests, we obtained promising results about the effectiveness of our language for both visualizing and editing ontologies. A complete description of our evaluation study is given in [5].

For a complete description of both the syntax and the semantics of Graphol we refer the reader to [5] and to the Graphol web site³, where it is also possible to download currently available software tools for our language. In the rest of the paper we instead discuss how the Graphol demonstration will be carried out.

2 The Graphol demonstration

In this demo we will show the process we devised to obtain an OWL 2 ontology starting from the specification of a Graphol diagram. Such a process relies on both existing

³ <http://www.dis.uniroma1.it/~graphol>

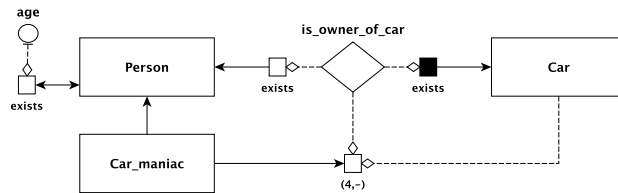


Fig. 1: A simple Graphol ontology

open source tools and original software components. More in detail, to draw a Graphol ontology we make use of the yEd editor for graphs⁴, which we equip with a palette containing all and only the symbols needed for Graphol. yEd allows us to save the ontology in GraphML⁵, a popular XML-based file format for encoding graphs.

An example of a Graphol ontology obtained through yEd is given in Figure 1. In the figure, the reader can see that in Graphol classes (i.e., *Person*, *Car_maniac*, *Car*), object properties (i.e., *is_owner_of_car*), and data properties (i.e., *age*) are modeled by labeled rectangles, diamonds, and circles, respectively, similarly to ER diagrams. The white (resp. black) square labeled with *exists* is a graphical constructor that takes as input a property, through a dashed arrow whose end node is a small diamond, and returns the domain (resp. the range) of the property. Such squares can have also different labels, to denote different constructs. In the example, the label $(4, -)$ on the white square taking as input the *is_owner_of_car* property specifies a cardinality restriction on the domain of such property, i.e., it denotes all individuals participating at least 4 times to *is_owner_of_car*. The solid arrow always indicates a subsumption relation. This means that *Car_maniac* is a subclass of *Person*, and also of the complex class obtained through the cardinality restriction, which implies that a car maniac owns at least four cars.

Furthermore, the ontology in the example says that the domain of *is_owner_of_car* is *Person*, its range is *Car*, and also that each *Person* has an *age*, and that the domain of *age* is *Person*. Also, the additional dash orthogonal to the edge connecting *age* to its domain specifies that this property is functional.

The above example uses only a limited sets of constructors available in Graphol. Participants to the demo will be provided with the yEd editor and the Graphol palette to draw their own ontologies, experiencing the entire expressive power of the language.

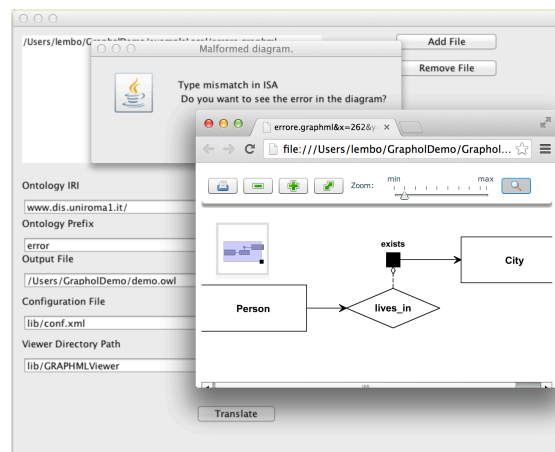


Fig. 2: The Graphol2OWL tool

⁴ http://www.yworks.com/en/products_yed_about.html

⁵ <http://graphml.graphdrawing.org/>

To both check the correctness of the specification and translate it into OWL 2, we developed a dedicated tool. The tool provides a syntactic validation of a given diagram: while parsing the GraphML file, if a portion of the graph is found that does not respect the Graphol syntax, the tool reports an error to the user in a pop-up window and visualizes this portion by means of an external yEd viewer. A screenshot of this tool showing an error identified in a Graphol diagram is given in Figure 2. In this example, the error consists in linking a class to a property with a solid arrow, which actually corresponds to a wrong subsumption between a concept and role.

The translator to obtain OWL 2 encodings from Graphol will be used during the demo. We will also show the compatibility of the produced OWL 2 functional syntax file with popular tools for ontology editing and management, like Protégé⁶.

3 Future Work

Our main future work on Graphol is the development of editing tools (stand-alone systems or plugins of existing ontology development environments) tailored to the specification of ontologies in our graphical language and integrated with state-of the art reasoners. At the same time, we are working to improve ontology visualization in Graphol, by investigating mechanisms to automatically extract ontology views at different levels of detail on the basis of specific user requests.

References

1. do Amaral, F.N.: Model outlines: A visual language for DL concept descriptions. *Semantic Web J.* 4(4), 429–455 (2013)
2. Antonioli, N., Castanò, F., Coletta, S., Grossi, S., Lembo, D., Lenzerini, M., Poggi, A., Virardi, E., Castracane, P.: Ontology-based data management for the italian public debt. In: *Proc. of FOIS (2014)*, to Appear
3. Brockmans, S., Volz, R., Eberhart, A., Löffler, P.: Visual modeling of OWL DL ontologies using UML. In: *Proc. of ISWC*. pp. 198–213. Springer (2004)
4. Cerans, K., Ovcinnikova, J., Liepins, R., Sprogis, A.: Advanced OWL 2.0 ontology visualization in OWLGrEd. In: *In Proc. of DB&IS*. pp. 41–54 (2012)
5. Console, M., Lembo, D., Santarelli, V., Savo, D.F.: The Graphol language for ontology specification, available at <http://www.dis.uniroma1.it/~graphol/documentation/GrapholLVPrel.pdf>
6. Dau, F., Eklund, P.W.: A diagrammatic reasoning system for the description logic \mathcal{ALC} . *J. Vis. Lang. Comput.* 19(5), 539–573 (2008)
7. Krivov, S., Williams, R., Villa, F.: GrOWL: A tool for visualization and editing of OWL ontologies. *J. of Web Semantics* 5(2), 54–57 (2007)
8. Object Management Group: Ontology definition metamodel. Tech. Rep. formal/2009-05-01, OMG (2009), available at <http://www.omg.org/spec/ODM/1.0>
9. Staab, S., Studer, R. (eds.): *Handbook on Ontologies*. International Handbooks on Information Systems, Springer, 2nd edn. (2009)

⁶ A preview of the demo is available at <http://www.dis.uniroma1.it/~graphol/research.html>.

LIVE: a Tool for Checking Licenses Compatibility between Vocabularies and Data

Guido Governatori^{1*}, Ho-Pun Lam¹, Antonino Rotolo², Serena Villata³,
Ghislain Ateazing⁴, and Fabien Gandon³

¹ NICTA Queensland Research Laboratory
firstname.lastname@nicta.com.au

² University of Bologna
antonino.rotolo@unibo.it

³ INRIA Sophia Antipolis
firstname.lastname@inria.fr

⁴ Eurecom
auguste.ateazing@eurecom.fr

Abstract In the Web of Data, licenses specifying the terms of use and reuse are associated not only to datasets but also to vocabularies. However, even less support is provided for taking the licenses of vocabularies into account than for datasets, which says it all. In this paper, we present a framework called LIVE able to support data publishers in verifying licenses compatibility, taking into account both the licenses associated to the vocabularies and those assigned to the data built using such vocabularies.

1 Introduction

The license of a dataset in the Web of Data can be specified within the data, or outside of it, for example in a separate document linking the data. In line with the Web of Data philosophy [3], licenses for such datasets should be specified in RDF, for instance through the Dublin Core vocabulary¹. Despite such guidelines, still a lot of effort is needed to enhance the association of licenses to data on the Web, and to process licensed material in an automated way. The scenario becomes even more complex when another essential component in the Web of Data is taken into account: the vocabularies. Our goal is to support the data provider in assigning a license to her data, and verifying its compatibility with the licenses associated to the adopted vocabularies. We answer this question by proposing an online framework called LIVE² (LIncenses VErification) that exploits the formal approach to licenses composition proposed in [2] to verify the compatibility of a set of heterogeneous licenses. LIVE, after retrieving the licenses associated to the vocabularies used in the dataset under analysis, supports data providers in verifying whether the license assigned to the dataset is compatible with those of the vocabularies, and returns a warning when this is not the case.

* NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Centre of Excellence program.

¹ <http://purl.org/dc/terms/license>

² The online tool is available at <http://www.eurecom.fr/~ateazing/licenseChecker/>

2 The LIVE framework

The LIVE framework is a Javascript application, combining HTML and Bootstrap. Hence, installation has no prerequisite. Since the tool is written in Javascript, the best way to monitor the execution time is with the `performance.now()` function. We use the 10 LOD datasets with the highest number of links towards other LOD datasets available at <http://lod-cloud.net/state/#links>. For each of the URLs in Datahub, we retrieve the VoID³ file in Turtle format, and we use the `voidChecker` function⁴ of the LIVE tool to retrieve the associated license, if any. The input of the LIVE framework (Figure 1) consists in the dataset (URI or VOiD) whose license has to be verified. The framework is composed by two modules. The first module takes care of retrieving the vocabularies used in the dataset, and for each vocabulary, retrieves the associate license⁵ (if any) querying the LOV repository. The second module takes as input the set of licenses (i.e., the licenses of the vocabularies used in the dataset as well as the license assigned to the dataset) to verify whether they are compatible with each others. The result returned by the module is a *yes/no* answer. In case of negative answer, the data provider is invited to change the license associated to the dataset and check back again with the LIVE framework whether further inconsistencies arise.

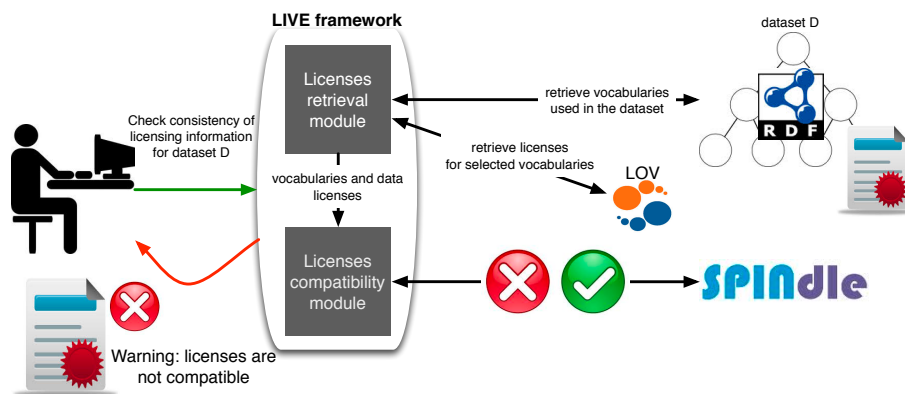


Figure 1. LIVE framework architecture.

Retrieving licensing information from vocabularies and datasets. Two use-cases are taken into account: a SPARQL endpoint, or a VoID file in Turtle syntax. In the first use case, the tool retrieves the named graphs present in the repository, and then the user is asked to select the URI of the graph that needs to be checked. Having that information, a SPARQL query is triggered, looking for entities declared as `owl:Ontology`,

³ <http://www.w3.org/TR/void/>

⁴ <http://www.eurecom.fr/~atamezin/licenseChecker/voidChecker.html>

⁵ Note that the LIVE framework relies on the dataset of machine-readable licenses (RDF, Turtle syntax) presented in [1].

voaf:Vocabulary or object of the void:vocabulary property. The final step is to look up the LOV catalogue to check whether they declare any license. There are two options for checking the license: (i) a “strict checking” where the FILTER clause contains exactly the namespace of the submitted vocabulary, or (ii) a “domain checking”, where only the domain of the vocabulary is used in the FILTER clause. This latter option is recommended in case only one vocabulary has to be checked for the license. In the second use case, the module parses a VOID file using a N3 parser for Javascript⁶, and then collects the declared vocabularies in the file, querying again LOV⁷ to check their licensing information. When the URIs of the licenses associated to the vocabularies and the dataset are retrieved, the module retrieves the machine-readable description of the licenses in the dataset of licenses [1].

Licenses compatibility verification. The logic proposed in [2] and the licenses compatibility verification process has been implemented using SPINdle [4] – a defeasible logic reasoner capable of inferencing defeasible theories with hundredth of thousand rules.

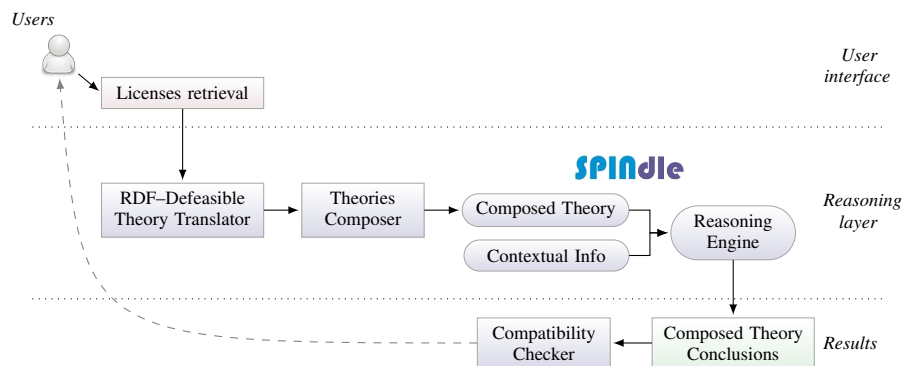


Figure 2. Licenses compatibility module.

As depicted in Figure 2, after receiving queries from users, the selected licenses (represented using RDF) will be translated into the DFL formalism supported by SPINdle using the *RDF-Defeasible Theory Translator*. That is, each RDF-triple will be translated into a defeasible rule based on the subsumption relation between the *subject* and *object* of a RDF-triples. In our case, we can use the subject and object of the RDF-triples as the antecedent and head of a defeasible rule, respectively. Besides, the translator also supports direct import from the Web and processing of RDF data into SPINdle theories.

The translated defeasible theories will then be composed into a single defeasible theory based on the logic proposed in [2], using the *Theories Composer*. Afterwards, the

⁶ <https://github.com/RubenVerborgh/N3.js>

⁷ Since LOV endpoint does not support the JSON format in the results, we have uploaded the data in eventmedia.eurecom.fr/sparql.

composed theory, together with other contextual information (as defined by user), will be loaded into the SPINdle reasoner to perform a compatibility check before returning the results to the users.

We have evaluated the time performances of the LIVE framework in two steps. First, we evaluate the time performances of the licenses compatibility module: it needs about 6ms to compute the compatibility of two licenses. Second, we evaluate time performances (Chrome v. 34) of the whole LIVE framework for the 10 LOD datasets with the highest number of links towards other LOD datasets, considering both the licenses retrieval module and the licenses compatibility one. The results show that LIVE provides the compatibility evaluation in less than 5 seconds for 7 of the selected datasets. Time performances of LIVE are mostly affected by the first module while the compatibility module does not produce a significant overhead. For instance, consider Linked Dataspaces⁸, a dataset where we retrieve the licensing information in both the dataset and the adopted vocabularies. In this case, LIVE retrieves in 13.20s 48 vocabularies, the license for the dataset is CC-BY, and the PDDL license is attached one of the vocabularies⁹. The time for verifying the compatibility is 8ms, leading to a total of 13.208s.

3 Future perspectives

We have introduced the LIVE framework for licenses compatibility. The goal of the framework is to verify the compatibility of the licenses associated to the vocabularies exploited to create a RDF dataset and the license associated to the dataset itself. Several points have to be taken into account as future work. More precisely, in the present paper we consider vocabularies as data but this is not the only possible interpretation. For instance, we may see vocabularies as a kind of compiler, such that, after the creation of the dataset then the external vocabularies are no more used. In this case, what is a suitable way of defining a compatibility verification? We will investigate this issue as well as we will evaluate the usability of the online LIVE tool to subsequently improve the user interface.

References

1. Cabrio, E., Arosio, A.P., Villata, S.: These are your rights: A natural language processing approach to automated rdf licenses generation. In: ESWC2014, LNCS (2014)
2. Governatori, G., Rotolo, A., Villata, S., Gandon, F.: One license to compose them all - a deontic logic approach to data licensing on the web of data. In: International Semantic Web Conference (1). Lecture Notes in Computer Science, vol. 8218, pp. 151–166. Springer (2013)
3. Heath, T., Bizer, C.: Linked Data: Evolving the Web into a Global Data Space. Morgan & Claypool (2011)
4. Lam, H.P., Governatori, G.: The making of SPINdle. In: Proceedings of RuleML, LNCS 5858. pp. 315–322. Springer (2009)

⁸ <http://270a.info/>

⁹ <http://purl.org/linked-data/cube>

The Map Generator Tool*

Valeria Fionda¹, Giuseppe Pirrò², Claudio Gutierrez³,

¹ Department of Mathematics, University of Calabria, Italy

² WeST, University of Koblenz-Landau, Germany

³ DCC, Universidad de Chile, Chile

Abstract. We present the MaGe system, which helps users and developers to build maps of the Web graph. Maps *abstract* and represent in a *concise* and machine-readable way *regions* of information on the Web.

1 Introduction

The Web is a large and interconnected information space (usually modeled as a graph) commonly accessed and explored via navigation enabled by browsers. To cope with the size of this huge (cyber)space, Web users need to track, record and specify conceptual regions on the Web (e.g., a set of Web pages; friends and their interests; a network of citations), for their own use, for exchanging, for further processing. Users often navigate large fragments of the Web, to discover and isolate very few resources of interest and struggle to keep *connectivity information* among them. The idea of a map of a Web region is essentially that of *representing* in a *concise way* information in the region in terms of *connectivity* among a set of *distinguished* resources (nodes).

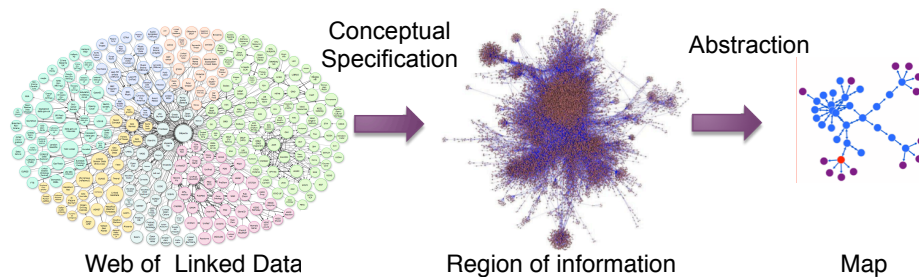


Fig. 1. Building maps of the Web.

With the advent of the Web of Data [4], maps to describe and navigate information on the Web in a machine-processable way become more feasible. The key new technical support is: (i) the availability of a standard infrastructure, based on the Resource Description Framework (RDF), for the publishing/interlinking of structured data on the Web; (ii) a community of developers; (iii) languages to specify regions of information on the Web [1]. Fig. 1 sketches the high level process of building maps of the Web via the Map Generator (MaGe) system.

* V. Fionda was supported by the European Commission, the European Social Fund and the Calabria region.

2 The Map framework

The idea of a map on the Web is to represent in a concise and comprehensive way connectivity information between pairs of distinguished nodes. Given a conceptual region G of information on the Web, there can be several maps of G with different level of detail (i.e., nodes and edges to be included).

Formally, let $\Gamma = (V_\Gamma, E_\Gamma)$ be a Web region, where V_Γ and E_Γ are the set of nodes and edges respectively. Then:

- $u \rightarrow v$ denotes an edge $(u, v) \in E$.
- $u \twoheadrightarrow v$ denotes a path from u to v in Γ .
- Let $N \subseteq V$. Then, $u \twoheadrightarrow_N v$ if and only if there is a path from u to v in Γ not passing through intermediate nodes in N .

Let $V_M \subseteq V$ be the set of distinguished nodes of the Web region $\Gamma = (V_\Gamma, E_\Gamma)$, i.e., those that we would like to represent.

Definition 1 (Map) A map $M = (V_M, E_M)$ of $\Gamma = (V_\Gamma, E_\Gamma)$ is a graph such that $V_M \subseteq V_\Gamma$ and each edge $(x, y) \in E_M$ implies $x \twoheadrightarrow y$ in Γ .

A basic (and highly used) example of map of the Web are bookmarks. In this case, V_M is the set of nodes highlighted or marked, and $E_M = \emptyset$, that is, there is no connectivity recorded among them. An important idea is that of a *good* map, i.e., a map which represents connectivity among the distinguished nodes and avoids redundant edges [3].

Definition 2 (Good map) A map $M = (V_M, E_M)$ of $\Gamma = (V_\Gamma, E_\Gamma)$ is good if and only if:

1. $\forall x, y \in V_M$ $x \twoheadrightarrow_{V_M} y$ in Γ implies $x \rightarrow y$ in M
2. $\forall x, y \in V_M$ $x \rightarrow y$ in M implies $x \twoheadrightarrow_{V_M} y$ in Γ .

Good maps have the nice properties (i) uniqueness and (ii) low complexity of computation. Indeed, given a region $\Gamma = (V_\Gamma, E_\Gamma)$ and a set of distinguished nodes $V_M \subseteq V_\Gamma$ there exists a unique good map $M = (V_M, E_M)$ that is computable in $\mathcal{O}(|V_M| \times (|V_\Gamma \setminus V_M| + |E_\Gamma|))$ by an adaptation of the BFS algorithm.

3 MaGe: Building Maps of the Web

Maps are built on top of regions of information on the Web. To automate the process of generating regions, MaGe uses the NAUTILOD [1,2] language. Given an expression ex , NAUTILOD enables to extract a Web region $\Gamma = (V_\Gamma, E_\Gamma)$ such that V_Γ and E_Γ are the set of nodes and edges *visited* while evaluating ex . Once the region has been obtained, MaGe computes good maps as sketched in Section 2 considering the set of distinguished nodes $V_M = \{s\} \cup T$, where s is the node in the region that corresponds to the seed URI where the evaluation of ex starts and T are the nodes satisfying ex .

MaGe has been implemented in Java and is available for download⁴. It includes two main modules: the *selection* and the *abstraction* modules. The first

⁴ The MaGe website: <http://mapsforweb.wordpress.com>

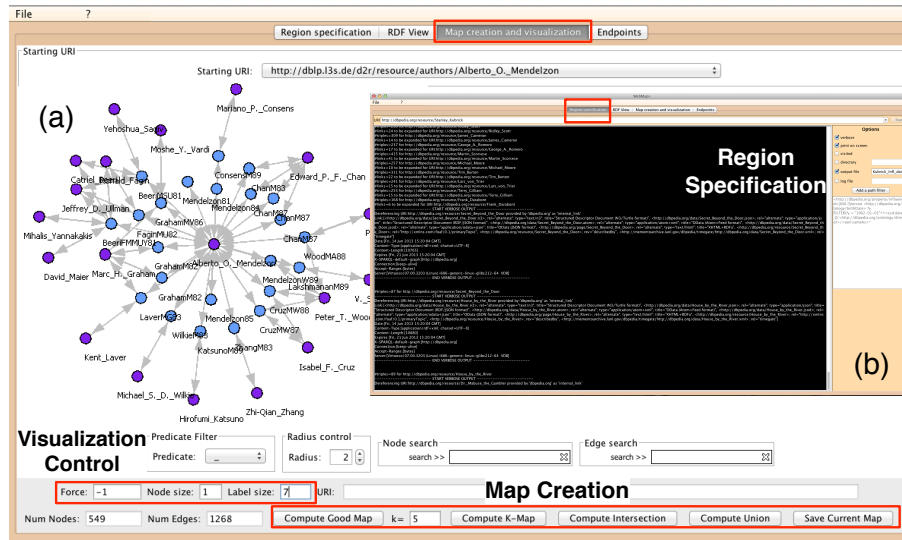


Fig. 2. The GUI of the MaGe tool.

one is responsible for the implementation of the NAUTiLOD language. In particular, given a seed URI and an expression, this module retrieves a Web region and a set of distinguished nodes. The second module, given the Web region and the set of distinguished nodes leverages the map framework to build maps. The decoupling between selection and abstraction enables to use the two functionalities also separately. MaGe is endowed with a GUI, which is shown in Fig. 2. It includes four main tabs. The first one (Fig. 2 (b)) is used to specify the region via a NAUTiLOD expression. The second and fourth display the region retrieved in RDF and the expression endpoints, respectively. The third tab (Fig. 2 (a)) deals with the creation of maps and their visualization. Both regions and maps can be saved in RDF allowing their storage, sharing, reuse and exchange. We now provide an example that we plan to show (along with others) in the demo. A video explaining how to use the tool is available at <http://youtu.be/BsvAiX3n968>.

Maps of Influence. An influence network is a graph where nodes are persons and edges represent influence relations. We leverage information taken from dbpedia.org and the property `dbpprop:influenced`.

Example 3 *Build a map of a region containing people that have influenced, or have been influenced by Stanley Kubrick (SK) up to distance 6. The distinguished nodes must be scientists.*

The region can be specified via the following NAUTiLOD expression. Here, the URI of SK in DBpedia (`dbpedia:Stanley_Kubrick`) is used as seed node:

$$\text{dbpprop:influenced}\langle 1-6\rangle[\text{ASK } \{?p \text{ rdf:type dbpedia:Scientist.}\}]$$

In the expression, the notation $\langle 1-6\rangle$ is a shorthand for the concatenation of (up to) six steps of the predicate `dbpprop:influenced`, while the ASK query in the test `[]` is used to filter the distinguished nodes (i.e., scientists).

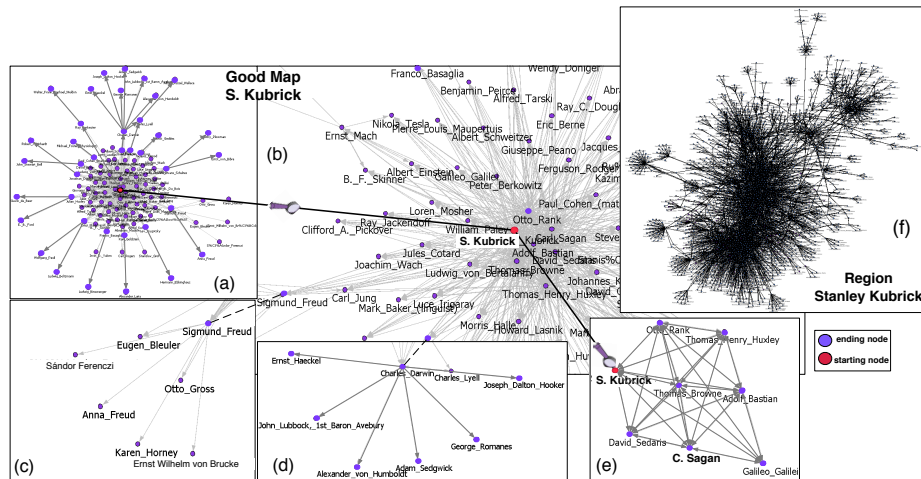


Fig. 3. Region (f) and good map (a) for SK with some zooms (b)-(e).

Fig. 3 (f) reports the region associated to the influence network of SK. The region contains 2981 nodes and 7893 edges. Indeed, it is very difficult to identify the distinguished nodes and more importantly connectivity among them and with the seed node. Fig. 3 (a)-(e) show the good map of this region (109 nodes; 2629 edges). The abstraction provided by the good map enables to identify the influence path, for instance, between SK and C. Segan (Fig. 3 (e)).

4 Conclusions

The availability of machine-processable information at a Web scale opens new perspectives toward the development of systems for the harnessing of knowledge on the Web. We contend that maps, key devices in helping human navigation in information spaces, are also meaningful on the Web space. They are useful navigation cues and powerful ways of conveying complex information via concise representations. Effectively, they play the role of *navigational charts*, that is, tools that provide users with abstractions of regions of information on the Web. We have implemented the MaGE system to generate maps. During the demo we will show maps in different domains including bibliographic networks.

References

1. V. Fionda, C. Gutierrez, and G. Pirrò. Extracting Relevant Subgraphs from Graph Navigation. In *ISWC (Posters & Demos)*, 2012.
2. V. Fionda, C. Gutierrez, and G. Pirrò. Semantic Navigation on the Web of Data: Specification of Routes, Web Fragments and Actions. In *WWW*, 2012.
3. V. Fionda, C. Gutierrez, and G. Pirrò. Knowledge Maps of Web Graphs. In *KR*, 2014.
4. T. Heath and C. Bizer. *Linked Data: Evolving the Web into a Global Data Space*. Morgan & Claypool, 2011.

Named Entity Recognition using FOX

René Speck and Axel-Cyrille Ngonga Ngomo

AKSW, Department of Computer Science, University of Leipzig, Germany
{speck, ngonga}@informatik.uni-leipzig.de

Abstract. Unstructured data still makes up an important portion of the Web. One key task towards transforming this unstructured data into structured data is named entity recognition. We demo FOX, the Federated knOwledge eXtraction framework, a highly accurate open-source framework that implements RESTful web services for named entity recognition. Our framework achieves a higher F-measure than state-of-the-art named entity recognition frameworks by combining the results of several approaches through ensemble learning. Moreover, it disambiguates and links named entities against DBpedia by relying on the AGDISTIS framework. As a result, FOX provides users with accurately disambiguated and linked named entities in several RDF serialization formats. We demonstrate the different interfaces implemented by FOX within use cases pertaining to extracting entities from news texts.

1 Introduction

The Semantic Web vision requires the data on the Web to be represented in a machine-readable format. Given that a significant percentage of the data available on the Web is unstructured, tools for transforming text into RDF are of central importance. In this demo paper, we present FOX, the federated knowledge extraction framework.¹ It integrates state-of-the-art named entity recognition (NER) frameworks by using ensemble learning (EL). By these means, FOX can achieve up to 95.23% F-measure where the best of the current state-of-the-art system (Stanford NER) achieves 91.68% F-measure. In this paper, we aim to demonstrate several of the features of FOX, including the large number of input and output formats it supports, different bindings with which FOX can be integrated into Java and Python code and the easy extension model underlying the framework. Our framework is already being used in several systems, including SCMS [5], ConTEXT [3] and IR frameworks [8]. The approach underlying FOX is presented in [7], which will be presented at the same conference. All features presented herein will be part of the demonstration.

2 Demonstration

The goal of the demonstration will be to show the whole of the FOX workflow from the gathering and preprocessing of input data to the generation of RDF data. In addition, we will show how to configure and train FOX after it has been enhanced with a

¹ FOX online demo:<http://fox-demo.aksw.org>

FOX project page:<http://fox.aksw.org>.

Source code, evaluation data and evaluation results:<http://github.com/AKSW/FOX>.

novel NER tool or EL algorithm. Further, we will present FOX's feedback RESTful service to improve the training and test datasets. In the demonstration, we also go over the Python² and Java bindings³ for an easy use of FOX's RESTful service within an application. At the end we will explain how to use the FOX Java interfaces to integrate future algorithms.

2.1 Workflow

The workflow underlying FOX consists of four main steps: (1) preprocessing of the unstructured input data, (2) recognizing the Named Entities (NE), (3) linking the NE to resources using AGDISTIS [9] and (4) converting the results to an RDF serialization format.

Preprocessing FOX allows users to use a URL, text with HTML tags or plain text as input data (see the top left part of Figure 1). The input can be carried out in a form (see the center of Figure 1) or via FOX's web service. In case of a URL, FOX sends a request to the given URL to receive the input data. Then, for all input formats, FOX removes HTML tags and detects sentences and tokens.

The screenshot shows the FOX online demo request form. At the top, there is a navigation bar with links: Home, Demo, Downloads, Doc, and Java Doc. Below the navigation bar, there is a 'Go Back' link. The form contains several fields and a 'Submit' button, each with a callout box explaining its function:

- Input Format:** A dropdown menu currently set to 'text/html'. Callout: 'Choose input format'.
- Extraction Type:** A dropdown menu currently set to 'ner'. Callout: 'Choose extraction task'.
- Input:** A text area containing the text: 'The foundation of the University of Leipzig in 1409 initiated the city's development into a centre of German law. The philosopher and mathematician Gottfried Wilhelm Leibniz was born in Leipzig in 1646, and attended the university from 1661-1666.' Callout: 'Input field'.
- Output Format:** A dropdown menu with 'JSON-LD' selected. A list of other options is visible: N-Triples, RDF/JSON, RDF/XML, Turtle, TriG, and N-Quads. Callout: 'Choose output format'.
- Examples:** A section listing the same output format options as above.
- Fox Light:** A dropdown menu currently set to 'OFF'. Callout: 'Choose NER approach'.
- Submit:** A yellow button with the text 'Submit'. Callout: 'Submit request'.

Fig. 1. Request form of the FOX online demo.

We will use text examples, URLs and text with HTML tags to show how FOX gathers or cleans them for the sake of entity recognition.

² <https://pypi.python.org/pypi/foxpy>

³ <https://github.com/renespeck/fox-java>

Entity Recognition Our approach relies on four state-of-the-art NER tools so far: (1) the Stanford Named Entity Recognizer (Stanford) [2], (2) the Illinois Named Entity Tagger (Illinois) [6], (3) the Ottawa Baseline Information Extraction (Balie) [4] and (4) the Apache OpenNLP Name Finder (OpenNLP) [1]. FOX allows using a particular NER approach which is integrated in it (see bottom right of Figure 1). To this end, FOX light has to be set to the absolute path to the class of the tool to use. If FOX light is off, then FOX utilizes these four NER tools in parallel and stores the received NEs for further processing. It maps the entity types of each of the NER tools to the classes `Location`, `Organization` and `Person`. Finally, the results of all tools are merged by using FOX’s EL layer as discussed in [7]. We will show the named entities recognized by FOX and contrast these with those recognized by the other tools. Moreover, we will show the runtime log that FOX generates to point to FOX’s scalability.

Entity Linking FOX makes use of AGDISTIS [9], an open-source named entity disambiguation framework able to link entities against every linked data knowledge base, to disambiguate entities and to link them against DBpedia. In contrast to lookup-based approaches, our framework can also detect resources that are not in DBpedia. In this case, these are assigned their own URIs. Moreover, FOX provides a Java interface and a configuration file for easy integration of other entity linking tools. We will show the messages that FOX generates and sends to AGDISTIS as well as the answers it receives and serializes.

Serialization Formats FOX is designed to support a large number of use cases. To this end, our framework can serialize its results into the following formats: JSON-LD⁴, N-Triples⁵, RDF/JSON⁶, RDF/XML⁷, Turtle⁸, TriG⁹, N-Quads¹⁰. FOX allows the user to choose between these formats (see bottom left part of Figure 1). We will show how the out of FOX looks like in the different formats and point to how they can be parsed.

3 Evaluation and Results

We performed a thorough evaluation of FOX by using five different datasets and comparing it with state-of-the-art NER frameworks (see Table 1). Our evaluation shows that FOX clearly outperforms the state of the art. The details of the complete evaluation are presented in [7]. The evaluation code and datasets are also available at FOX’s Github page, i.e., <http://github.com/AKSW/FOX>.

⁴ <http://www.w3.org/TR/json-ld>

⁵ <http://www.w3.org/TR/n-triples/>

⁶ <http://www.w3.org/TR/rdf-json>

⁷ <http://www.w3.org/TR/REC-rdf-syntax>

⁸ <http://www.w3.org/TR/turtle>

⁹ <http://www.w3.org/TR/trig>

¹⁰ <http://www.w3.org/TR/n-quads>

Table 1. Comparison of the F-measure of FOX with the included NER tools. Best results are marked in bold font.

	token-based					entity-based				
	News	News*	Web	Reuters	All	News	News*	Web	Reuters	All
FOX	92.73	95.23	68.81	87.55	90.99	90.70	93.09	63.36	81.98	90.28
Stanford	90.34	91.68	65.81	82.85	89.21	87.66	89.72	62.83	79.68	88.05
Illinois	80.20	84.95	64.44	85.35	79.54	76.71	83.34	54.25	83.74	76.25
OpenNLP	73.71	79.57	49.18	73.96	72.65	67.89	75.78	43.99	72.89	67.66
Balie	71.54	79.80	40.15	64.78	69.40	69.66	80.48	35.07	68.71	67.82

4 Conclusion

We will present FOX, a NER framework which relies on EL and demonstrate how it can be used. In future work, we will extend the number of tools integrated in FOX. Moreover, we will extend the tasks supported by the framework. In particular, we aim to integrate tagging, keyword extraction as well as relation extraction in the near future.

References

1. J Baldridge. The opennlp project, 2005.
2. Jenny Rose Finkel, Trond Grenager, and Christopher Manning. Incorporating non-local information into information extraction systems by gibbs sampling. In *ACL*, pages 363–370, 2005.
3. Ali Khalili, Sören Auer, and Axel-Cyrille Ngonga Ngomo. context – lightweight text analytics using linked data. In *11th Extended Semantic Web Conference (ESWC2014)*, 2014.
4. David Nadeau. Balie—baseline information extraction: Multilingual information extraction from text with machine learning and natural language techniques. Technical report, Technical report, University of Ottawa, 2005.
5. Axel-Cyrille Ngonga Ngomo, Norman Heino, Klaus Lyko, René Speck, and Martin Kaltenböck. SCMS - Semantifying Content Management Systems. In *Proceedings of the International Semantic Web Conference*, 2011.
6. Lev Ratinov and Dan Roth. Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, CoNLL '09, pages 147–155, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.
7. René Speck and Axel-Cyrille Ngonga Ngomo. Ensemble learning for named entity recognition. In *In Proceedings of the International Semantic Web Conference*, Lecture Notes in Computer Science, 2014.
8. Ricardo Usbeck. Combining linked data and statistical information retrieval. In *11th Extended Semantic Web Conference, PhD Symposium*. Springer, 2014.
9. Ricardo Usbeck, Axel-Cyrille Ngonga Ngomo, Sören Auer, Daniel Gerber, and Andreas Both. Agdistis - agnostic disambiguation of named entities using linked open data. In *Submitted to 12th International Semantic Web Conference, 21-25 October 2013, Sydney, Australia*, 2013.

A Linked Data Platform adapter for the Bugzilla issue tracker

Nandana Mihindukulasooriya,
Miguel Esteban-Gutiérrez, and Raúl García-Castro

Center for Open Middleware
Ontology Engineering Group, Escuela Técnica Superior de Ingenieros Informáticos
Universidad Politécnica de Madrid, Spain
{nmihindu,mesteban,rgarcia}@fi.upm.es

Abstract. The W3C Linked Data Platform (LDP) specification defines a standard HTTP-based protocol for read/write Linked Data and provides the basis for application integration using Linked Data. This paper presents an LDP adapter for the Bugzilla issue tracker and demonstrates how to use the LDP protocol to expose a traditional application as a read/write Linked Data application. This approach provides a flexible LDP adoption strategy with minimal changes to existing applications.

1 Introduction

The W3C Linked Data Platform (LDP) is an initiative to produce a standard protocol and a set of best practices for the development of read/write Linked Data applications [1]. The LDP protocol provides the basis for a novel paradigm of application integration using Linked Data¹ in which each application exposes its data as a set of Linked Data resources and the application state is driven following the REST design principles [2].

Some advantages of this approach over traditional SOAP-based web services include: (a) global identifiers for data that can be accessed using the Web infrastructure and typed links between data from different applications [3]; (b) the graph-based RDF data model that allows consuming and merging data from different sources without having to do complex structural transformations; and (c) explicit semantics of data expressed in RDF Schema or OWL ontologies which can be aligned and mapped to data models of other applications using techniques such as ontology matching.

This approach is more suitable when the integration is data-intensive and the traceability links between different applications are important. The Application Lifecycle Management (ALM) domain, in which heterogeneous tools are used in different phases of the software development lifecycle, provides a good use case for this approach. The ALM iStack project² has developed a prototype for

¹ <http://www.w3.org/DesignIssues/LinkedData.html>

² <https://sites.google.com/a/centeropenmiddleware.com/alm-istack/>

integrating ALM tools by using the LDP protocol and this paper presents an LDP adapter developed to LDP-enable the Bugzilla³ issue tracker.

2 An LDP adapter for Bugzilla

The three main alternatives for LDP-enabling an application are: (a) native support built into the application; (b) an application plugin; and (c) an LDP adapter. Providing native support requires modification to the application and not all applications allow extensions through plugins. As we have seen in the early stages of web services [4], adapters provide a more flexible mechanism to gradually adopting a technology while using the existing tools with minimum changes, and we have leveraged this approach.

An application is defined in terms of its data model and business logic. An LDP-enabled application exposes the data as Linked Data and allows to drive its business logic following the REST design principles. Thus to LDP-enable an application, its data model should be expressed in RDF by mapping it to a new ontology or by reusing existing vocabularies. In the Bugzilla adapter, the Bugzilla native data model is mapped to the ALM iStack ontology⁴. The adapter exposes the Bugzilla data as LDP resources by transforming the data between the ALM iStack ontology and the Bugzilla native model so that LDP clients can consume RDF data from Bugzilla as if it was a native LDP application.

The Bugzilla LDP adapter, which is a JavaEE web application, consists of three main layers: (a) *LDP layer*, (b) *transformation layer*, and (c) *application gateway layer*, as illustrated in Figure 1.

The **LDP layer** handles the LDP communications and exposes the Bugzilla data as LDP resources. This layer is built using the LDP4j framework⁵ which provides a middleware for the development of read/write Linked Data applications [5]. The concepts such as bugs, products, product versions, and users are mapped to LDP containers which list these entities and allow creating new entities. Each entity such as a bug, a product, or a user is mapped to an LDP resource with its own URI that can be used by clients to access them.

The **transformation layer** handles data validation and transformation. This includes extracting information from RDF data, validating them based on application restrictions, and mapping them to the Bugzilla model. The ALM iStack ontology is generic so that it can be used with other issue trackers (*e.g.*, JIRA⁶, Redmine⁷); thus there is an impedance mismatch between the ontology and the Bugzilla native model which is managed by the adapter.

The **application gateway layer** handles the communication with the Bugzilla instance using its XML-RPC remote interface. Because the Bugzilla bug tracker is also accessed using its web UI, the adapter synchronizes with the

³ <http://www.bugzilla.org/>

⁴ <http://delicias.dia.fi.upm.es/ontologies/alm-istack.owl>

⁵ <http://www.ldp4j.org/>

⁶ <https://www.atlassian.com/software/jira>

⁷ <http://www.redmine.org/>

Bugzilla instance based on user-defined policies. In addition there are several cross-cutting services such as configuration management, consistency, security, and synchronization which are utilized by multiple layers.

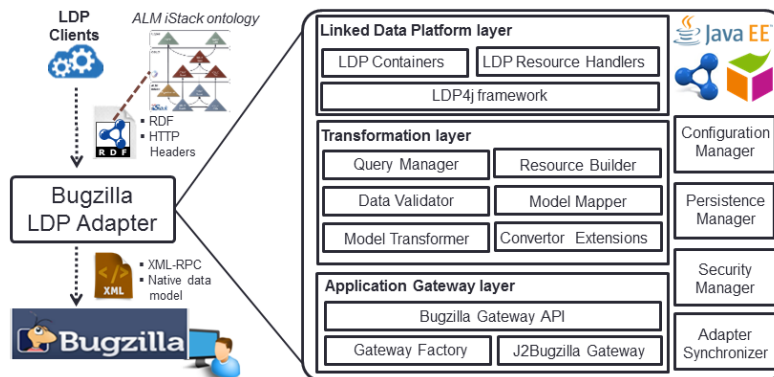


Fig. 1. High-level architecture of the Bugzilla adapter

3 Demonstration

This demonstration shows how LDP clients can use the adapter to access the Bugzilla bug tracker and to perform tasks such as discovering bugs reported against a product, modifying the status or the other properties of the bug, or creating new bugs (e.g., Fig. 2 shows a creation request and response).

```

POST /ita/containers/bugs HTTP/1.1
Host: localhost
Content-Type: text/turtle

@prefix ai:      <http://delicias.dia.fi.upm.es/ontologies/alm-istack#> .
...rest of the namespace declarations are omitted for brevity.
<> a ai:ClientDefect; 2
1 dcterms:creator <http://localhost/ita/ldp/resources/users/9> ;
3 dcterms:title "Bugzilla adapter build is broken"^^xsd:string;
4 dcterms:description "Bugzilla adapter build fails due to a test failure"^^xsd:string;
5 oslc_asset:relatedAsset <http://localhost/ita/ldp/resources/productversions/16> ;
6 ai:relatedIncident <https://bugzilla.mozilla.org/show_bug.cgi?id=730698> .

HTTP/1.1 201 Created
Location: http://localhost/ita/ldp/resources/bugs/12 7
Link: <http://www.w3.org/ns/ldp#Resource>; rel='type' 8
Content-Length: 0
    
```

Fig. 2. Creation of a new bug using the Bugzilla LDP adapter

For example, a continuous integration server in an integrated ALM setting encounters a build failure. Thus, the “integration server agent” (1) wants to report a defect (2) titled “Bugzilla adapter build is broken” (3) with description “Bugzilla adapter build fails due to a test failure” (4) for the “version 1.0

of the *Bugzilla Adapter*” product (5) that is related to the “*issue 730698*” in “<https://bugzilla.mozilla.org/>” (6). The LDP client converts this message to an LDP request according to the ALM iStack ontology as shown in Figure 2.

Once this request is received by the adapter, it extracts the necessary information, transforms it into the Bugzilla model using a mapping between the ontology and Bugzilla models, and creates a bug in the Bugzilla instance using its remote XML-RPC interface. Once created, the Bugzilla instance returns the identifier for the bug inside Bugzilla. Then, the adapter generates an URI for the bug and manages the mapping between the identifier given by the Bugzilla and the URI. Any information that does not fit into the Bugzilla model such as links to external applications is maintained in the adapter. Finally, the adapter returns the URI using the Location header (7) and lets the client know it is an LDP resource using the “type” link relation (8) according to the LDP protocol.

The LDP client or other external applications can access and link to the bug using the URI returned by the adapter. In addition, clients can modify the bug using the PUT operation with modified RDF data which then will be propagated to the Bugzilla instance following a similar process.

4 Conclusion

In this paper, we presented the Bugzilla LDP adapter and provided an overview of how to build adapters for LDP-enabling existing applications in order to use them as read/write Linked Data applications. With minimal changes to the existing application, the Bugzilla LDP adapter enables semantic integration of the Bugzilla tool with other LDP-enabled applications and makes possible to have typed links between application data.

Acknowledgments: The authors are partially supported by the ALM iStack project of the Center for Open Middleware.

References

1. Speicher, S., Arwe, J., Malhotra, A.: Linked Data Platform 1.0 (June 2014) W3C Candidate Recommendation, <http://www.w3.org/TR/ldp/>.
2. Mihindikulasooriya, N., García-Castro, R., Esteban-Gutiérrez, M.: Linked Data Platform as a novel approach for Enterprise Application Integration. In: Proceedings of the 4th International Workshop on Consuming Linked Data (COLD2013), Sydney, Australia (Oct 2013)
3. Heath, T., Bizer, C.: Linked Data: Evolving the Web into a Global Data Space. Synthesis lectures on the semantic web: theory and technology **1**(1) (2011) 1–136
4. Benatallah, B., Casati, F., Grigori, D., Nezhad, H.R.M., Toumani, F.: Developing adapters for web services integration. In: Advanced Information Systems Engineering, Springer (2005) 415–429
5. Esteban-Gutiérrez, M., Mihindikulasooriya, N., García-Castro, R.: LDP4j: A framework for the development of interoperable read-write Linked Data applications. In: Proceedings of the 1st ISWC Developers Workshop, Riva del Garda, Italy (Oct 2014)

LED: curated and crowdsourced Linked Data on Music Listening Experiences

Alessandro Adamou¹, Mathieu d'Aquin¹, Helen Barlow¹ and Simon Brown²

¹ The Open University, United Kingdom
{alessandro.adamou, mathieu.daquin, helen.barlow}@open.ac.uk

² Royal College of Music, United Kingdom
simon.brown@rcm.ac.uk

Abstract. We present the Listening Experience Database (LED), a structured knowledge base of accounts of listening to music in documented sources. LED aggregates scholarly and crowdsourced contributions and is heavily focused on data reuse. To that end, both the storage system and the governance model are natively implemented as Linked Data. Reuse of data from datasets such as the BNB and DBpedia is integrated with the data lifecycle since the entry phase, and several content management functionalities are implemented using semantic technologies. Imported data are enhanced through curation and specialisation with degrees of granularity not provided by the original datasets.

Keywords: Linked Data, Crowdsourcing, Digital Humanities, Data Workflow

1 Introduction

Most research on listening to music focuses on investigating associated cognitive processes or analysing its reception by critics or commercial indicators such as sales. There is only sporadic research on the cultural and aesthetic position of music among individuals and societies over the course of history. One obstacle to this kind of research is the sparsity of primary source evidence of listening to music. Should such evidence be compiled, we argue that the adoption of explicit structured semantics would help highlight the interactions of listeners with a range of musical repertoires, as well as the settings where music is performed.

With the **Listening Experience Database (LED)**¹, we aim at covering this ground. LED is the product of a Digital Humanities project focused on gathering documented evidence of listening to music across history and musical genres. It accepts contributions from research groups in humanities as well as the crowdsourcing community, however, the data management workflow is supervised to guarantee that minimum scholarly conventions are met. Being conceived with data reuse in mind, LED is natively implemented as Linked Data. All the operations in the data governance model manipulate triples within, and across, named RDF graphs that encode provenance schemes for users of the system.

¹ LED, online at <http://www.open.ac.uk/Arts/LED>

Several content management functionalities available in LED, such as content authoring, review, reconciliation and faceted search, incorporate Linked Data reuse. Reused datasets include DBpedia² and the British National Bibliography (BNB)³, with music-specific datasets currently under investigation. Reused data are also enhanced, as the LED datamodel is fine-grained and allows for describing portions of documents and excerpts, which are not modelled in the datasets at hand. LED therefore also aims at being a node by its own right in the Linked Data Cloud, providing unique content and contributing to existing data too. At the time of writing, the LED dataset stores about 1,000 listening experience records contributed by 25 users, half of whom being volunteers from the crowd.

2 Related work

A similar effort in aggregating structured data in primary evidence was already carried out for reading experiences [1], though the process was not data-driven and the resulting Linked Data were only marginally aligned. We also acknowledge a project being carried out, which gathers direct personal experiences of young users of the system, albeit with a minimal data structure⁴. We also drew inspiration from earlier accounts of using DBpedia for music, such as the *dbrec* recommender [3]. Crowdsourcing is also gaining the attention of the Semantic Web community, with very recent attempts at tackling data quality aspects [4].

3 The Listening Experience Database

We define a **listening experience** (LE) as a documented (i.e. with a quotable and citable source) engagement of an individual in an event where some piece of music is played. In terms of conceptual modelling, a LE is a subjective *event*, and one *document* describing it is the quoted evidence reported in the database.

The lifecycle of data in LED involves the roles of *contributor*, *consumer* and *gatekeeper*, and states called *draft*, *submitted*, *public* and *blacklisted*. Every artifact stored in the system exists in one or more of these states (except blacklisted ones, which exclude all other states), and a state determines if a user with a certain role can “see” an artifact or not. What these artifacts are, depends on the specific phases in the workflow, which are transitions between these states.

Authoring. Contributors populate the knowledge base by entering data on a LE and its associated entities. The entry forms are dynamic and provide suggestions and autocompletion data from LED and external datasets in real time (cf. Figure 1). Artifacts declared during this phase remain in a draft state, only to enter a submitted state once the contributor submits the LE to gatekeepers.

Review. Privileged users with the gatekeeper role review a submitted artifact and either promote it to the public state, or reject it for blacklisting, or demote it

² DBpedia, <http://dbpedia.org>

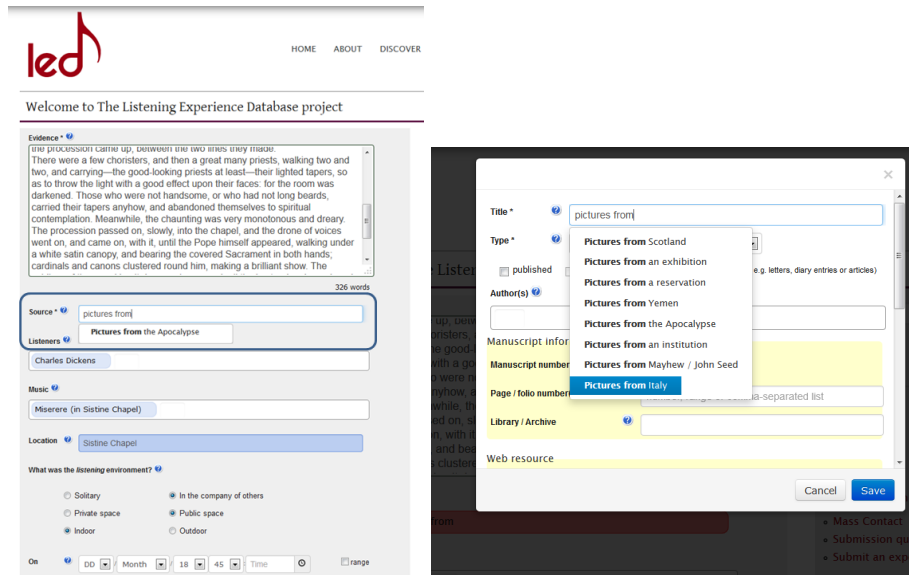
³ British National Bibliography, <http://bnb.data.bl.uk>

⁴ Experiencing Music, <http://experiencingmusic.com>

to draft again, which they can do by either taking over the artifact and amending its data themselves, or sending it back to the original contributor.

Reconciliation. Gatekeepers can align and merge duplicate artifacts that are found to match. They can compare candidate duplicates with other artifacts in LED and third-party data. This operation does not modify their state.

Faceted search. Consumers can navigate LE’s by filtering keyword search results by bespoke criteria which are not necessarily stored in LED, but also reused from third-party datasets. Only public artifacts contribute to searches.



(a) Listening experience submission. (b) Autocompletion from the BNB dataset.

Fig. 1: Example of data entry for “Pictures from Italy” by Charles Dickens.

With a native Linked Data implementation, we can immediately integrate reuse with every stage of the data lifecycle starting with data entry, and eliminate *a posteriori* revision and extraction phases from the workflow, thereby reducing the time-to-publish of our data and having them linked right from the beginning. Also, the named graph model of quad-stores can encode provenance information with the granularity of atomic statements [2], thus lending itself to fine-grained and complex trust management models.

To encode the above workflow entirely in RDF, we used the named graph paradigm in order to represent states and artifacts. Deciding on the scale of the latter was an issue: while we intended to give gatekeepers control on single RDF triples (or quads, from the named graph perspective), and to contributors a way to support the truth or falsehood of a triple, this can be complex and time-consuming. Therefore, artifacts are encapsulated into LE’s, musical works,

literary sources, agents (e.g. people, groups or organisations) and places: these are, for instance, the classes of artifacts that gatekeepers may want to review or reconcile. However, LE's remain the core artifacts of the system: only by creating or editing them can their associated artifacts be generated.

The LED knowledge base is partitioned into *data spaces*, each belonging to a user or role. Every contributor owns two RDF graphs, one for draft artifacts and one for submitted ones. Thus, we can keep track of which contributors support a fact by reusing it (e.g. `<Messiah_(oratorio) composer Georg_Frideric_Handel>`). There is a single graph for public artifacts, and one for blacklisted ones. Contributors have access to the graphs they own plus the public graph; gatekeepers can access every user's submitted graph and the public and blacklist graphs. State transitions are realised by parametric SPARQL queries that selectively move RDF triples across these graphs. Along with these data spaces there are rules that determine the visibility of triples to each user, depending on the content of their private graphs. In general, these rules assume contributors have greater confidence in the facts in their possession, and when missing, they should trust those provided by the community or other datasets.

4 Demonstration

The audience will be given a live demonstration of the LED system, but from the point of view of users with the privileged roles of *contributor* and *gatekeeper*. We will show the benefits of reusing data from indexed datasets during the entry phase, as well as the implementation of our governance model in Linked Data and its effects on the representation of a resource as seen by the general public or a specific user. Data reuse and enhancement will be demonstrated through a LE entry form to be auto-populated in real time and open to input by audience members. To demonstrate the governance model, we will run two distinct entries with shared data through the whole draft-submission-gatekeeping lifecycle. We will then show how differently the shared data and their RDF representations appear to each user, based on the trust and provenance policies in place.

References

1. Matthew Bradley. The Reading Experience Database. *Journal of Victorian Culture*, 15(1):151–153, 2010.
2. Jeremy J. Carroll, Christian Bizer, Patrick J. Hayes, and Patrick Stickler. Named graphs, provenance and trust. In Allan Ellis and Tatsuya Hagino, editors, *WWW*, pages 613–622. ACM, 2005.
3. Alexandre Passant. dbrec - music recommendations using DBpedia. In Peter F. Patel-Schneider, Yue Pan, Pascal Hitzler, Peter Mika, Lei Zhang, Jeff Z. Pan, Ian Horrocks, and Birte Glimm, editors, *International Semantic Web Conference (2)*, volume 6497 of *Lecture Notes in Computer Science*, pages 209–224. Springer, 2010.
4. Elena Simperl, Maribel Acosta, and Barry Norton. A semantically enabled architecture for crowdsourced linked data management. In Ricardo A. Baeza-Yates, Stefano Ceri, Piero Fraternali, and Fausto Giunchiglia, editors, *CrowdSearch*, volume 842 of *CEUR Workshop Proceedings*, pages 9–14. CEUR-WS.org, 2012.

WhatTheySaid: Enriching UK Parliament Debates with Semantic Web

Yunjia Li, Chaohai Ding, and Mike Wald

School of Electronics and Computer Science,
University of Southampton, UK
{y12, cd8e10, mw}@ecs.soton.ac.uk

Abstract. To improve the transparency of politics, the UK Parliament Debate archives have been published online for a long time. However there is still a lack of efficient way to deeply analysis the debate data. WhatTheySaid is an initiative to solve this problem by applying natural language processing and semantic Web technologies to enrich UK Parliament Debate archives and publish them as linked data. It also provides various data visualisations for users to compare debates over years.

Keywords: linked data, parliamentary debate, semantic web

1 Introduction

The publicity of UK Parliament Debate, such as BBC Parliament¹ have exert tremendous influence on the transparency of politics in the UK. Political figures need to be responsible for what they have said in the debates as they are monitored by the public. However, it is still difficult currently to automatically analyse the debate archives to find the answer to questions such as: how the debates across months or even years are related to each other. For this purpose, we have developed WhatTheySaid² (WTS), which uses semantic Web and natural language processing (NLP) technologies to automatically enrich the UK Parliament debates and categorize them for searching, visualisation and comparison.

In UK, there are already applications, such as TheyWorkForYou³, to provide extended functions for users to search debates and view the performances of each Member of Parliament (MP), such as the voting history and recent appearances. The semantic Web approach is also applied in Polimedia [1] as a way to model Dutch Parliament debates and enrich them with named entities and external links to different media. In this demo, we refer to the data sources and the methodologies provided by those previous work and build more advanced features to fulfill the following requirements: (R1) Calculate the similarities between debates so that users can easily navigate through similar debates; (R2) Categorise debates into different topics and extract the key statements, so that

¹ http://www.bbc.co.uk/democracylive/bbc_parliament/

² <http://whattheysaid.org.uk>

³ <http://theyworkforyou.com>

users can easily spot the statements that are contradict to each other; (R3) Based on R2, link the debates to a fragment of debate video archive, so that users can watch the video fragment as the proof of the statement; (R4) Analyse the speeches of a particular MP and see how the sentiment is changing over time.

To demo the implementation of the requirements above, we have taken the UK House of Common debate data in 2013 from TheyWorkForYou as the sample dataset, and the following sections will go through the system.

2 Semantic Model of UK Parliament Debate

The WTS ontology⁴ models UK Parliament debate structure and involved agents. This ontology reuses some vocabularies such as FOAF⁵ and Ontology for Media Resource⁶. When designing this ontology, we have firstly referred to the data structure of TheyWorkForYou, where one debate is identified by a Heading and a Heading contains one or more Speeches. We have also added several attributes to Speech, such as sentimental score, primary topic, summarise text and related media fragment in order to save the data required to implement R2, R3 and R4 in Section 1.

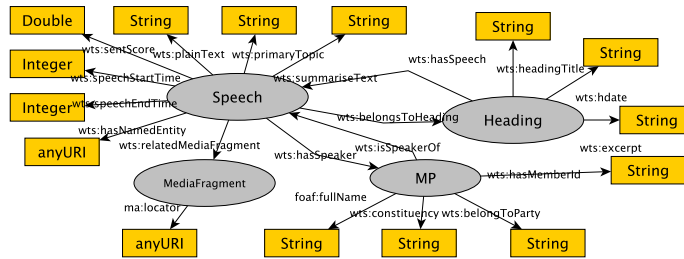


Fig. 1. WhatTheySaid Ontology

3 System Design and Walk-through

Figure 2 shows the architecture of WTS application. Our major data sources are the debate information from TheyWorkForYou, including debate date, speakers, headings, the text of speeches in each debate, etc., and the debate video with automatic transcripts provided by BBC Parliament archive. Then we use AlchemyAPI⁷ to proceed sentimental analysis on each speech in the debates so that

⁴ <http://www.whattheysaid.org.uk/ontology/v1/whattheysaid.owl>

⁵ <http://www.foaf-project.org>

⁶ <http://www.w3.org/TR/mediaont-10/>

⁷ <http://www.alchemyapi.com/>

each speech made by a speaker will be allocated with a score between 1.0 (positive) and -1.0 (negative). For speeches with more than 1000 characters, we also carry out topic detection and text summarisation using AlchemyAPI.

To link the debates to each other, we apply TF-IDF [3] algorithm to calculate the similarity scores between each two debates. We firstly merge the plain text of all the speeches in a debate into one big debate document d . Then, given a debate document collection D and $d \in D$, a word w , we calculate the weighting of each document W_d :

$$W_d = f_{w,d} * \log(|D|/f_{w,D}) \quad (1)$$

where $f_{w,d}$ equals the number of times w appears in d , $|D|$ is the size of corpus, and $f_{w,D}$ is the number of documents in which w appears in D [3]. In information retrieval, the Vector Space Model (VSM) represents each document in a collection as a point in a space and the semantically similarity of words is depended on the space distance of related points [4]. When the W_d is calculated for each document, we use cosine similarity⁸ for the vector space to come up with the similarity score between any two debate documents. On the user interface, every time a debate document is viewed, we will list the top ten debates that similar to this debate, so that users can easily navigate through similar debates.

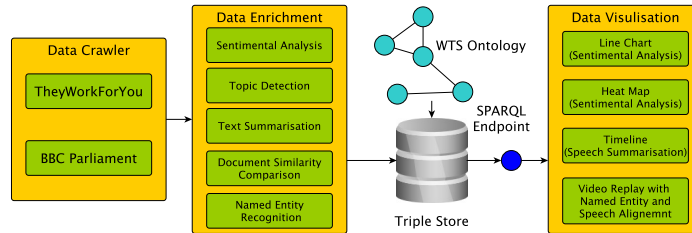


Fig. 2. WhatTheySaid Architecture Diagram

For named entity recognition, we use DBpedia Spotlight⁹ to extract named entities and interlink those concepts to the speeches, where they are mentioned. All the enrichment information are saved in a triple store implemented by rdfstore-js¹⁰, which also exposes a SPARQL Endpoint data querying and visualisation. For the whole 2013 year’s debate, we have collected 68968 speeches and more than 400K named entities (with duplication) have been recognised. Using the model defined in Figure 1, we have generated more than 1.2 million triples.

⁸ http://en.wikipedia.org/wiki/Cosine_similarity

⁹ <https://github.com/dbpedia-spotlight>

¹⁰ <https://github.com/antoniogarrote/rdfstore-js>

We visualise the enriched debate data in various ways. Firstly, we use both heat map and line chart to visualise the sentiment scores of speeches for each MP on yearly (see Figure 3(a)) and monthly basis respectively. We also provide a timeline visualisation (Figure 3(b)) for the statements in different topics made by a certain MP. To implement R3, we have referred to the previous work [2] and designed a replay page with the transcript and named entities aligned with the fragments of debate video¹¹. The full demo is available online¹² and the RDF dataset is published for download¹³. We are planning to expand the application with more debates from early years, so that debates across years can be interlinked and enriched for analysis.

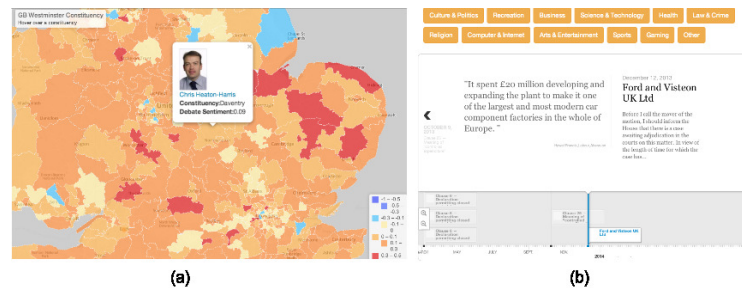


Fig. 3. WhatTheySaid Data Visualisation

4 Acknowledgement

This mini-project is funded by the EPSRC Semantic Media Network. We also would like to thank Yves Raimond from BBC and Sebastian Riedel from UCL for the support of this mini-project.

References

1. Juric, D., Hollink, L., Houben, G.J.: Bringing parliamentary debates to the semantic web. Detection, Representation, and Exploitation of Events in the Semantic Web (2012)
2. Li, Y., Rizzo, G., Troncy, R., Wald, M., Wills, G.: Creating enriched youtube media fragments with nerd using timed-text (2012)
3. Ramos, J.: Using tf-idf to determine word relevance in document queries. In: Proceedings of the First Instructional Conference on Machine Learning (2003)
4. Turney, P.D., Pantel, P., et al.: From frequency to meaning: Vector space models of semantics. Journal of artificial intelligence research 37(1), 141–188 (2010)

¹¹ Due to copyright issues, we cannot make the debate video publicly available.

¹² <http://whattheysaid.org.uk>

¹³ <http://whattheysaid.org.uk/download/wtstrip1e.ttl>

Multilingual Disambiguation of Named Entities Using Linked Data

Ricardo Usbeck^{♠◇}, Axel-Cyrille Ngonga Ngomo[◇], Wencan Luo[♡], and Lars
Wesemann[♠]

◇ University of Leipzig, Germany ,

♠ R & D, Unister GmbH, Leipzig, Germany,

♡ University of Pittsburgh, United States of America

email: {usbeck|ngonga}@informatik.uni-leipzig.de

Abstract. One key step towards extracting structured data from unstructured data sources is the disambiguation of entities. With AGDISTIS, we provide a time-efficient, state-of-the-art, knowledge-base-agnostic and multilingual framework for the disambiguation of RDF resources. The aim of this demo is to present the English, German and Chinese version of our framework based on DBpedia. We show the results of the framework on texts pertaining to manifold domains including news, sports, automobiles and e-commerce. We also summarize the results of the evaluation of AGDISTIS on several languages.

1 Introduction

A significant portion of the information on the Web is still only available in textual format. Addressing this information gap between the Document Web and the Data Web requires amongst others the extraction of entities and relations between these entities from text. One key step during this processing is the disambiguation of entities (also known as entity linking). The AGDISTIS framework [7] (which will also be presented at this conference) addresses two of the major drawbacks of current entity linking frameworks [1,2,3]: time complexity and accuracy. With AGDISTIS, we have developed a framework that achieves polynomial time complexity and outperforms the state of the art w.r.t. accuracy. The framework is knowledge-base-agnostic (i.e., it can be deployed on any knowledge base) and is also language-independent. In this demo, we will present AGDISTIS deployed on three different languages (English, German and Chinese) and three different knowledge bases (DBpedia, the German DBpedia and the Chinese DBpedia). To the best of our knowledge, we therewith provide the first Chinese instantiation of entity linking to DBpedia. We will also demonstrate the AGDISTIS web services endpoints for German, English and Chinese disambiguation and show how data can be sent to the endpoints. Moreover, the output format of AGDISTIS will be explained. An online version of the demo is available at <http://agdistis.aksw.org/demo>.

2 Demonstration

Within our demonstration, we aim to show how AGDISTIS can be used by non-expert as well as expert users. For non-experts, we provide a graphical user interface (GUI). Experts can choose to use the REST interfaces provided by the tool or use a Java snippet to call the REST interface. The whole of this functionality, which will be described in more details in the following sections, will also be demonstrated at the conference.

2.1 AGDISTIS for non-expert users

A screenshot of the AGDISTIS GUI is shown in Figure 1. This GUI supports the following workflow.

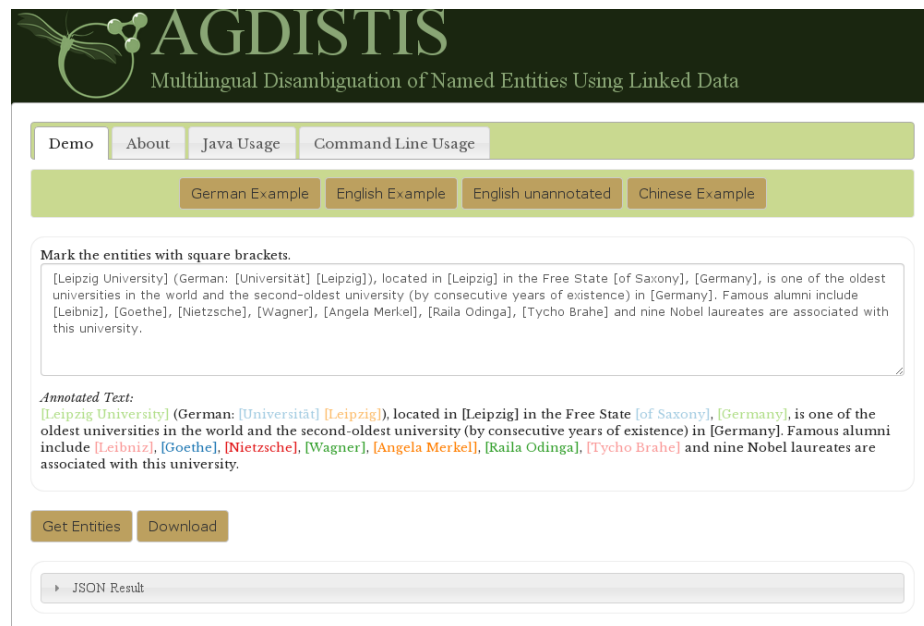


Fig. 1: Screenshot of the demo with an English example which is already annotated.

Entity Recognition After typing or pasting text into the input field, users can choose between either annotating the entities manually or having the entities detected automatically. In the first case, the labels of the entities are to be marked by using square brackets (see central panel of Figure 1). In the case of an automatic annotation, we send the text to the FOX framework, which has been shown to outperform the state of the art in [6]. We will demonstrate this feature by using both manually pre-annotated text and text without annotations

in our examples (see upper bar of Figure 1). Moreover, we will allow the crowd to enter arbitrary texts that pertain to their domain of interest.

Automatic Language Detection Once the user has set which entities are to be disambiguated, the marked-up text is sent to the language detection module based on [5]. We chose this library because it is both precise (> 99% precision) and time-efficient. If the input is detected to belong to one of the languages we support (i.e., German, Chinese, English), then we forward the input to a dedicated AGDISTIS instance for this given language. In all other cases, an error message is shown to the user, pointing towards the language at hand not being supported. The main advantage of this approach is that the user does not need to select the language in which the text is explicated manually, thus leading to an improved user experience. We will demonstrate this feature by entering text in different languages (German, English, French, Chinese, etc.) and presenting the output of the framework for each of these test cases.

Entity Linking This is the most important step of the whole workflow. The annotated text is forwarded to the corresponding language-specific deployment of AGDISTIS, of which each relies on a language-specific version of DBpedia 3.9. The approach underlying AGDISTIS [7] is language-independent and combines breadth-first search and the well-known HITS algorithm. In addition, string similarity measures and label expansion heuristics are used to account for typos and morphological variations in naming. Moreover, Wikipedia-specific surface forms for resources can be used.

Output Within the demo the annotated text is shown below the input field where disambiguated entities are colored to highlight them. While hovering a highlighted entity the disambiguated URI is shown. We will demonstrate the output of the entity linking by using the examples shown in the upper part of Figure 1. The output of the system will be shown both in a HTML version and made available as a download in JSON. Moreover, we will allow interested participants to enter their own examples and view the output of the tool.

2.2 AGDISTIS for expert users

To support different languages we set up a REST URI for each of the language versions. Each of these endpoints understands two mandatory parameters: (1) `text` which is an UTF-8 and URL encoded string with entities annotated with XML-tag `<entity>` and (2) `type='agdistis'` to disambiguate with the AGDISTIS algorithm. In the future, several wrappers will be implemented to use different entity linking algorithms for comparison. Following, a CURL¹ snippet shows how to address the web service, see also <http://agdistis.aksw.org>:

```
curl --data-urlencode "text='<entity>Barack Obama</entity> arrives
in <entity>Washington, D.C.</entity>.'" -d type='agdistis'
{AGDISTIS URL}/AGDISTIS
```

¹ <http://curl.haxx.se/>

3 Evaluation

English and German Evaluation. AGDISTIS has been evaluated on 8 different datasets from diverse domains such as news, sports or business reports. For English datasets AGDISTIS is able to outperform the currently best disambiguation framework, TagMe2, on three out of four datasets by up to 29.5% F-measure. Considering the only German dataset available for named entity disambiguation, i.e., `news.de` [4], we are able to outperform the only competitor DBpedia Spotlight by 3% F-measure.

Chinese Evaluation. We evaluated the Chinese version of AGDISTIS within a question answering setting. To this end, we used the multilingual benchmark provided in QALD-4². Since the Chinese language is not supported, we extended the QALD-4 benchmark by translating the English questions to Chinese and inserted the named entity links manually. The accuracies achieved by AGDISTIS for the train and test datasets are 65% and 70% respectively.

4 Conclusion

We presented the demo of AGDISTIS for three different languages on three different DBpedia-based knowledge bases. In future work, we aim to create a single-server multilingual version of the framework that will intrinsically support several languages at the same time. To this end, we will use a graph merging algorithm to combine the different versions of DBpedia to a single graph. The disambiguation steps will then be carried out on this unique graph.

Acknowledgments This work has been supported by the ESF and the Free State of Saxony and the FP7 project GeoKnow (GA No. 318159).



References

1. Paolo Ferragina and Ugo Scaiella. Fast and accurate annotation of short texts with wikipedia pages. *IEEE software*, 29(1), 2012.
2. Pablo N. Mendes, Max Jakob, Andres Garcia-Silva, and Christian Bizer. Dbpedia spotlight: Shedding light on the web of documents. In *Proceedings of the 7th International Conference on Semantic Systems (I-Semantics)*, 2011.
3. Andrea Moro, Alessandro Raganato, and Roberto Navigli. Entity linking meets word sense disambiguation: a unified approach. *TACL*, 2:231–244, 2014.
4. Michael Röder, Ricardo Usbeck, Sebastian Hellmann, Daniel Gerber, and Andreas Both. N3 - a collection of datasets for named entity recognition and disambiguation in the nlp interchange format. In *LREC*, 2014.
5. Nakatani Shuyo. Language detection library for java, 2010.
6. René Speck and Ngonga Ngomo. Ensemble learning for named entity recognition. In *International Semantic Web Conference*. 2014.
7. Ricardo Usbeck, Axel-Cyrille Ngonga Ngomo, Sören Auer, Daniel Gerber, and Andreas Both. Agdistis - agnostic disambiguation of named entities using linked open data. In *International Semantic Web Conference*. 2014.

² <http://greententacle.techfak.uni-bielefeld.de/~cunger/qald>

The Wikipedia Bitaxonomy Explorer

Tiziano Flati and Roberto Navigli

Dipartimento di Informatica
Sapienza Università di Roma

Abstract. We present WiBi Explorer, a new Web application developed in our laboratory for visualizing and exploring the bitaxonomy of Wikipedia, that is, a taxonomy over Wikipedia articles aligned to a taxonomy over Wikipedia categories. The application also enables users to explore and convert the taxonomic information into RDF format. The system is publicly accessible at wibitaxonomy.org and all the data is freely downloadable and released under a CC BY-NC-SA 3.0 license.

1 Introduction

Knowledge modeling is a long-standing problem which has been addressed in a variety of ways (see [8] for a survey). If we leave aside knowledge-lean taxonomy learning approaches [9], a typical and widespread model consists of knowledge resources and multilingual dictionaries which provide concepts and relationships between concepts. The scenario is characterized by two types of resources: those, such as BabelNet [6], which provide general untyped relationships, and those, such as DBpedia [1], in which edges model arbitrarily labelled predicates over concepts (e.g., *dbpedia-owl:birthPlace*).

In neither of these resource types, however, is any explicit attention paid to hypernymy as a distinct relation type. Instead, hypernymy has been proven to be a relevant relation type capable of ameliorating systems in several hard tasks in Natural Language Processing [2, 7]. Indeed, even restricting to Wikipedia, no high-quality, large-scale taxonomy is yet available, which exhibits high coverage for both Wikipedia pages and categories.

WiBi [4] is a project set up with the specific aim of providing hypernymy relations over Wikipedia and our tests confirm it as the best current resource for taxonomizing both Wikipedia pages and categories in a joint fashion with state-of-the-art results. Here we present a Web application for visualizing and exploring our bitaxonomy of Wikipedia. The interface also offers a customization of the “view” and allows the export of data into RDF, in line with today’s Semantic Web trend.

2 The Wikipedia Bitaxonomy

WiBi [4] is an approach which aims at building a bitaxonomy of Wikipedia, that is, automatically extracting two taxonomies, one for Wikipedia pages and one for Wikipedia categories, aligned to one another.

The bitaxonomy is built thanks to a three-phase approach that i) first builds a taxonomy for the Wikipedia pages, then ii) leverages this partial information to iteratively infer new hypernymy relations over Wikipedia categories while at the same time increasing the page taxonomy, and finally iii) refines the obtained category taxonomy by means of three ad-hoc heuristics that cope with structural problems affecting some categories. As a result, a bitaxonomy is obtained where each element - either page or category - is associated with one or more hypernyms and where elements of one taxonomy are aligned (i.e. linked) to elements of the other taxonomy. In order to transfer hypernymy knowledge from either one of the two Wikipedia sides to the other side, the whole process remarkably, and as a key feature, exploits categorization edges (here called *cross-edges*) manually provided by Wikipedians, which connect any page on one side to its categories on the other side and vice versa. Extensive comparison has been carried out on two datasets of 1,000 pages and categories each, against all the available knowledge resources, including MENTA, DBpedia, YAGO, WikiTaxonomy and WikiNet (for an extensive survey, see [5]). Results show that WiBi surpasses all competitors not only in terms of quality, with the highest precision and recall, but also in terms of coverage and specificity.

3 The demo interface

Here we present a Web-based visual explorer for displaying the two aligned taxonomies of WiBi, centered on any given Wikipedia item of interest chosen by the user. The interface easily integrates search facilities with customization tools which personalize the experience from a user's point of view.

The home page. An excerpt of the interface's home page is shown in Fig. 1(a). As can be seen, this page has been kept very clean with as few elements as possible. On the top of the page a navigation bar contains links to i) the *about* page, which contains release information about the website content, ii) a *download* area, where it is possible to obtain the data underlying the interface and iii) the *search* page, which represents the core contribution of this work.

The search page mainly contains a text area in which the user is requested to input her query of interest, additionally opting for searching through either the page inventory, the category inventory or both, thanks to dedicated radio buttons. After the query is sent, the search engine tries to match the input text against the whole database of Wikipedia pages (or categories) and, if a match is found, the engine displays the final result to the user. Otherwise, the query is interpreted as a lemma and the user is returned with the (possible) list of all Wikipedia pages/categories whose lemma matches against the query.

The result page. Starting from the Wikipedia element provided by the user, the objective of the result page is to show a relevant excerpt of the bitaxonomy, that is, the nearest (or more relevant) nodes connected to it, drawn from both of the two taxonomies. To do this, WiBi Explorer performs a series of steps:

1. Start a DFS of maximum length δ_1 from the given element p of a taxonomy. As a result, a subgraph $ST_1 = (SV_1, SE_1)$ is obtained;

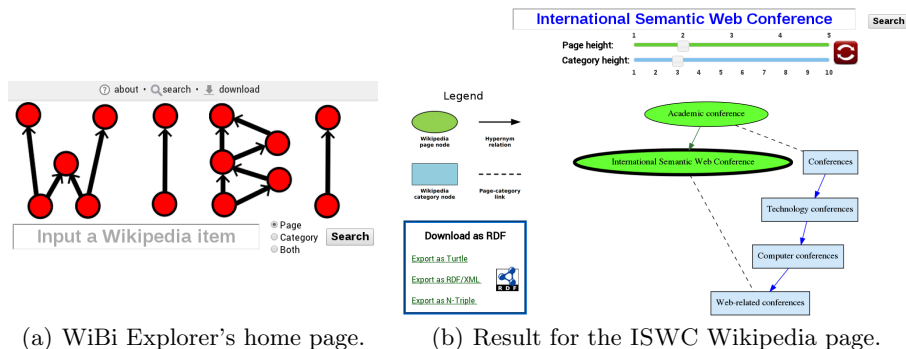


Fig. 1. The Wikipedia Bitaxonomy Explorer overview.

2. Collect all the nodes $\pi(p)$ belonging to the other taxonomy (i.e, those whose cross-edges are incident to p). Start a DFS of maximum length δ_2 from each element in $\pi(p)$. As a result, a subgraph $ST_2 = (SV_2, SE_2)$ is obtained;
3. Display ST_1 and ST_2 , as well as all the possible cross-edges linking nodes of the two subgraphs. Prune out low-connected nodes from the displayed bitaxonomy.

As a result, the interface displays a meaningful excerpt of the two taxonomies, centered on the issued query. The result for the Wikipedia page INTERNATIONAL SEMANTIC WEB CONFERENCE is shown in Fig. 1(b).

Customization of the view Since a user might be interested in a more general view of the bitaxonomy, two additional sliders are provided to the user in order to manually adjust the two maximum depths δ_1 and δ_2 (see Fig. 1(b) on top). Moreover, the interface provides the user with the capability to click on nodes and interactively explore different parts of the taxonomy. The application thus acts as a dynamic explorer that enables users to navigate through the structure of the bitaxonomy and discover new relations as the visit proceeds.

4 Converting data to RDF

Interestingly, data can also be exported in RDF format, in line with recent work on (linguistic) linked open data and the Semantic Web [3]. To this end, the explorer is backed by the Apache Jena framework (<https://jena.apache.org/>) and thus also integrates a single-click functionality that seamlessly converts the displayed data into RDF format. The user can opt for Turtle, RDF/XML or N-Triple format (see blue box in Fig. 1(b), bottom left). An excerpt of a view of the bitaxonomy converted into RDF for the query ISWC is shown in Fig. 2. As can be seen, several namespaces have been used: WiBi specific entities encode Wikipedia items, while standard SKOS's subsumption relations (*skos:narrower* and *skos:broader*) encode is-a relations.

5 Conclusions

We have proposed the Wikipedia Bitaxonomy Explorer, a new, flexible and extensible Web interface that allows the navigation of the recently created Wikipedia

```

@prefix wibi: <http://wibitaxonomy.org/> .
@prefix wibi-model: <http://wibitaxonomy.org/model/wibi#> .
@prefix skos: <http://www.w3.org/2004/02/skos/core#> .

wibi:International_Semantic_Web_Conference a skos:Concept;
  wibi-model:hasWikipediaCategory <http://wibitaxonomy.org/Category:Web-related_conferences> ;
  skos:broader wibi:Academic_conference .

<http://wibitaxonomy.org/Category:Conferences> a skos:Concept;
  wibi-model:hasWikipediaPage wibi:Academic_conference ;
  skos:narrower <http://wibitaxonomy.org/Category:Technology_conferences> .

```

Fig. 2. RDF excerpt of the taxonomy view for the ISWC Wikipedia page.

Bitaxonomy [4]. In addition to default settings, several parameters concerning the general appearance of the results can also be customized according to the user’s preferences. The demo is available at wibitaxonomy.org, it is seamlessly integrated into the BabelNet interface (<http://babelnet.org/>) and the data is freely downloadable under a CC BY-NC-SA 3.0 license.

Acknowledgments



The authors gratefully acknowledge the support of the
ERC Starting Grant MultiJEDI No. 259234.



The authors also acknowledge support from the LIDER project (No. 610782), a Coordination and Support Action funded by the EC under FP7.

References

1. Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., Hellmann, S.: DBpedia - a crystallization point for the Web of Data. *Web Semantics* 7(3), 154–165 (2009)
2. Cui, H., Kan, M.Y., Chua, T.S.: Soft Pattern Matching Models for Definitional Question Answering. *ACM Transactions on Information Systems* 25(2) (2007)
3. Ehrmann, M., Ceconi, F., Vannella, D., McCrae, J.P., Cimiano, P., Navigli, R.: Representing Multilingual Data as Linked Data: the Case of BabelNet 2.0. In: *Proc. of LREC 2014*. pp. 401–408. Reykjavik, Iceland
4. Flati, T., Vannella, D., Pasini, T., Navigli, R.: Two Is Bigger (and Better) Than One: the Wikipedia Bitaxonomy Project. In: *Proc. of ACL 2014*. pp. 945–955. Baltimore, Maryland
5. Hovy, E.H., Navigli, R., Ponzetto, S.P.: Collaboratively built semi-structured content and Artificial Intelligence: The story so far. *Artificial Intelligence* 194, 2–27 (2013)
6. Navigli, R., Ponzetto, S.P.: BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial Intelligence* 193, 217–250 (2012)
7. Snow, R., Jurafsky, D., Ng, A.: Semantic taxonomy induction from heterogeneous evidence. In: *Proc. of the COLING-ACL 2006*. pp. 801–808
8. Van Harmelen, F., Lifschitz, V., Porter, B.: *Handbook of knowledge representation*, vol. 1. Elsevier (2008)
9. Velardi, P., Faralli, S., Navigli, R.: OntoLearn Reloaded: A graph-based algorithm for taxonomy induction. *Computational Linguistics* 39(3), 665–707 (2013)

Enhancing Web intelligence with the content of online video fragments

Lyndon Nixon¹, Matthias Bauer¹, and Arno Scharl^{1,2}

¹ MODUL University, Vienna, Austria
lyndon.nixon@modul.ac.at

² webLyzard technology gmbh, Vienna, Austria
scharl@webLyzard.com

Abstract. This demo will show work to enhance a Web intelligence platform which crawls and analyses online news and social media content about climate change topics to uncover sentiment and opinions around those topics over time to also incorporate the content within non-textual media, in our case YouTube videos. YouTube contains a lot of organisational and individual opinion about climate change which currently can not be taken into account by the platforms sentiment and opinion mining technology. We describe the approach taken to extract and include the content of YouTube videos and why we believe this can lead to improved Web intelligence capabilities.

1 Introduction

Web intelligence refers to use of technologies to extract knowledge from data on the Web, in particular, the learning of how opinions or sentiment towards specific topics or concepts change over time by analyzing the content of time-specific Web data such as activity streams on social media platforms, news stories from trustworthy newsfeeds and press releases from relevant organisations. The webLyzard Web intelligence platform ³, which has been in development since many years of university R&D, collects and analyzes big data repositories gathered from a range of electronic sources and uses state-of-the-art Web intelligence tools to reveal flows of relevant information between stakeholders through trend analyses, benchmarks and customized reports. One key domain to which the platform has been applied is climate change [1], and the insights provided by webLyzard are being used by the NOAA in the US to inform their online communication.

A public demonstrator “Media Watch on Climate Change” is available at <http://www.ecoresearch.net/climate>. This demonstrator analyses news articles from British and US news sources, social media and the (RSS) PR feeds of Fortune 1000 companies. For example, for any searched term, the frequency of mentions of the term over a selected time period can be seen, the level of positive or negative sentiment expressed around that term, and extent of disagreement across sources. The individual sources can be explored and the content displayed,

³ <http://webLyzard.com>

e.g. text of a news article or company press release, or in social media a tweet or a YouTube video. While mentions of terms within the textual sources is being used to provide deep analytics on frequency, sentiment and disagreement over time for that term, any use of the same term within the YouTube videos which are crawled continually by the platform is disregarded, as details of the video content is not available to the internal analytics tools of the platform.

In the MediaMixer project <http://mediamixer.eu>, whose goal was to promote innovative media technology supporting fragments and semantics to industry use cases [2], a collaboration with webLyzard led to a prototype platform where the content of fragments of crawled YouTube videos could be exposed to the platforms analytics capabilities and hence video fragments could be made available to platform search and data visualisation components. This demo paper describes the approach taken and the resulting implementation (under the name videoLyzard)⁴ as well as how we believe this work can help lead to improved Web intelligence capabilities for stakeholders such as the NOAA.

2 Technical process and workflow

A new server-side processing pipeline has been created which takes a batch of YouTube video URLs from the webLyzard crawl component and processes them by getting transcripts of each YouTube video, performing Named Entity Recognition (NER) over the transcripts, and generating on that basis an annotation for each YouTube video which identifies the temporal fragments of the video and the entities which occur in each fragment. These annotations are exported back into the webLyzard platform and on that basis access to video fragments matching search terms is made possible. videoLyzard makes use of different semantic and multimedia technologies to:

- split videos into distinct temporal fragments (generally corresponding with the sentence level in the accompanying speech), using the Media Fragment URI specification to refer to the fragments by URL ⁵
- extract distinct entities from the textual transcript of the video, using the aggregation of Named Entity Recognition, or NER, Web services called NERD ⁶, and attaching entity annotations to a temporal fragment of the video
- normalizing entity identifications to DBpedia URIs, thus using Linked Data to provide a Web-wide unique identification for each concept, disambiguating terms which are ambiguous in natural language and connecting annotations to additional metadata about each entity
- create machine-processable annotations of the video in RDF, using the LinkedTV Ontology ⁷ which follows the Open Annotation Model ⁸ with specific extensions for multimedia annotation

⁴ <http://webLyzard.com/video>

⁵ <http://www.w3.org/TR/media-frag/>

⁶ <http://nerd.eurecom.fr>

⁷ <http://linkedtv.eu/ontology>

⁸ <http://www.openannotation.org/>

- enabling the computer-aided ‘semantic search’ over video at the fragment level by storing the generated RDF in a triple store (Sesame) where it can be queried using SPARQL in combination with queries to complementary Linked Data repositories.

3 Implementation and results

The public demonstrator⁹ initially incorporated 297 annotated YouTube videos (the videos crawled by MediaWatch in September and October 2013). Now more YouTube videos are returned for a search (listed in the Documents tab) based on locating the search term within the video transcript, and the search can be expanded to show each video fragment containing the search term (listed in the Quotes tab). For the purpose of easily browsing through transcribed videos, a new Video Tab (cf. top right corner, Figure 1) plays back the entire video (from the Documents tab) or just the fragment which matched the search (from the Quotes tab). An administrator interface is also available to allow admins to launch new video processing batches via videoLyzard as well as monitor running processes through to the successful export of the generated video annotations. Considering the small video dataset which has already been analyzed, which is much lower than the total number of YouTube videos in webLyzards crawl index, it is noteworthy how much new relevant content can be uncovered now that the platform search is able to include video fragments in its search index. For example, the search term “hydroelectricity” in the live site returns a total of 3 YouTube videos for the period November 2013 to July 2014. On the other hand, the prototype with videoLyzard annotations finds 6 YouTube video fragment matches for the 2 month period alone, all matched against semantically similar terms (hydropower, hydro geothermal, hydro-electric) which have been normalized to the DBPedia term for hydroelectricity in the NER step.

4 Future work and conclusion

Our next step is that the number of annotated YouTube videos will be scaled up, with the goal to reach to near real time accessibility to annotated YouTube content. Based on knowledge of common terms in the specific domain (climate change), we also plan to research means to clean up YouTube’s automatic transcriptions prior to performing a more domain-specific NER over them. Since a video fragment at sentence level typically does not contain enough context for viewer understanding, we also want to explore less granular fragmentation of the videos for playback, as well as use of semantic and sentiment information attached to fragments to drive exploration of related (or opposing) fragments around a topic. In conclusion, given the growing use of audiovisual content to

⁹ Available at <http://link.weblyzard.com/video-showcase> with an increasing number of annotated videos.

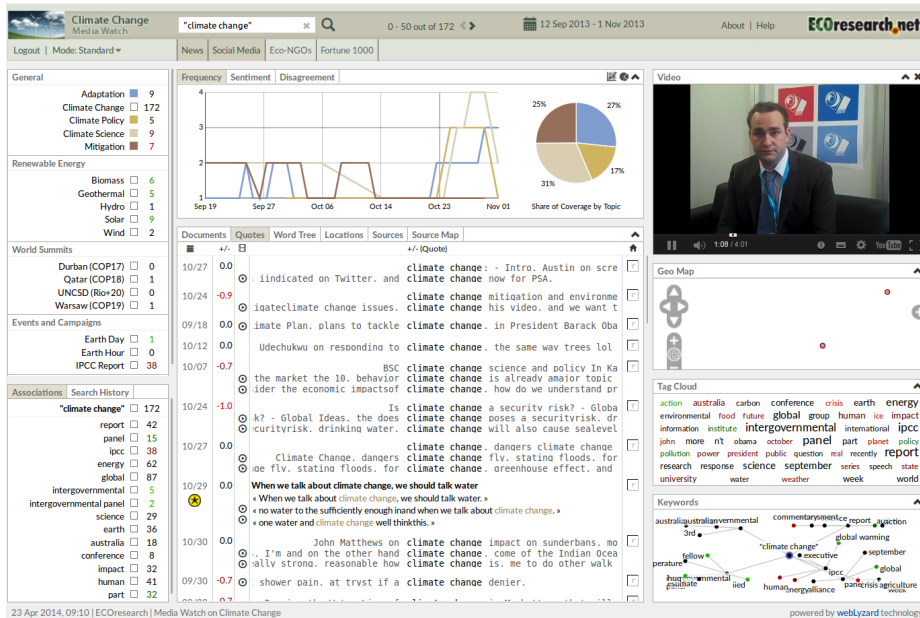


Fig. 1: Video fragment search and playback in webLyzard

communicate about topics online as opposed to just text, Web intelligence platforms miss out on significant amounts of information when they do not consider video material such as that being shared by organisations and individuals on YouTube. The videoLyzard prototype shows that even a small amount of video analysis can uncover additional intelligence for stakeholders, with semantic technologies playing a key role in associating content to distinct entities.

Acknowledgments

This work is supported by the EU FP7 funded Support Action MediaMixer (www.mediamixer.eu)

References

1. "Media Watch on Climate Change - Visual Analytics for Aggregating and Managing Environmental Knowledge from Online Sources". A Scharl, A Hubmann-Haidvogel, A Weichselbraun, H-P Lang and M Sabou. In 46th Hawaii International Conference on Systems Sciences (HICSS-46), Maui, USA, January 2013
2. "Second Demonstrators", L Nixon et al., MediaMixer Deliverable D2.2.3, April 2014

EMBench: Generating Entity-Related Benchmark Data

Ekaterini Ioannou¹ * and Yannis Velegrakis²

¹ Technical University of Crete, Greece, ioannou@softnet.tuc.gr

² University of Trento, Italy, velgias@disi.unitn.eu

Abstract. The *entity matching* task aims at identifying whether instances are referring to the same real world entity. It is considered as a fundamental task in data integration and cleaning techniques. More recently, the entity matching task has also become a vital part in techniques focusing on *entity search* and *entity evolution*. Unfortunately, the existing data sets and benchmarking systems are not able to cover the related evaluation requirements. In this demonstration, we present EMBench; a system for benchmarking entity matching, search or evolution systems in a generic, complete, and principled way. We will discuss the technical challenges for generating benchmark data for these tasks, the novelties of our system with respect to existing similar efforts, and explain how EMBench can be used for generating benchmarking data.

1 Introduction

The entity matching task aims at identifying instances representing the same real world entity, such as an author or a conference [6]. Existing matching approaches are typically based on some similarity function that measures syntactic and semantic proximity of two instances. Depending on the results of this comparison, it is decided whether the two instances are matching or not. More advance matching approaches exploit relationships between instances [1,2], the use of blocking for reducing the required processing time [7,8], and using information encoded in the available schemata [3,4].

Despite the many different techniques for entity matching there is no evaluation methodology that covers all the aspects of matching tasks or at least giving the user the ability to test the aspects of interest. Most matching techniques have followed their own ad-hoc evaluation approach, tailored to their own specific goals. Comparison among entity matching systems and selection of the best system for a specific task at hand is becoming a challenge. Developers can not easily test the new features of the products they develop against competitors, practitioners can not make informative choices for the most suitable tool to use, and researchers can neither compare the techniques they are developing against those already existing, neither identify existing limitations that can serve as potential research directions.

In this demonstration, we will present and discuss the EMBench system for benchmarking entity matching systems in a generic, complete, and principled way [5]. The

* This research has been co-financed by the European Union (European Social Fund ESF) and Greek national funds through the Operational Program “Education and Lifelong Learning” of the National Strategic Reference Framework (NSRF) - Research Funding Program: Thalys. Investing in knowledge society through the European Social Fund.

system provides a series of scenarios that cover the majority of the matching situations that are met in practice and which the existing matching systems are expected to support. EMBench is fully configurable and allows the dynamic (i.e., on-the-fly) generation of the different test cases in terms of different sizes and complexities both at the schema and at the instance level. The fact that the entity matching scenarios are created in a principled way, allows the identification of the actual type of heterogeneities that the under evaluation matching system does not support. This is a fundamental difference from other existing benchmark or competition-based approaches that come with a static set of cases that do not always apply in all the real world scenarios.

The following URL provides an online access to the system as well as the sources code and binary file, and the details can be found in the full version of the paper [5]:

<http://db.disi.unitn.eu/pages/EMBench/>

2 Entity Matching Scenarios

To generate test cases in a systematic way, we introduce the notion of a *scenario*. A scenario is a tuple $\langle e_n, I, e_r \rangle$ where e_n is an entity, I is an entity collection, and e_r an entity from I referred to as the *ground truth*. The scenario is said to be *successfully executed* by an entity matching technique if the technique returns the entity e_r as a response when provided as input the pair $\langle e_n, I \rangle$, i.e., returns e_r as the best match of e_n in the entity collection I .

EMBench creates a scenario by first selecting an entity e_r from the collection I and a series of modifiers f_1, f_2, \dots, f_n . It then applies the modifiers over the selected entity, i.e., $e_r \xrightarrow{f_1} e_1, \xrightarrow{f_2} \dots \xrightarrow{f_n} e_n$, and generates as a scenario the triple $\langle e_n, I, e_r \rangle$.

Each modifier reflects a specific heterogeneity that matching tasks are frequently requested to detect. An example of such a category of modifiers is *Syntactic Variations* and it includes modifiers such as misspellings, word permutations, aliases, abbreviations, and homonymity. *Structural Variations* is another category of modifiers. These modifiers exploit variations on the attribute level. For example, we might have entities that use a set of attributes to describe some information while others entities use just one attribute (e.g., human names might be split into first name and last name, or may not). Another category is *Entity Evolution* simulating scenarios in which the entities have modifications due to time. These modifications can be, for example, changes in the attribute values, elimination of attributes, or addition of new attributes.

An important feature of the system is that the data engineer that created the scenarios can choose not only the case but also the size of data instance to generate. In this way the matching algorithm is tested not only in terms of effectiveness but also in terms of efficiency (scalability).

3 The EMBench System

Figure 1(a) illustrates the architecture of the system. As shown, EMBench maintains a *Repository* that contains the data used during the collection generation. The synthetic data generated by EMBench are not completely random strings but are based on real world values following realistic scenarios. This is achieved by *Shredders*, i.e., software components that receive a source and shreds it into a series of *Column Tables*. The system incorporates general purpose shredders (e.g., relational databases, XML files) as

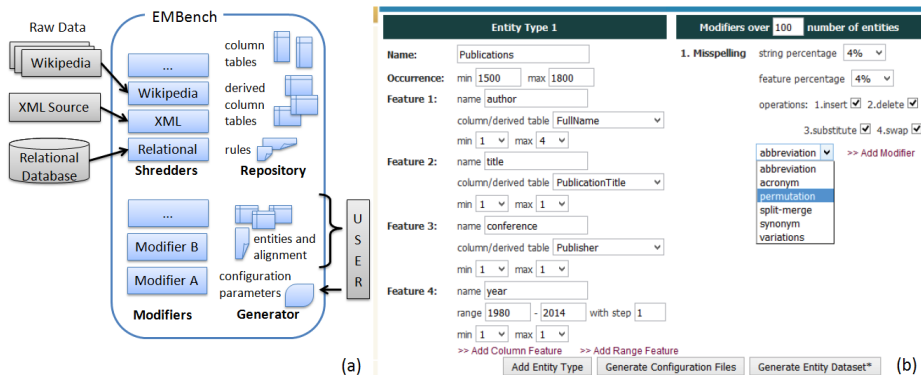


Fig. 1. (a) An illustration of the EMBench’s architecture. (b) A screenshot of the EMBench GUI for creating an entity collection.

well as shredders specifically designed for popular systems (e.g., Wikipedia, DBPedia, Amazon, IMDb, DBLP, OKKAM).

The system also supports cleaning the repetitive, overlapping, or complementary information in the resulted column tables. Among the processes incorporated for this functionality, we have rules that specify how the values of the column tables are to be combined together or modified and guide the creation of a new set of column tables, referred to as the *Derived Column Tables*. Note that a derived column table may be created through an identify function rule, meaning that it is considered a derived table without any modification.

There is no need to shred the original sources or to create the derived column tables every time the benchmark needs to run. Once they are created, they remain in the repository until deleted or overwritten. Actually, the current version of EMBench contains a *Default Data Collection* that is considered sufficient for the realistic evaluation of matching tasks. For instance, it contains 49299 feminine names, 74079 masculine names, 4003 diseases, 84847 companies, and 11817 universities.

The *Entity Generator* creates an entity collection I of N entities by constructing an entity for every tuple of the populated table R . Each such entity will have M attributes, one for every of the M attributes of the table R . EMBench provides two options for selecting the N values from the derived column table: (i) a random selection with or without repetitions, and (ii) select values following the Zipfian distribution.

As mentioned in Section 2, EMBench includes a set of *Entity Modifiers* that modify in various ways the data of an entity collection and construct a new entity collection with a high degree of heterogeneity. The used modifiers, their order and the modification degree is something that is specified by a set of configuration parameters. These parameters have some default values in the system but can also be modified by the user.

Overall, EMBench offers three main functionalities. The first is to create a source repository by importing data using shredders. The second is to generate entity collections using the data from the source repository. The third functionality is to evaluate matching algorithms. To ease the use of these functionalities, EMBench is in general fully parametrized through a configuration file. In addition, EMBench is accompanied

with a user interface that allows the specification of the parameters that build the configuration file on-the-fly and run EMBench (shown in Figure 1(b)).

4 Demonstration Highlights

In the proposed demonstration we will discuss with the audience the functionalities and abilities of EMBench. We will particularly focus on the following four parts.

A. Using EMBench. During the first part we will discuss the two available ways for using EMBench. The first is the usage through a configuration file, which allows providing a description of the functionalities that can be executed, for example which shredders to run, or which matching tasks to evaluate. The second usage is through the EMBench GUI (shown in Figure 1(b)). The GUI provides an alternative mechanism for selecting EMBench’s configuration and executing the functionalities of EMBench.

B. Repository and Default Data Collection. The second part of the demonstration focuses on the repository. We will present the data included in the default data collection, and illustrate how to use existing EMBench shredders for importing additional data. We will also explain how to create, configure, and execute new shredders.

C. Creating Entity Collections. In the subsequent part of the demonstration we will present the creation of collections. This includes describing the schema for the entities to be generated (e.g., maximum number of entity attributes, value distribution, column tables). It also includes the specification and configuration of the modifiers.

D. Evaluating Algorithms using EMBench. The last part of the demonstration focuses on illustrating how EMBench can be used for evaluating algorithms. We will discuss the metrics that are currently incorporated in EMBench and how additional ones can be easily implemented. Furthermore, we will present and illustrate the supported matching-related tasks (i.e., one-to-one matching and blocking).

The demonstration is intended for researchers and practitioners alike. The conference participant will have the opportunity to understand the principles behind the benchmark. This will help the participants in evaluating and testing new matching systems in order to select the one that best fits a task at hand, but will also give valuable insight on how to design and improve matching systems.

References

- [1] I. Bhattacharya and L. Getoor. Deduplication and group detection using links. In *LinkKDD*, 2004.
- [2] X. Dong, A. Halevy, and J. Madhavan. Reference reconciliation in complex information spaces. In *SIGMOD*, 2005.
- [3] J. Euzenat and P. Shvaiko. *Ontology matching*. Springer-Verlag, 2007.
- [4] F. Giunchiglia, P. Shvaiko, and M. Yatskevich. S-Match: an algorithm and an implementation of semantic matching. In *Semantic Interoperability and Integration*, 2005.
- [5] E. Ioannou, N. Rassadko, and Y. Velegrakis. On generating benchmark data for entity matching. *J. Data Semantics*, 2(1), 2013.
- [6] E. Ioannou and S. Staworko. Management of inconsistencies in data integration. In *Data Exchange, Information, and Streams*, 2013.
- [7] G. Papadakis, E. Ioannou, C. Niederée, T. Palpanas, and W. Nejdl. Beyond 100 million entities: large-scale blocking-based resolution for heterogeneous data. In *WSDM*, 2012.
- [8] S. Whang, D. Menestrina, G. Koutrika, M. Theobald, and H. Garcia-Molina. Entity resolution with iterative blocking. In *SIGMOD*, 2009.

Demonstration of multi-perspectives exploratory search with the Discovery Hub web application

Nicolas Marie^{1,2} and Fabien Gandon¹

¹ WIMMICS, INRIA Sophia-Antipolis, Sophia Antipolis, France

² Alcatel-Lucent Bell Labs, Nozay, France

{firstname.lastname}@inria.fr

Abstract. This paper describes a demonstration of the Discovery Hub exploratory search system. The demonstration focuses on the exploration of topics through several perspectives.

1 Introduction

Exploratory search refers to cognitive-consuming search tasks like learning or investigation. There is a need to develop systems optimized for supporting exploratory search as the today widely-used search engines fail to efficiently support exploratory search [1]. Linked data offers exciting perspectives in this context and several systems were already published. We want to reach a new level in linked data based exploration by allowing the users to unveil *knowledge nuances* corresponding to specific facets of interest of the topic explored. In this demonstration paper we first give a brief overview of linked data based exploratory search systems. Then we present the Discovery Hub web application and focus more particularly on its multi-perspectives exploration capacity. Finally we present the demonstration scenario we propose.

2 Linked data based exploratory search

The amount of contributions at the crossroad of semantic search and exploratory search is increasing today. The linked data knowledge, and especially DBpedia, allows to design new information retrieval approaches and interaction models that efficiently support exploratory search tasks. **Yovisto**³ (2009) is an academic videos platform that retrieves topics' suggestions that are semantically related to the users' query. The objective is to ease the exploratio of the videos collection. **Lookup Explore Discover**⁴ (2010) helps the users to compose queries about topics of interest by suggesting related query-terms. Once the query is composed the system retrieves the results from others several services such as search engines and social networks. **Aemoo**⁵ (2012) offers a graph-view on topics of interest. The graph shows their neighborhood filtered by a semantic pattern.

³ <http://www.yovisto.com/>

⁴ <http://sisinflab.poliba.it/led/>

⁵ <http://wit.istc.cnr.it/aemoo>

The users can reverse the filtering to show more surprising knowledge. They can also ask for explanations (cross-references in Wikipedia) about the relations between the shown resources. The **Seevl**⁶ (2013) demonstrator is a music discovery platform implementing a linked data based recommendation algorithm. The DBpedia semantics are also used in Seevl to support browsing (e.g. by music genres, band members) and to provide explanations about the recommendations (showing the shared properties between the artists). **Linked Jazz**⁷ (2013) aims to capture the relations within the American jazz community in RDF. The authors rely on a large amount of jazz people interviews transcripts. These transcripts are automatically processed and then finely analyzed through a crowd-sourced approach.

The approaches recently published in the literature produced good results when evaluated. Nevertheless a common limit of the existing linked data based exploratory search systems is the fact they constrain the exploration through single results selection and ranking schemes. The users cannot influence the retrieved results to reveal specific aspects of knowledge that interest them in particular.

3 Multi-perspectives exploratory search

The framework and models implemented by the Discovery Hub application was presented in [2]. Contrary to other systems it does not pre-compute and store the results for later retrieving. Instead it computes the results on demand thanks to a semantics-sensitive graph traversal algorithm. The algorithm is applied on a small amount of data stored in a local and transient triple store. The data is incrementally imported at query time using SPARQL queries sent to the targeted SPARQL endpoint (DBpedia in the case of Discovery Hub). The objective of this step is to identify a set of relevant results related to the initial topics of interest that will be explored by the user [2]. The web application demonstrating the framework, called Discovery Hub, is available online⁸ and was showcased in several screencasts⁹.

The fact the results are computed at query-time allows to let the users control several computation parameters through the interface and to offer multi-perspective exploratory search. Indeed, the objects described in linked data datasets can be rich, complex and approached in many manners. For example, a user can be interested in a painter (e.g. *Claude Monet* or *Mary Cassat*) in many ways: works, epoch, movement, entourage, social or political contexts and more. The user may also be interested by basic information or unexpected ones depending on his actual knowledge about the topic. He may also want to explore the topic through a specific culture or area e.g. impressionism in American or French culture.

⁶ <http://play.seevl.fm>

⁷ <http://linkedjazz.org/>

⁸ <http://discoveryhub.co>, current CPU-intensive experiments might slow down the search temporarily

⁹ <https://www.youtube.com/user/wearediscoveryhub/videos>

The framework allows three operations for building such exploration perspectives, detailed in [3]. (1) The users can **specify criteria of interest** and disinterest that are used by the framework during the sample importation and its computation. The DBpedia categories are used for this purpose, see Figure 1. The objective is to guide the algorithm in order to retrieve results that are more specifically related to the aspects that interest the user, see example of queries and results in Table 1. (2) It is possible to **inject randomness** in the algorithm values in order to modify the ranking scheme and expose more unexpected results¹⁰. (3) With the proposed framework it is easy to **change the data source** used to process the query¹¹. In the context of DBpedia it enables the use of the DBpedia international chapters like the French, German, Italian ones¹² to leverage cultural bias.

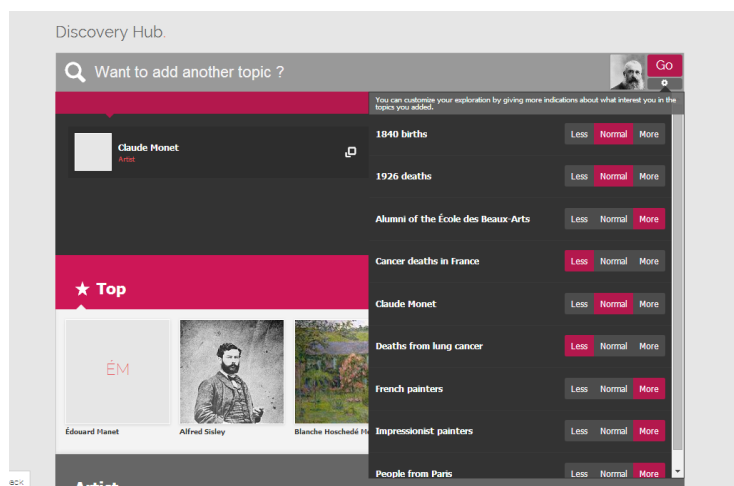


Fig. 1. Discovery Hub interface, criteria of interest specification query

4 Demonstration scenario

The demonstration will be constituted of a sequence of interactions like the ones presented in the previously published screencasts. First the user launches a simple query (query 1 in Table 1), he examines the results. At this point we show the audience the Discovery Hub functionalities supporting the exploration and understanding. We will focus on the faceted browsing aspect, the explanations and the redirections toward third-party platforms. During this step we will engage the conversation about how we compute the results and what is the software architecture.

¹⁰ this advanced query mode is supported by the framework but currently not available through the interface and will be integrated soon

¹¹ idem 10

¹² wiki.dbpedia.org/Internationalization/Chapters?v=190k

Table 1. Results of three queries about Claude Monet using the criteria specification

Query	Claude Monet (1)	Claude Monet (2)	Claude Monet (3)
Criteria	None	Impressionist painters + Artists from Paris - People from Le Havre - Alumni of the École des Beaux-Arts - French painters -	Impressionist painters - Artists from Paris + People from Le Havre + Alumni of the École des Beaux-Arts + French painters +
Results			
1	Pierre-Auguste Renoir	Theodore Robinson	Pierre-Auguste Renoir
2	Alfred Sisley	Edouard Manet	Gustave Courbet
3	Edouard Manet	Alfred Sisley	Edgar Degas
4	Mary Cassatt	Wladyslaw Podkowiński	Jacques-Louis David
5	Camille Pissarro	Leslie Hunter	Jean-Baptiste-Camille Corot
6	Edgar Degas	Theodore Earl Butler	Jean-François Millet
7	Charles Angrand	Lilla Cabot Perry	Paul Cézanne
8	Gustave Courbet	Frank Weston Benson	Marc Chagall

During the results examination we will voluntarily focus on the French impressionist painters that were close to Monet. At this point the user might be interested in the relations of Monet with the non-French impressionists (query 2 in Table 1). We will explain the querying system for criteria of interest specification and then emphasize the differences between the results obtained with query 1 and 2.

To continue in the same logic we will submit the query 3 as well as a query with a high level of randomness and one using the French chapter of DBpedia in several tabs. We let the audience compare the results. We seek an interactive demonstration by encouraging the audience to try the application while commenting the system more than a strict and pre-defined sequence of interactions (which serves only to start the interactions).

5 Conclusion and perspectives

Discovery Hub is a linked data based exploratory search system built on the top of DBpedia. With this demonstration we want to show the value of linked data for exploratory search. Mature datasets like DBpedia allow the creation of new information retrieval approaches as well as new interaction models. More specifically we want to demonstrate the multi-perspectives exploratory search capacities of Discovery Hub. Thanks to the demonstration track we hope to have discussions with other researchers about the perspectives we envision for Discovery Hub. It notably includes an approach where the user can specify or change the specified criteria of interest interactively in order to re-rank the results without relaunching the whole query-process.

References

1. G. Marchionini. Exploratory search: from finding to understanding. *Communications of the ACM*, 49(4):41–46, 2006.
2. N. Marie, F. Gandon, M. Ribière, and F. Rodio. Discovery hub: on-the-fly linked data exploratory search. In *Proceedings of the 9th International Conference on Semantic Systems*. ACM, 2013.
3. N. Marie, M. Gandon, Alain Giboin, and E. Palagi. Exploratory search on topics through different perspectives with dbpedia. In *Proceedings of the 10th International Conference on Semantic Systems*. ACM, 2014.

Modeling and Monitoring Processes exploiting Semantic Reasoning

Piergiorgio Bertoli¹, Francesco Corcoglioniti², Chiara Di Francescomarino², Mauro Dragoni², Chiara Ghidini², Michele Nori¹, Marco Pistore¹, and Roberto Tiella²

¹ SayService, Trento, Italy bertoli|nori|pistore@sayservice.it

² FBK—IRST, Trento, Italy corcoglionio|dfmchiara|dragoni|ghidini|tiella@fbk.eu **

Abstract. Data about process executions has witnessed a notable increase in the last decades, due to the growing adoption of Information Technology systems able to trace and store this information. Meanwhile, Semantic Web methodologies and technologies have become more and more robust and able to face the issues posed by a variety of new domains, taking advantage of reasoning services in the “big data” era. In this demo paper we present *ProMo*, a tool for the collaborative modeling and monitoring of Business Process executions. Specifically, by exploiting semantic modeling and reasoning, it enables the reconciliation of business and data layers as well as of static and procedural aspects, thus allowing business analysts to infer knowledge and use it to analyze process executions.

1 Introduction

The last decades have witnessed a rapid and widespread adoption of Information Technology (IT) to support business activities in all phases. As a side effect, IT systems have made available huge quantities of data about process executions, thus enabling (i) to monitor the actual execution and the progress of (instances of) Business Processes (BPs); (ii) to provide statistical analysis; (iii) to detect deviations of process executions from process models (e.g., [1]); and (iv) to identify problems in process executions.

Meanwhile, Semantic Web technologies have known an important growth and have made available powerful reasoning services able to reason on complex domains, as well as technologies able to deal with huge quantities of data. This opens the way to the use of Semantic Web technologies for process modeling and monitoring and for the analysis of processes characterizing complex scenarios as those of large organizations.

In these complex scenarios, knowledge can be classified in two orthogonal ways. First, we distinguish between a *dynamic dimension*, which concerns the procedures and the activities carried out by the organization for realizing specific objectives, and a *static dimension*, which concerns the organization structure (e.g., the role hierarchy), the data structure (e.g., the document organization), and the relationships among these and other domain entities. Then, knowledge can be ascribed to two layers: the *IT layer*, which concerns the actual data items processed by IT systems; and the *business layer*,

** This work is supported by “ProMo - A Collaborative Agile Approach to Model and Monitor Service-Based Business Processes”, funded by the Operational Programme “Fondo Europeo di Sviluppo Regionale (FESR) 2007-2013 of the Province of Trento, Italy.

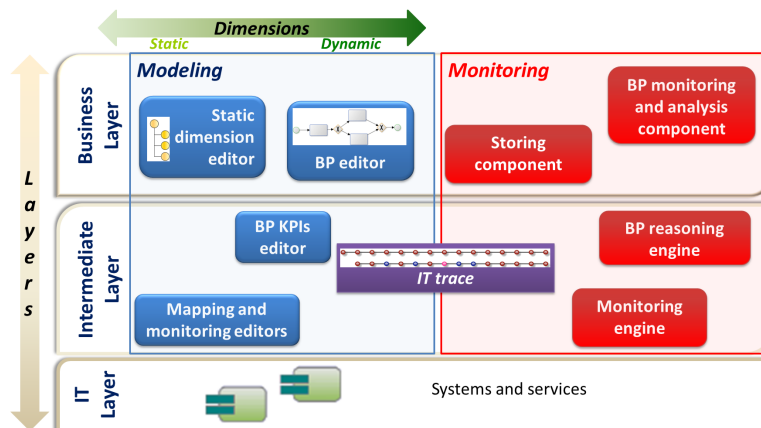


Fig. 1: *ProMo* overview

which concerns the models of the dynamic and static aspects of the organization domain. Given this frame, two main challenges need to be faced: (i) bridging the unavoidable gap between the business and the data layer; and (ii) reconciling the static and dynamic dimensions so to make them available for monitoring and analysis purposes.

In this demo we present and showcase *ProMo*, a tool that exploits Semantic Web technologies to address the above challenges through an integrated representation of knowledge, enabling the collaborative modeling, monitoring and analysis of business processes. By reconciling all these different dimensions and layers, *ProMo* overcomes existing approaches. In the remainder we describe how *ProMo* reconciles the business and IT layers and the static and dynamic dimensions, introducing the *ProMo* main components that will be demonstrated live during the Posters and Demo session.

2 Reconciling Business and IT layers

Aligning the business and IT layers is a difficult task. For example, process monitoring at the IT layer cannot observe data exchanged on paper documents or user activities not mediated by IT systems, and thus brings only partial information on which activities were executed and what data or artifacts they produced. Even when IT data exists, it is not easy to associate it to a specific process instance. Indeed, IT services can be shared by process classes and instances, and traced information can be hard to disambiguate.

ProMo solution to this problem is based on the introduction of an *intermediate layer* (Figure 1), which enables the communication between the business and the IT layers through an intermediate model. Such a model formalizes the relationships between business models and information extracted at the IT layer and relies on the integrated representation of all the information collected about a process execution (the *IT-trace*).

To accomplish its goal, *ProMo* integrates a *modeling* component and a *monitoring* component. At the business level, the *modeling* component provides *MoKi-ProMo*, a customized version of the MediaWiki-based³ tool MoKi [2] for the collaborative modeling of processes and ontologies. At the intermediate layer *ProMo* provides (i) mapping

³ <http://www.mediawiki.org>

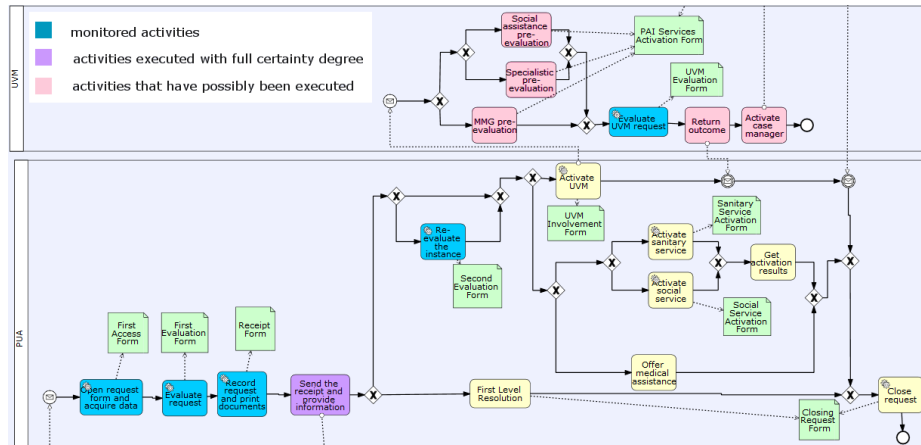


Fig. 2: *MoKi-ProMo* visualization of a reconstructed (partial) trace

and monitoring editors that allow *IT Experts* (taking advantage of the *Domain Experts* modeling) to specify, respectively, aggregation/monitoring rules and the relationships between business models and the information extracted at IT level; and (ii) an editor for defining interesting Key Performance Indicators (KPIs) to be monitored. Specifically, the input required at the intermediate layer is provided by experts by using the *DomainObject* language [3] for defining mapping properties, an ad-hoc rule language for monitoring rules, and SPARQL queries for business KPIs.

At run-time, whenever an IT-level event occurs, it is captured and handled by the monitoring component. In detail, the event is managed by the monitoring engine, which, based on the specification and rules defined at design-time, correlates and aggregates events, produces new control events, monitors and maps the events to the corresponding one(s) at the business layer and eventually produces the IT-trace. The information in the IT-trace, which in many cases is only partial with respect to a complete execution flow of a designed process model, is hence passed to a reasoning engine. Such an engine, by taking advantage of the business knowledge, reconstructs missing information by applying model-driven satisfiability rules [4] and the reconstructed trace is then visualized by the BP monitoring and analysis component. Figure 2 shows how a reconstructed (partial) execution trace is visualized in *MoKi-ProMo*, pointing out the path possibly taken by the process execution and distinguishing between monitored and reconstructed (with some certainty degree) activities. The reconstructed IT-trace is then recorded in a semantic-based knowledge store, which is then queried by the BP monitoring and analysis component in order to provide monitoring services at business level. An implementation built on top of current Semantic Web technologies aims at coping with large quantities of data and high data rates typical of real application scenarios.

3 Reconciling Static and Dynamic Dimensions

Although different in their nature, static and dynamic knowledge about an organization domain are strictly related and should be jointly considered in order to obtain a comprehensive view of the organization processes. Importantly, reconciliation of these two

dimensions should be done both at the business layer, allowing an explicit representation of the links between static and dynamic model elements (e.g., the fact that a process activity operates on a certain document), and at the data layer, allowing the collection, integration and comprehensive querying of static and procedural data.

At the business layer, *ProMo* solution is represented by the modeling component of *MoKi-ProMo*, which allows different experts (e.g., *Business Designers*, *Knowledge Engineers* and *Domain Experts*) to collaboratively model the different static and dynamic aspects describing the domain (see Figure 1). Specifically, *MoKi-ProMo* allows *Domain Experts* and *Knowledge Engineers* to collaboratively model the static aspects of the domain in form of OWL 2 ontologies. Concerning the dynamic aspects, *MoKi-ProMo* customizes the *Oryx* editor⁴ for the BPMN modeling of business processes by introducing symbol variations (e.g., special data objects for explicitly capturing data structures). Moreover, *MoKi-ProMo* also provides an interface allowing *Business Analysts* and *Domain Experts* to edit KPIs of interest, thus enabling them to access IT data from a (static and dynamic) business perspective.

At the IT layer, *ProMo* solution consists in exploiting a Domain ontology [5], consisting of an upper-level cross-domain core and a domain-dependent extension, and a BPMN [6] ontology to build an integrated semantic model combining static and procedural knowledge acquired at modeling time, together with knowledge about IT-data. By leveraging scalable Semantic Web technologies for data storage, reasoning and querying, the semantic model enables *Business Analysts* to query asserted and inferred knowledge and bring execution data analysis at business level. In particular, analytical SPARQL queries combining static and dynamic dimensions with data derived from the IT-layer can be formulated and evaluated, such as the number of times a path is followed or an actor instance executes a business activity, or the average time spent by an actor of a given category to complete the process. Experiments carried out in the context of an Italian use case have shown the applicability of the approach in realistic scenarios [5].

References

1. van der Aalst, W.M.P.: Process Mining: Discovery, Conformance and Enhancement of Business Processes. 1st edn. Springer Publishing Company, Inc. (2011)
2. Ghidini, C., Rospocher, M., Serafini, L.: Conceptual modeling in wikis: a reference architecture and a tool. In: 4th Int. Conf. on Information, Process, and Knowledge Management (eKNOW). (2012) 128–135
3. Bertoli, P., Kazhamiakin, R., Nori, M., Pistore, M.: SMART: Modeling and monitoring support for business process coordination in dynamic environments. In: 15th Int. Conf. on Business Inf. Systems (BIS) - Workshops. Volume 127 of LNBIP., Springer (2012) 243–254
4. Bertoli, P., Di Francescomarino, C., Dragoni, M., Ghidini, C.: Reasoning-based techniques for dealing with incomplete business process execution traces. In: 13th Conf. of Italian Association for Artificial Intelligence (AI*IA). Volume 8249 of LNCS., Springer (2013) 469–480
5. Di Francescomarino, C., Corcoglioniti, F., Dragoni, M., Bertoli, P., Tiella, R., Ghidini, C., Nori, M., Pistore, M.: Semantic-based process analysis. In: 13th Int. Semantic Web Conference (ISWC) - In-use track. (2014) (to appear).
6. Rospocher, M., Ghidini, C., Serafini, L.: An ontology for the Business Process Modelling Notation. In: 8th Int. Conf. on Formal Ontology in Inf. Systems (FOIS). (2014) (to appear).

⁴ <http://bpt.hpi.uni-potsdam.de/Oryx/>

WikipEvent: Temporal Event Data for the Semantic Web

Ujwal Gadiraju, Kaweh Djafari Naini, Andrea Ceroni, Mihai Georgescu, Dang Duc Pham, Stefan Dietze, and Marco Fisichella

L3S Research Center, Leibniz Universität Hannover, Germany
{gadiraju, naini, ceroni, georgescu, pham, dietze, fisichella}@L3S.de

Abstract. In this demo we present *WikipEvent*, an exploratory system that captures and visualises continuously evolving complex event structures, along with the involved entities. The framework facilitates entity-centric and event-centric search, presented via a user-friendly interface and supported by temporal snippets from corresponding Wikipedia page versions. The events detected and extracted using different mechanisms are exposed as freely available Linked Data for further reuse.

Keywords: Events; Temporal Evolution; Wikipedia; RDF; Interface

1 Introduction

Exploratory search systems help users to search, navigate, and discover new facts and relationships. We detect and extract events from different sources and merge these events into a unique *event repository*. Each event is described primarily in terms of (i) a list of entities (Wikipedia pages) participating in the event, (ii) a textual description of the event, (iii) start and end dates of the event, and (iv) the extraction method used to obtain the event. We further classify entities as people, organizations, artifacts, and locations by exploiting the class hierarchy defined in YAGO2 [2], since different entity categories play different roles while participating in an event.

In this demo, we showcase two popular usecase scenarios. First, we show that *WikipEvent* can be used in order to explore the events in which particular entities have been involved. Secondly, we show the suitability of *WikipEvent* to explore the evolution of entities based on the events these are involved in. Both these scenarios can be additionally surveyed based on the temporal dimension. In addition to the events that are presented using a *timeline*¹, we adopt a versioning approach introduced previously[1] in order to display significant versions of wikipages corresponding to the entities involved in the event.

WikipEvent facilitates the understanding of events and their related entities on a temporal basis. Instead of exploration of isolated knowledge bases, WikipEvent takes advantage of the complimentary nature of sources contributing to the underlying event repository.

¹ Interface: <http://wikipeventdemo.l3s.uni-hannover.de/WikiEventEntity/>

2 WikipEvent Data

We extract events from three sources; Wikipedia Current Events portal[3], Yago[2], and an event detection method called *Co-References*[4]. Firstly, the WikiTimes project² provides an API for 50,000 events between 2001-2013, acquired from the Wikipedia Current Events portal. The second event source is the YAGO2 ontology including entities which describe events, e.g. 2011_Australian_Open as well as facts connecting entities, e.g. $\langle \text{BobDylan} \rangle \text{ wasBornIn } \langle \text{Duluth} \rangle$. The Co-References method introduced in [4] extracts Wikipedia pages (entities) related to an event by using the Wikipedia edit history. The edits corresponding to a Wikipedia page (entity) are analysed for indications of the occurrence of an event involving that entity.

The resulting repository contains more than 2.6 million events, extracted from the different sources as shown in Table 1. The contribution from the sources is skewed due to two reasons. Firstly, they cover very different time periods: YAGO2 contains events and temporal facts spanning over thousands of years, Current Events captures events since 2001 only, and, for performance reasons, the Co-References method has been restricted to edits in 2011. In addition, Co-References exclusively analyses entities of type *politician*, while YAGO2 and Current Events contain almost all the Wikipedia pages. To facilitate comparisons across entities occurring in all three sources, in the rest of this section we will consider only those events that occurred in 2011 and involved politicians.

Source	Total	Politicians	Politicians 2011
<i>All</i>	2,629,740	50,168	1,401
<i>YAGO2</i>	2,578,547	42,399	360
<i>Current Events</i>	50,951	7,527	799
<i>Co-Reference</i>	242	242	242

Table 1: Number of events within the event repository, split by different sources.

Co-References is able to detect events with different duration and granularity (from a wrestling match to the Egypt Revolution). YAGO2 mostly contains high-level and well-known events represented through temporal facts regarding entities, often lacking textual descriptions. Current Events portal contains daily events which have a self explanatory textual description and are reliable, thanks to the high level of control within Wikipedia. The complimentary nature of the different sources in terms of complexity (number of participants), duration, and granularity of events is evident. Due to these reasons, also the schemas used to represent events across different sources are distinct, yet overlapping. While certain properties (for instance for time points) are overlapping yet follow different conventions, we lifted the events from different sources into a unified dataset following Linked Data principles and deploying a joint RDF schema.

Exposing Events Data as RDF.

We have exposed the *WikipEvent events* data through a public Linked Data

² <http://data.l3s.de/dataset/wiketimes>

interface using D2R Server³, enabling URI dereferencing via content negotiation and providing a public SPARQL endpoint. This data can be accessed and queried via <http://wikipevent.l3s.uni-hannover.de/> and using our SPARQL endpoint⁴.

Property	Value
<code>lode:atPlace</code>	Virginia
<code>we:event_description</code>	A Virginia judge rules the states prohibition of gay marriage unconstitutional.
<code>we:event_end</code>	Thu Feb 13 01:00:00 CET 2014
<code>we:event_id</code>	Wikitimes_169403
<code>we:event_source</code>	Wikitimes
<code>we:event_start</code>	Thu Feb 13 01:00:00 CET 2014
<code>we:event_story</code>	NULL
<code>lode:involved</code>	Virginia, Gay marriage
<code>lode:involvedAgent</code>	NULL
<code>rdfs:label</code>	events #1
<code>we:reference</code>	http://www.usatoday.com/story/news/nation/2014/02/13/virginia-same-sex-marriage/5473687/
<code>rdf:type</code>	<code>we:Event</code>

Fig. 1: Example of event related data as an RDF repository.

We represent events through established event RDF vocabularies, to facilitate reuse, interpretation and linking of our data through third parties. In particular, we use properties from the LODÉ ontology⁵ to map different properties pertaining to events in our dataset, for instance, property `lode:atPlace` is used as predicate for stating venues of events. Figure 1 presents an example of an event and its entailing properties in our event repository.

3 Use Case Scenarios

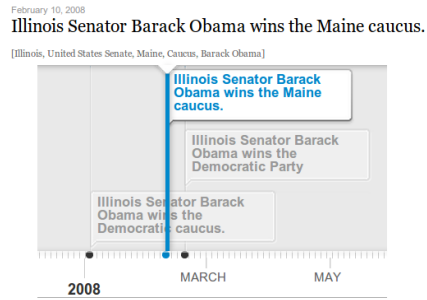
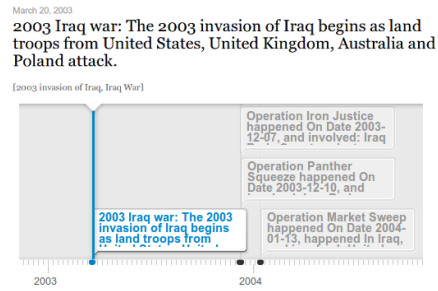
WikipEvent can help students, scholars, historians, or journalists by facilitating temporal search focussed on either the entities at hand or the events. Thus, the WikipEvent framework can be used to satisfy two primary scenarios - entity-based and event-based information needs. Results are presented through a user-friendly interface, that supports faceted search, query-path tracing, query completion, temporal settings, and is abridged with events' sources as well as filters for related entities. The underlying versioning system [1], helps us to identify *significant* wikipage revisions of the entities involved in the event. These significant revisions of wikipages are also presented to the user in addition to the timeline of events. As introduced in our previous work, a *significant revision* is one where the edits between preceding and succeeding revision are above a certain threshold. Here, the notion of significance is modeled based on the Cosine distance between successive revisions [1].

Entity-based Search. Users may want to learn about entities of interest with respect to their temporal participation in events. For example, journalists might aim at studying political affiliations of individuals, and the campaigns

³ <http://d2rq.org/>

⁴ <http://wikipevent.l3s.uni-hannover.de/snorql/>

⁵ <http://linkedevents.org/ontology/>

Fig. 2: Entity-centric search: *Barack Obama*.Fig. 3: Event-centric search: *Iraq War*.

they participated in. Existing systems however, make it cumbersome to easily access this information. The WikipEvent interface overcomes this challenge by presenting a timeline of events that an entity is involved in. Additional filters for relevant entities help users to navigate through the retrieved results. Figure 2 presents an example of an entity-based search on WikipEvent.

Event-based Search. Historic events are a subject of interest to a wide array of people, ranging from students to archivists. WikipEvent facilitates a free-text search for events. Events relevant to a given query are presented in the form of a continuous timeline (Figure 3), while highlighting the entities involved. WikipEvent enables users to sift through event related information on a temporal basis, in order to learn more about the events and the participating entities.

To gain a complete understanding of the WikipEvent framework, we point the reader to a demo video ⁶.

References

1. Andrea Ceroni, Mihai Georgescu, Ujwal Gadiraju, Kaweh Djafari Naini, and Marco Fisichella. Information evolution in wikipedia. In *Proceedings of the 10th International Symposium on Open Collaboration, OpenSym '14*. ACM, 2014.
2. Johannes Hoffart, Fabian M Suchanek, Klaus Berberich, and Gerhard Weikum. Yago2: a spatially and temporally enhanced knowledge base from wikipedia. *Artificial Intelligence*, 194:28–61, 2013.
3. Giang Binh Tran and Mohammad Alrifai. Indexing and analyzing wikipedia’s current events portal, the daily news summaries by the crowd. In *Proceedings of the companion publication of the 23rd international conference on World wide web companion*, pages 511–516, 2014.
4. Tuan A. Tran, Andrea Ceroni, Mihai Georgescu, Kaweh Djafari Naini, and Marco Fisichella. WikipEvent: leveraging wikipedia edit history for event detection. In *Web Information Systems Engineering - WISE 2014, Thessaloniki, Greece, 12-14 October 2014, Proceedings*, 2014.

⁶ A demo video is available on the home screen of the web interface.

Towards a DBpedia of Tourism: the case of Tourpedia

Stefano Cresci, Andrea D’Errico, Davide Gazzé, Angelica Lo Duca,
Andrea Marchetti, Maurizio Tesconi

Institute of Informatics and Telematics, National Research Council,
via Moruzzi 1, 56124 Italy
email: [name].[surname]@iit.cnr.it

Abstract. In this paper we illustrate Tourpedia, which would be the DBpedia of tourism. Tourpedia contains more than half a million places, divided in four categories: accommodations, restaurants, points of interests and attractions. They are related to eight locations: Amsterdam, Barcelona, Berlin, Dubai, London, Paris, Rome and Tuscany, but new locations are continuously added. Information about places were extracted from four social media: Facebook, Foursquare, GooglePlaces and Booking and were integrated in order to build a unique catalogue. Tourpedia provides also a Web API and a SPARQL endpoint to access data.

1 Introduction

The concept of Semantic Web was introduced by Tim Berners Lee in 2001[2]. His main idea consisted in migrating from the Web of documents to the Web of data. The purpose of the Web of data is to connect concepts and contents to each other, instead of simply connecting documents. Thus the Web of data has led to the conversion of existing documents to linked data [6], and to the creation of new datasets¹. Among them, one of the most exploited datasets is DBpedia², which is the linked data version of Wikipedia³.

DBpedia is available in different languages. Its English version contains about 4.0 million things, classified in different categories, including people, places, creative works, organizations, species and diseases. However, DBpedia, as well as Wikipedia, contains only a small number of things related to the tourism domain, such as accommodations and restaurants. In addition, to the best of our knowledge, only few linked datasets have been implemented in the field of tourism. Among them, the case of El Viajero⁴, which provides information about more than 20.000 travel guides, pictures, videos and posts, and that of Accommodations in Tuscany⁵, which contains the list of accommodations in

¹ For a list of shared datasets, please look at: <http://datahub.io>.

² <http://dbpedia.org>

³ <http://wikipedia.org>

⁴ <http://datahub.io/dataset/elviajero>

⁵ <http://datahub.io/dataset/grrt>

Tuscany, Italy. For more details about datasets about tourism, please refer to: <http://datahub.io/dataset?q=tourism>.

In this paper we illustrate Tourpedia, which would be the DBpedia of Tourism. Tourpedia is reachable through its portal⁶ and is available also in the datahub.io platform⁷.

Tourpedia was developed within the OpeNER Project⁸ (Open Polarity Enhanced Name Entity Recognition), whose main objective is to implement a pipeline to process natural language.

The usage of Tourpedia could be very various. For example, it could be used to perform named entity disambiguation in tourism domain, or to extract the most appreciated points of interest in a town.

2 Tourpedia

Figure 1 illustrates the Tourpedia architecture. The Data Extraction module consists of four ad-hoc scrapers, which extract data from four social media: Facebook⁹, Foursquare¹⁰, Google Places¹¹ and Booking¹². We chose these social media firstly because they are very popular and secondly because they provide an easy way to extract data. The scrapers of Facebook, GooglePlaces and Foursquare exploit the RESTful APIs the social media provide, while the Booking scraper extracts information from each accommodation page.

The Named Entity repository contains two main datasets, which belong to the specific domain of tourism: Places and Reviews about places. The dataset of Places contains more than 500.000 places in Europe divided in four categories: accommodations, restaurants, points of interest and attractions¹³. At the moment the following locations are covered: Amsterdam, Barcelona, Berlin, Dubai, London, Paris, Rome and Tuscany. Places were elaborated and integrated through the Data Integration module in order to build a unique catalogue. Data Integration was performed by using a merging algorithm based on distance and string similarity.

The dataset of Reviews contains about 600.000 reviews about places. Reviews were analysed through the OpeNER pipeline in order to extract their sentiment.

2.1 Web application

Tourpedia provides also a Web application¹⁴ [5], which shows the sentiment about places on an interactive map, which is Google Maps-like.

⁶ <http://tour-pedia.org>

⁷ <http://datahub.io/dataset/tourpedia>

⁸ <http://www.opener-project.eu>

⁹ <http://www.facebook.com>

¹⁰ <http://foursquare.com>

¹¹ <https://plus.google.com/u/0/local>

¹² <http://www.booking.com>

¹³ <http://tour-pedia.org/about/statistics.html>

¹⁴ <http://tour-pedia.org/gui/demo/>

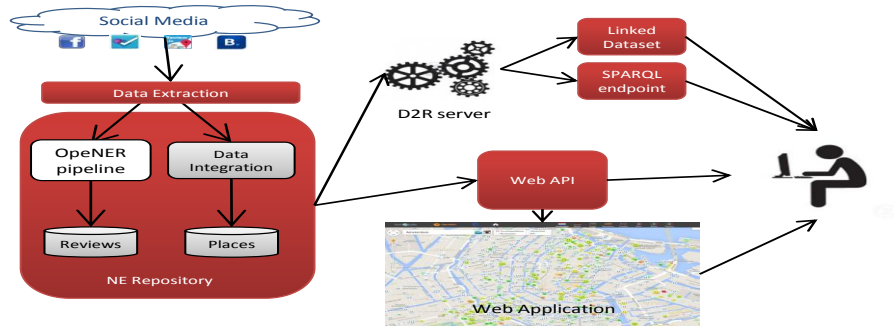


Fig. 1. The architecture of Tourpedia.

The sentiment of a place is calculated as a function of all the sentiments of the reviews about that place. In order to retrieve the sentiment of a review, the OpeNER pipeline was used. In particular, each place is associated to zero or more reviews extracted from social media (i.e. Facebook, Foursquare and Google Places). Each review is processed through the OpeNER pipeline and is associated to a rate, which expresses its specific sentiment.

2.2 Linked Data

Tourpedia is exposed as a linked data node and provides a SPARQL endpoint¹⁵. The service is implemented through the use of a D2R server¹⁶. For each place, the following ontologies are used to represent it: VCARD [9] and DBpedia OWL¹⁷, for generic properties; Acco [8], Hontology [4] and GoodRelations [7] for domain-specific properties. In a previous work [1], we illustrated the employed ontologies and structures of accommodations as linked data. In order to fulfill the principles of linked data [3], each location is linked to the same location in DBpedia.

2.3 Web API

Tourpedia provides a RESTful API¹⁸ to access places and statistics. The output of each request can be JSON, CSV and XML. For example, a search request about Places is an HTTP URL of the following form:

`http://tour-pedia.org/api/getPlaces?parameters`

where `parameters` must be at least one one of the following: *location* (the location of the places), *category* (the type of the places such as accomodation), attraction, restaurant, poi), and *name* (the keyword to be searched).

¹⁵ <http://tour-pedia.org/sparql>

¹⁶ <http://d2rq.org/>

¹⁷ <http://wiki.dbpedia.org/Ontology>

¹⁸ <http://tour-pedia.org/api>

3 Conclusions and Future Work

In this paper we have illustrated Tourpedia, which would be the DBpedia of Tourism. It could be interesting a deeper connection between Tourpedia and DBpedia. At the moment, in fact, only locations are connected to DBpedia. As future work, we are going to align also attractions and points of interest contained in Tourpedia to DBpedia.

Tourpedia could be exploited both by tourism stakeholders to get the sentiment about touristic places and by common users.

At the moment, the procedure to update datasets is manual. As future work, we are going to define a semi-automatic procedure to update them and to add new locations.

Acknowledgements

This work has been carried out within OpeNER project, co-funded by the European Commission under the FP7 (7th Framework Programs Grant Agreement n. 296451).

References

1. Bacciu, C., Lo Duca, A., Marchetti, A., Tesconi, M.: Accommodations in Tuscany as Linked Data. In: Proceedings of The 9th edition of the Language Resources and Evaluation Conference (LREC 2014). pp. 3542–3545 (May, 26–31 2014)
2. Berners-Lee, T., Hendler, J., Lassila, O.: The semantic web. *Scientific American* 284(5), 34–43 (May 2001), <http://www.sciam.com/article.cfm?articleID=00048144-10D2-1C70-84A9809EC588EF21>
3. Bizer, C., Heath, T., Berners-Lee, T.: Linked data - the story so far. *Int. J. Semantic Web Inf. Syst.* 5(3), 1–22 (2009)
4. Chaves, M.S., de Freitas, L.A., Vieira, R.: Hontology: A multilingual ontology for the accommodation sector in the tourism industry. In: Filipe, J., Dietz, J.L.G. (eds.) KEOD. pp. 149–154. SciTePress (2012)
5. Cresci, S., D’Errico, A., Gazzé, D., Lo Duca, A., Marchetti, A., Tesconi, M.: Tourpedia: a Web Application for Sentiment Visualization in Tourism Domain. In: Proceedings of The OpeNER Workshop in The 9th edition of the Language Resources and Evaluation Conference (LREC 2014). pp. 18–21 (May, 26 2014)
6. Heath, T., Bizer, C.: *Linked Data: Evolving the Web into a Global Data Space*. Morgan & Claypool, 1st edn. (2011), <http://linkeddatabook.com/>
7. Hepp, M.: Goodrelations language reference. Tech. rep., Hepp Research GmbH, Innsbruck (2011)
8. Hepp, M.: Accommodation ontology language reference. Tech. rep., Hepp Research GmbH, Innsbruck (2013)
9. Iannella, R., McKinney, J.: VCARD ontology. Available at: <http://www.w3.org/TR/vcard-rdf/>. Tech. rep. (2013)

Using Semantics for Interactive Visual Analysis of Linked Open Data

Gerwald Tschinkel¹, Eduardo Veas¹, Belgin Mutlu¹ and Vedran Sabol^{1,2}

¹ Know-Center `gtschinkel|eveas|bmutlu|vsabol@know-center.at`

² Graz University of Technology

Abstract. Providing easy to use methods for visual analysis of Linked Data is often hindered by the complexity of semantic technologies. On the other hand, semantic information inherent to Linked Data provides opportunities to support the user in interactively analysing the data. This paper provides a demonstration of an interactive, Web-based visualisation tool, the “Vis Wizard”, which makes use of semantics to simplify the process of setting up visualisations, transforming the data and, most importantly, interactively analysing multiple datasets using brushing and linking methods.

1 Introduction

An objective of the CODE³ project is to make Linked Data accessible to novice users by providing easy to use methods for visual data analysis. This is hard to achieve with current Linked Data tools, which require user’s knowledge of semantic technologies (such as SPARQL). This paper demonstrates how semantic information can be used to support the interactive analytical process, without the need for users to understand the complexities of the underlying technology.

Within CODE we use the RDF Data Cube Vocabulary⁴ for describing statistical datasets. Our “Vis Wizard”⁵ tool provides an intuitive, easy to use interface supporting visualisation and interactive analysis of RDF Cubes. In the Vis Wizard we utilise semantic information from Linked Data to support the user in:

1. Selecting and configuring the visualisations
2. Aggregating datasets
3. Brushing and linking over multiple datasets

This paper illustrates the use of semantic technologies in a visual analytics tool that enables novice users to perform complex operations and analyses on Linked Data. The demonstration focuses mainly on step 3, with a screencast of the demonstration also being available⁶.

³ <http://code-research.eu>

⁴ <http://www.w3.org/TR/vocab-data-cube>

⁵ <http://code.know-center.tugraz.at/vis>

⁶ <http://youtu.be/aBfuGhgVaxA>

Related work: A wide range of tools offers functionalities for visualising and interacting with data, but only a few rely on semantic information to support the analytical process. Tableau [6] provides a mighty visualisation toolset, however it does not make use of semantic information for assisting the user. The CubeViz Framework [5] facilitates visual analytics on RDF Data Cubes, but does not use semantics for the user interface. CubeViz supports no brushing, no possibility to compare datasets directly and no automatic selection of visualisations. Cammarano et al. [1] introduces a method to automatically analyse data attributes and map them to visual properties of the visualisation. Even so, this does not include an automatic selection of visualisation types.

2 The Linked Data Vis Wizard

The underlying thought is to make the user capable of visually analysing data without knowing about the concept of Linked Data or RDF Data Cubes. However, the Vis Wizard utilises the available semantic information to support users in interacting with the data and performing analytical tasks.

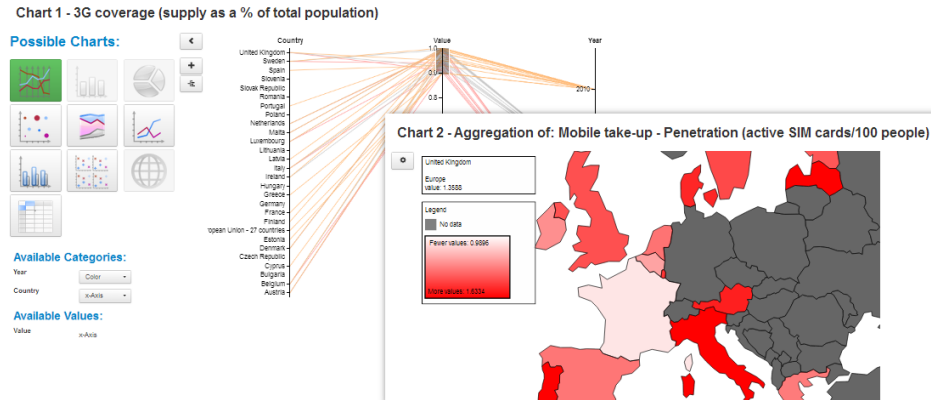


Fig. 1. Two RDF Data Cubes are shown in the Vis Wizard. Brushing the 3G coverage value in the parallel coordinates highlights corresponding countries in the geo-chart.

Scenario: Figure 1 compares two datasets taken from the EU Open Data Endpoint⁷ in the Vis Wizard. The first one, shown in parallel coordinates, represents the 3G coverage in Europe, as percentage value, per country for each year. The second dataset, shown in the geo-chart, contains active SIM cards per 100 people (encoded by colour-grading) for countries in Europe. In the following we use the Vis Wizard to gain insights into the data and ascertain the datasets correlate.

2.1 Interactive Visual Analysis

Selecting and configuring the visualisation: The first step is to find an appropriate visual representation for the given dataset. Within the 10 supported

⁷ <http://open-data.europa.eu>

charts only those are made available which can actively and meaningfully be used with the provided data. For example, the geo-chart is only available if the data contains a geographic dimension. After the chart was selected, the user can adjust the mapping of data onto the visual properties (e.g. axes, colours, item sizes etc.) of the chart, whereby only suitable mappings are offered. Chart selection and the data mapping is computed by an algorithm [3] comparing the semantic information in the RDF Data Cube with the visual capabilities of the chart, which are described using the Visual Analytics Vocabulary⁸.

Aggregation: We provide a dialogue for aggregating the data and creating a new Data Cube. In the scenario shown in Fig 1 the second dataset was averaged over the years and visualised over the countries. Using semantics we differentiate between dimensions and measures and enable validation of the user choices.

For suggesting charts and supporting aggregation we are utilising RDF datatype, occurrence and persistence.

Brushing and Linking: The idea behind brushing and linking is to combine different visualisations to overcome the shortcomings of single techniques [2]. Interactive changes made in one visualisation are automatically reflected in the other ones. Our scenario contains two separate datasets: the first dataset has the dimensions “country” and “year”, the second dataset has only “country”. For conventional tools it is hard to provide interaction over different datasets, because relationships between them are usually not explicitly available. In cases when columns are labelled using equal strings guessing the relationships may be possible, but when labels differ, e.g. a dimension in dataset A is called “Country” while in the dataset B it is called “State”, the relation cannot be established. In such cases the burden of understanding the structure of the datasets and linking them together falls on the user. Within RDF Data Cubes, each dimension has an URI which is (by definition) unique and can be used to establish the connection between datasets, making linking and brushing over different datasets possible.

Applied to our scenario the following interactive analysis is performed (see Fig. 1): The user applies a brush on the first dataset by selecting a specific value range in the “3G coverage” dimension using the parallel coordinates chart. Countries outside of the selected range are greyed out in the geo-chart, which shows the second dataset (SIM card penetration). Obviously, a high 3G coverage correlates with high SIM card penetration (red), with one exception - France.

It should be noted that the functionality of linking data over different datasets, or even different endpoints, depends on the quality of the semantic information: the URIs of the cube-dimensions in different datasets need to be consistent. If datasets use different, domain-specific URI namespaces, linking the data will not be possible.

3 Evaluation

We conducted a formative evaluation to explore if our goals regarding the usability of the Vis Wizard could be achieved and to ascertain that users were able

⁸ <http://code.know-center.tugraz.at/static/ontology/visual-analytics.owl>

to analyse complex datasets. Eight test users participated which executed six tasks, where one task was exclusively about linking and brushing. Test users had a good knowledge of computers, but were not familiar with semantic data. We conducted a quantitative subjective workload test, using the simplified NASA R-TLX, and a qualitative thinking aloud test. More details on the evaluation, including methodology, test users and results are available under [4].

The functionality supporting the choice and configuration of the visualisation was much appreciated, but users pointed out that immediately suggesting the most suitable visualisation would have been even more helpful. The task regarding brushing in the scatterplot had a very high subjective performance of accomplishing (the median was 91.25 on a scale from 0 to 100, 100 being the highest value achievable). The conclusion of our evaluation is that, while several usability issues still need to be fixed, the overall advantage is clearly observable.

4 Conclusion and Future Work

Within this research we have observed a high potential in using semantic information for improving interaction in visual analytics. It has been shown that the user supporting techniques were helpful in gaining insights from the data, without spending much time in selecting and configuring visualisations or analysing how to link the datasets manually.

As for our purpose the correctness of the semantic annotations of the data is essential, the stability of our approach could be improved by implementing the use of URI aliases. We will also explore the possibilities to rank the visualisations in order to, given a particular dataset, automatically show the most suitable one.

Acknowledgements. This work is funded by the EC FP7 projects CODE (grant 296150) and EEX-CESS (grant 600601). The Know-Center GmbH is funded by Austrian Federal Government within the Austrian COMET Program, managed by the Austrian Research Promotion Agency (FFG).

References

1. Cammarano, M., Dong, X.L., Chan, B., Klingner, J., Talbot, J., Halevey, A., Hanrahan, P.: Visualization of heterogeneous data. In: IEEE Information Visualization
2. Keim, D.A.: Information visualization and visual data mining. In: IEEE Transactions on Visualization and computer graphics (2002)
3. Mutlu, B., Höfler, P., Tschinkel, G., Veas, E.E., Sabol, V., Stegmaier, F., Granitzer, M.: Suggesting visualisations for published data. In: Proceedings of IVAPP 2014. pp. 267–275 (2014)
4. Sabol, V., Tschinkel, G., Veas, E., Hoefler, P., Mutlu, B., Granitzer, M.: Discovery and visual analysis of linked data for humans. In: Accepted for publication at the 13th International Semantic Web Conference (2014)
5. Salas, P.E., Martin, M., Mota, F.M.D., Breitman, K., Auer, S., Casanova, M.A.: Publishing statistical data on the web. In: Proceedings of 6th International IEEE Conference on Semantic Computing. IEEE 2012, IEEE (2012)
6. Stolte, C., Hanrahan, P.: Polaris: A system for query, analysis and visualization of multi-dimensional relational databases. IEEE Transactions on Visualization and Computer Graphics 8, 52–65 (2002)

Exploiting Linked Data Cubes with OpenCube Toolkit

Evangelos Kalampokis^{1,2}, Andriy Nikolov³, Peter Haase³, Richard Cyganiak⁴,
Arkadiusz Stasiewicz⁴, Areti Karamanou^{1,2}, Maria Zotou^{1,2}, Dimitris Zeginis^{1,2},
Efthimios Tambouris^{1,2}, Konstantinos Tarabanis^{1,2}

¹ Centre for Research & Technology - Hellas, 6th km Xarilaou-Thermi, 57001, Greece

² University of Macedonia, Egnatia 156, 54006 Thessaloniki, Greece
{ekal, akarm, mzotou, zegin, tambouris, kat}@uom.gr

³ fluid Operations AG, Alttrottstraße 31, 69190 Walldorf, Germany
{andriy.nikolov, peter.haase}@fluidops.com

⁴ Insight Centre for Data Analytics, Galway, Ireland
{richard.cyganiak, arkadiusz.stasiewicz}@insight-centre.org

Abstract. The adoption of the Linked Data principles and technologies has promised to enhance the analysis of statistics at a Web scale. Statistical data, however, is typically organized in data cubes where a numeric fact (aka measure) is categorized by dimensions. Both data cubes and linked data introduce complexity that raises the barrier for reusing the data. The majority of linked data tools are not able to support or do not facilitate the reuse of linked data cubes. In this demo we present the OpenCube Toolkit that enables the easy publishing and exploitation of linked data cubes using visualizations and data analytics.

Keywords: Linked data, statistics, data cubes, visualization, analytics.

1 Introduction

A major part of Open Data concerns statistics such as population figures, economic and social indicators. Analysis of statistical open data can provide value to both citizens and businesses in various areas such as business intelligence, epidemiological studies and evidence-based policy-making. Linked Data has emerged as a promising paradigm to enable the exploitation of the Web as a platform for data integration. As a result Linked Data has been proposed as the most appropriate way for publishing open data on the Web. Statistical data needs to be formulated as RDF data cubes [1] characterized by dimensions, slices and observations in order to unveil its full potential and value [2]. Processing of linked statistical data has only become a popular research topic in the recent years. Several practical solutions have been developed in this domain: for example, the LOD2 Statistical Workbench¹ brings together components developed in the LOD2 project by means of the OntoWiki² tool.

¹ <http://wiki.lod2.eu/display/LOD2DOC/LOD2+Statistical+Workbench>

² <http://aksw.org/Projects/OntoWiki.html>

In this demo paper we describe the OpenCube Toolkit that enable users to work with linked data cubes in an easy manner. In comparison with existing tools, our toolkit provides the following contributions:

- application development SDK allowing customized domain-specific applications to be built to support various use cases;
- new functionalities enabling users to better exploit linked data cubes;
- components supporting the whole linked data cube lifecycle.

2 OpenCube Toolkit

The OpenCube Toolkit³ integrates a number of components which enable the user to work with semantic statistical data at different stages of the lifecycle: from importing legacy data and exposing it as linked open data to applying advanced visualization techniques and complex statistical methods to it.

The Information Workbench (IWB) platform [3] serves as a backbone for the toolkit components. The components are integrated into a single architecture via standard interfaces provided by the IWB SDK: *widgets* (for UI controls) and *data providers* (for data importing and processing components). The overall UI design is based on the use of wiki-based templates providing dedicated views for RDF resources: an appropriate view template is applied to an RDF resource based on its type. All components of the architecture share the access to a common RDF repository (local or remote) and can retrieve data by means of SPARQL queries.

The OpenCube Toolkit demo uses datasets from the Linked Data version of Eurostat⁴ and can be currently accessed using the following link: <http://data.fluidops.net>.

2.1 Using the OpenCube Toolkit for data import, transformation, and publishing

Much of the relevant valuable statistical data are only available in various legacy formats, such CSV and Excel. To present these data in the form of linked RDF data cubes, they have to be imported, transformed into the RDF Data Cube format and made accessible for querying.

The **OpenCube TARQL**⁵ component enables cubes construction from legacy data via TARQL (Transformation SPARQL): a SPARQL-based data mapping language that enables conversion of data from RDF, CSV, TSV and JSON (and potentially XML and relational databases) to RDF. TARQL is tool for converting CSV files to RDF using SPARQL 1.1 syntax. It is built on top of Apache ARQ⁶. The OpenCube TARQL component includes the new release of TARQL. It brings several improvements, such as: streaming capabilities, multiple query patterns in one

³ <http://opencube-toolkit.eu>

⁴ <http://eurostat.linked-statistics.org>

⁵ <https://github.com/cygri/tarql>

⁶ <http://jena.apache.org/documentation/query/>

mapping file, convenient functions for typical mapping activities, validation rules included in mapping file, increased flexibility (dealing with CSV variants like TSV).

The R2RML⁷ language is a W3C standard for mappings from relational databases to RDF datasets. D2RQ⁸ is a platform for accessing relational databases as virtual, read-only RDF graphs. **D2RQ Extensions for Data Cube** cover the functionality of importing of raw data as data cubes by mapping raw data to RDF. The process of mapping the data cube with a relational data source includes: (a) mapping the tables to classes of entities, (b) mapping selected columns into cube dimensions and cube measures, (c) mapping selected rows into observation values, and (d) generate triples with data structure definition. The user, by providing information about the dataset, such as the data dimensions and related measures, will receive an R2RML mapping file, which as a result will be used to generate and store the output.

2.2 Using the OpenCube Toolkit to utilize statistical data

To make use of available statistical data cubes, the user requires, as a minimum, to be able to explore and, visualize the data. The next step involves being able to apply to these data relevant statistical analysis methods.

The **OpenCube Browser** enables the exploration of an RDF data cube by presenting two-dimensional slices of the cube as a table. Currently browser enables users to change the two dimensions that define the table of the browser and also change the values of the fixed dimensions and thus select a different slice to be presented. Moreover, the browser supports roll-up and drill-down OLAP operations through dimensions reduction and insertion respectively. Finally, the user can create and store a two-dimensional slice of the cube based on the data presented in the browser. Initially, the browser selects two dimensions to present in the table and sets up a fixed value for all other dimensions. Based on these it creates and sends a SPARQL query to the store to retrieve the appropriate data. For the drill-down and roll-up operations the browser assumes that a set of data cubes has been created out of the initial cube by summarizing observations across one or more dimensions. We assume that these cubes define an Aggregation Set.

The **OpenCube Map View** enables the visualization of RDF data cubes on a map based on their geospatial dimension. Initially, Map View presents to the user the supported types of visualization (including markers, bubbles, choropleth and heat maps) along with all the dimensions and their values in drop-down lists.

The user selects the type of visualization and a map appears that actually visualizes a one-dimension slice of the cube where the geospatial dimension is free and the other dimensions are randomly “fixed”. In addition, the user can click on an area or marker or bubble and see the details of the specific observation. The maps are created using OpenStreetMap⁹ and Leaflet¹⁰ open-source library.

⁷ <http://www.w3.org/TR/r2rml/>

⁸ <http://d2rq.org/>

⁹ <http://wiki.openstreetmap.org/wiki/Develop>

¹⁰ <http://leafletjs.com/>

To allow the user explore the data in a data cube, it is important that the used visualization controls are (i) interactive and (ii) adapted to the cube data representation. In this way the user can easily switch between different slices of the cube and compare between them. To this end, we implemented our **Chart-based Visualization** functionality. The charts can be inserted into a wiki page of an RDF resource and configured to show data cube slices. When viewing the page, the user can change the selection of dimension values to change the visualised cube slices. The SPARQL query to retrieve the appropriate data is constructed based on the slice definition, and the data is downloaded from the SPARQL endpoint dynamically.

When working with statistical data, a crucial requirement is the possibility to apply specialized analysis methods. One of the most popular environments for statistical data analysis is R¹¹. To use the capabilities of R inside the OpenCube Toolkit, we integrated it with our architecture through the **Statistical Analysis of RDF Data Cubes** component. R is run as a web service (using Rserve¹² package) and accessed via HTTP. Input data are retrieved using SPARQL queries and passed to R together with an R script. Then, R capabilities can be exploited in two modes: (i) as a widget (the script generates a chart, which is then shown on the wiki page) and (ii) as a data source (the script produces a data frame, which is then converted to RDF using defined R2RML mappings and stored in the data repository).

3 Conclusions

This demo paper presents the first release of the OpenCube Toolkit developed to enable easy publishing and reusing of linked data cubes. The toolkit smoothly integrates separate components dealing with different subtasks of the linked statistical data processing workflow to provide the user with a rich set of functionalities for working with statistical semantic data.

Acknowledgments. The work presented in this paper was partially carried out in OpenCube¹³ project, which is funded by the EC within FP7 (No. 611667).

References

1. Cyganiak, R., Reynolds, D.: The RDF Data Cube vocabulary, <http://www.w3.org/TR/vocab-data-cube/> (2013)
2. Kalampokis, E., Tambouris, E., Tarabanis, K.: Linked Open Government Data Analytics. In: Wimmer, M.A., Janssen, M., Scholl, H.J. (eds.) EGOV 2013. LNCS, vol. 8074, pp. 99-110. IFIP International Federation for Information Processing (2013)
3. Haase, P., Schmidt, M., Schwarte, A. Information Workbench as a Self-Service platform. COLD 2011, ISWC 2011, Shanghai, China (2011).

¹¹ <http://www.r-project.org/>

¹² <http://www.rforge.net/Rserve/>

¹³ <http://www.opencube-project.eu>

Detecting Hot Spots in Web Videos

José Luis Redondo García¹, Mariella Sabatino¹,
Pasquale Lisena¹, Raphaël Troncy¹

EURECOM, Sophia Antipolis, France,
{redondo, mariella.sabatino, pasquale.lisena, raphael.troncy}@eurecom.fr

Abstract. This paper presents a system that detects and enables the exploration of relevant fragments (called Hot Spots) inside educational online videos. Our approach combines visual analysis techniques and background knowledge from the web of data in order to quickly get an overview about the video content and therefore promote media consumption at the fragment level. First, we perform a chapter segmentation by combining visual features and semantic units (paragraphs) available in transcripts. Second, we semantically annotate those segments via Named Entity Extraction and topic detection. We then identify consecutive segments talking about similar topics and entities that we merge into bigger and semantic independent media units. Finally, we rank those segments and filter out the lowest scored candidates, in order to propose a summary that illustrates the Hot Spots in a dedicated media player. An online demo is available at <http://linkedtv.eurecom.fr/mediafragmentplayer>.

Keywords: Semantic Video Annotation, Media Fragments, Summarization

1 Introduction

Nowadays, people consume all kind of audiovisual content on a daily basis. From breaking news to satiric videos, personal recordings or cooking tutorials, we are constantly feed by video content to watch. A common practice by viewers consists in fast browsing through the video, using sometimes the key frames provided by the video sharing platform, with the risk of missing the essence of the video. This phenomena is even more obvious when it comes to educational web content. A study made over media entertainment streaming services reveals that the majority of partial content views (52.55%) are ended by the user within the first 10 minutes, and about 37% of these sessions do not last past the first five minutes [5]. In practice, it is difficult and time consuming to manually gather video insights that give the viewers a fair understanding about what the video is talking about. Our research tackles this problem by proposing a set of automatically annotated media fragments called Hot Spots, which intend to highlight the main concepts and topics discussed in a video. We also propose a dedicated exploring interface that eases the consumption and sharing of those hot spots.

The challenge of video segmentation has been addressed by numerous previous research. Some of them rely exclusively on low-level visual features such as color histograms or visual concept detection clustering operations [4]. Other approaches rely on text, leveraging the video transcripts and sometimes manual annotations and comments attached to the video [6] while the combination of both text and visual features is explored in [1]. Our approach combines also both visual and textual features with the added value of leveraging structured knowledge available in the web of data.

2 Generating and Exploring Hot Spots in Web Videos

This demo implements a multimodal algorithm for detecting and annotating the key fragments of a video in order to propose a quick overview about what are the main topics being discussed. We conduct an experiment over a corpora of 1681 TED talks ¹, a global set of conferences owned by the private non-profit Sapling Foundation under the slogan: "Ideas Worth Spreading"

2.1 Media Fragments Generation

First, we perform *shot* segmentation for each video using the algorithm described in [3]. Shots are the smallest unit in a video, capturing visual changes between frames but not necessary reflecting changes of topic being discussed in the video. Therefore, we introduce the notion of *chapters* corresponding to wider chunks illustrating particular topics. In order to obtain such fragments, we use specific marks embedded in the available video transcripts for all TED talks that indicate the start of new paragraphs. In a last step, those fragments are combined with visual shots. Hence, we adjust the boundaries of each chapter using both paragraph and shot boundaries.

2.2 Media Fragments Annotation

We rely on the subtitles available for the 1681 TED talks for annotating the media fragments which have been generated. More precisely, we detect topics and named entities. For the former, we have used the dedicated TextRazor topic detection method², while for the latter, we used the NERD framework [2]. Both entities and topics come with a relevance score which we use to give a weight to this particular semantic unit within the context of the video story. Topics and named entities are attached to a chapter.

2.3 Hot Spots Generation

Once all chapters are delimited and annotated, we iteratively cluster them, in particular, when temporally close segments are similar enough in terms of topics named entities. More precisely, we compute a similarity function between

¹ <http://www.ted.com/>

² <https://www.textrazor.com/documentation>

consecutive pairs of segments S_1 and S_2 until no new merges are possible. This comparison leverages on the annotations attached to each segment by analyzing the number of coincidences between topics $T = \max_3 \left\{ \sum_{topic_i} Rel_i \right\}$ and entities $E = \max_{5W's} \left\{ \sum_{entity_i} Rel_i \right\}$, where Rel_i is the TexRazor’s relevance:

$$d(S_1, S_2) = w_{topic} \cdot \left(\frac{|T_1 \cap T_2|}{\max\{|T_1|, |T_2|\}} \right) + w_{entity} \cdot \left(\frac{|E_1 \cap E_2|}{\max\{|E_1|, |E_2|\}} \right) \quad (1)$$

After this clustering process, the video is decomposed into less but longer chapters. However, there are still too many candidates to be proposed as Hot Spots. Therefore, we filter out those fragments which contain potentially less interesting topics. We define a function for measuring the interestingness of a video segment, which directly depends on the relevance and frequency of the annotations and which is inversely proportional to its length. In our current approach, the Hot Spots are those fragments whose relative relevance falls under the first quarter of the final score distribution.

In a last step, for each Hot Spot, we also generate a summarization to be shown in a dedicated media player where we highlight the main topics T and entities E which have been discovered.

2.4 Exploring Hot Spots within TED Talks

The Hot Spots and their summaries are visualized in a user friendly Media Fragment URI compliant media player. The procedure to get the Hot Spots for a particular Ted talk is the following: the user enters a valid TED Talk URL to get a landing page (Figure 1a). When the results are available, the hot spots are highlighted on the timeline together with the label of the most relevant chapter annotation (Figure 1b). This label can be extended to a broader set of entities and topics (Figure 1c). Finally, the user can always share those hot spots segments using media fragment URIs (Figure 1d).

3 Discussion

We have presented a demo for automatically discovering Hot Spots in online and educational videos. We leverage on visual analysis and background knowledge available in the web of data for detecting what fragments illustrate the best the main topics discussed in the video. Those Hot Spots allow the viewer to quickly decide if a video is worth watching and will provide incentive for consuming videos at the fragment level. In addition, Hot Spots can be explored in a dedicated media fragment player which also display the attached semantic annotations.

We plan to carry out an exhaustive evaluation of our approach involving real users feedback, in order to optimize the results of our Hot Spot detection algorithm and to improve the usability and efficiency of the developed interface.



Fig. 1: Visualizing the Hot Spots of a TED Talk (available at <http://linkedtv.eurecom.fr/mediafragmentplayer/video/bbd70fff-e828-4db5-80d0-1a4c9aea430e>)

We also plan to further exploit the segmentation results and their corresponding annotations for establishing links between fragments belonging to different videos in order to generate true hyperlinks within a closed collection such as TED talks and make results available following Linked Data principles.

References

1. S.-F. Chang, R. Manmatha, and T.-S. Chua. Combining text and audio-visual features in video indexing. In *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'05)*, 2005.
2. G. Rizzo and R. Troncy. NERD: A Framework for Unifying Named Entity Recognition and Disambiguation Extraction Tools. In *13th Conference of the European Chapter for Computational Linguistics (EACL'12)*, Avignon, France, 2012.
3. P. Sidiropoulos, V. Mezaris, I. Kompatsiaris, H. Meinedo, M. Bugalho, and I. Trancoso. Temporal video segmentation to scenes using high-level audiovisual features. *IEEE Transactions on Circuits and Systems for Video Technology*, 21(8):1163–1177, 2011.
4. C. G. Snoek and M. Worring. Multimodal video indexing: A review of the state-of-the-art. *Multimedia tools and applications*, 25(1):5–35, 2005.
5. H. Yu, D. Zheng, B. Y. Zhao, and W. Zheng. Understanding user behavior in large-scale video-on-demand systems. In *1st ACM SIGOPS/EuroSys European Conference on Computer Systems*, pages 333–344, 2006.
6. Z.-J. Zha, M. Wang, J. Shen, and T.-S. Chua. Text mining in multimedia. In *Mining Text Data*, pages 361–384. Springer, 2012.

EUROSENTIMENT: Linked Data Sentiment Analysis

J. Fernando Sánchez-Rada¹, Gabriela Vulcu², Carlos A. Iglesias¹, and Paul Buitelaar²

¹ Dept. Ing. Sist. Telemáticos, Universidad Politécnica de Madrid,
{jfernando,cif}@gsi.dit.upm.es,
<http://www.gsi.dit.upm.es>

² Insight, Centre for Data Analytics at National University of Ireland, Galway
{gabriela.vulcu,paul.buitelaar}@insight-centre.org,
<http://insight-centre.org/>

Abstract. Sentiment and Emotion Analysis strongly depend on quality language resources, especially sentiment dictionaries. These resources are usually scattered, heterogeneous and limited to specific domains of application by simple algorithms. The EUROSENTIMENT project addresses these issues by 1) developing a common language resource representation model for sentiment analysis, and APIs for sentiment analysis services based on established Linked Data formats (lemon, Marl, NIF and ONYX) 2) by creating a Language Resource Pool (a.k.a. LRP) that makes available to the community existing scattered language resources and services for sentiment analysis in an interoperable way. In this paper we describe the available language resources and services in the LRP and some sample applications that can be developed on top of the EUROSENTIMENT LRP.

Keywords: Language Resources, Sentiment Analysis, Emotion Analysis, Linked Data, Ontologies

1 Introduction

This paper reports our ongoing work in the European R&D project EUROSENTIMENT, where we have created a multilingual Language Resource Pool (LRP) for Sentiment Analysis based on a Linked Data approach for modelling linguistic resources.

Sentiment Analysis requires language resources such as dictionaries that provide a sentiment or emotion value to each word. Just as words have different meanings in different domains, the associated sentiment or emotion also varies. Hence, every domain has its own dictionary. The information about what each domain represents or how the entries for each domain are related is usually undocumented or implied by the name of each dictionary. Moreover, it is common that dictionaries from different providers use different representation formats. Thus, it is very difficult to use different dictionaries at the same time.

In order to overcome these limitations, we have defined a Linked Data Model for Sentiment and Emotion Analysis, which is based on the combination of several vocabularies: the NLP Interchange Format (NIF) [1], to represent information about texts, referencing text in the web with unique URIs; the Lexicon Model for Ontologies (lemon) [2], to provide lexical information, and differentiate between different domains and senses of a word; Marl [5], to link lexical entries or senses with a sentiment; and Onyx [3], that adds emotive information.

The use of a semantic format not only eliminates the interoperability issue, but it also makes information from other Linked Data sources available for the sentiment analysis process. The EUROSENTIMENT LRP generates language resources from legacy corpora, linking them with other Linked Data sources, and shares this enriched version with other users.

In addition to language resources, the pool also offers access to sentiment analysis services with a unified interface and data format. This interface builds on the NIF Public API, adding several extra parameters that are used in Sentiment Analysis. Results are formatted using JSON-LD and the same vocabularies as for language resources. The NIF-compatible API allows for the aggregation of results from different sources.

The project documentation³ contains further information about the EUROSENTIMENT format, APIs and tools.

2 Language Resources

The EUROSENTIMENT LRP contains a set of language resources (lexicons and corpora). The available EUROSENTIMENT language resources can be found here.⁴ The user can see the domain and the language of each language resource. At the moment the LRP contains resources for electronics and hotel domains in six languages (Catalan, English, Spanish, French, Italian and Portuguese) and we are currently working on adding more language resources from other domains like telco, movies, food and music. Table 1 shows the number of reviews in each available corpus and the number of lexical entries in each available lexicon.

A detailed description of the methodology for creating the domain-specific sentiment lexicons and corpora to be added in the EUROSENTIMENT LRP was presented at LREC 2014 [4].

The EUROSENTIMENT demonstrator⁵ shows how users can benefit from the LRP, including an interactive SPARQL query editor to access the resources and a faceted browser.

3 Sentiment Services

In addition to a model for language resources, EUROSENTIMENT also provides an API for sentiment and emotion analysis services. Several already existing ser-

³ <http://eurosentiment.readthedocs.org>

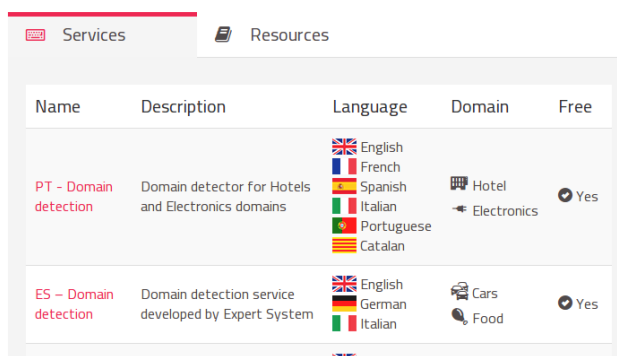
⁴ http://portal.eurosentiment.eu/home_resources

⁵ <http://eurosentiment.eu/demo>

Lexicons			Corpora		
Language	Domains	#Entities	Language	Domains	#Entities
German	General	107417	English	Hotel,Electronics	22373
English	Hotel,Electronics	8660	Spanish	Hotel,Electronics	18191
Spanish	Hotel,Electronics	1041	Catalan	Hotel,Electronics	4707
Catalan	Hotel,Electronics	1358	Portuguese	Hotel,Electronics	6244
Portuguese	Hotel,Electronics	1387	French	Electronics	22841
French	Hotel,Electronics	651			

Table 1. Summary of the resources in the LRP

vices in different languages have been adapted to expose this API. Any user can benefit from these services, which are conveniently listed in the EUROSENTIMENT portal. At the moment, the following services are provided in several languages: language detection, domain detection, sentiment and emotion detection, and text analysis.
















Name	Description	Language	Domain	Free
PT - Domain detection	Domain detector for Hotels and Electronics domains	 English  French  Spanish  Italian  Portuguese  Catalan	 Hotel  Electronics	<input checked="" type="checkbox"/> Yes
ES - Domain detection	Domain detection service developed by Expert System	 English  German  Italian	 Cars  Food	<input checked="" type="checkbox"/> Yes

Fig. 1. The LRP provides a list of available services

4 Applications Using the LRP

To demonstrate the capabilities of the EUROSENTIMENT LRP, we open-sourced the code of several applications that make use of the services and resources of the EUROSENTIMENT LRP. The applications are written in different programming languages and are thoroughly documented. Using these applications as a template, it is straightforward to immediately start consuming the services and resources. The code can be found on the EUROSENTIMENT Github repositories.⁶

⁶ <http://github.com/eurosentiment>



Fig. 2. Simple service that uses the resources in EUROSENTIMENT to analyse opinions in different languages and domains

Acknowledgements

This work has been funded by the European project EUROSENTIMENT under grant no. 296277

References

- Hellmann, S., Lehmann, J., Auer, S., Nitzschke, M.: Nif combinator: Combining nlp tool output. In: Knowledge Engineering and Knowledge Management, pp. 446–449. Springer (2012)
- McCrae, J., Spohr, D., Cimiano, P.: Linking lexical resources and ontologies on the semantic web with lemon. In: Antoniou, G., Grobelnik, M., Simperl, E., Parsia, B., Plexousakis, D., De Leenheer, P., Pan, J. (eds.) The Semantic Web: Research and Applications, Lecture Notes in Computer Science, vol. 6643, pp. 245–259. Springer Berlin Heidelberg (2011)
- Sánchez-Rada, J.F., Iglesias, C.A.: Onyx: Describing emotions on the web of data. In: ESSEM@ AI* IA. pp. 71–82. Citeseer (2013)
- Vulcu, G., Buitelaar, P., Negi, S., Pereira, B., Arcan, M., Coughlan, B., Sánchez-Rada, J.F., Iglesias, C.A.: Generating Linked-Data based Domain-Specific Sentiment Lexicons from Legacy Language and Semantic Resources. In: th International Workshop on EMOTION, SOCIAL SIGNALS, SENTIMENT & LINKED OPEN DATA, co-located with LREC 2014, LREC2014, Reykjavik, Iceland (May 2014)
- Westerski, A., Iglesias, C.A., Tapia, F.: Linked Opinions: Describing Sentiments on the Structured Web of Data. In: Proceedings of the 4th International Workshop Social Data on the Web (2011)

Property-based typing with LITEQ

Programming access to weakly-typed RDF data

Stefan Scheglmann¹, Martin Leinberger¹, Ralf Lämmel², Steffen Staab¹, Matthias Thimm¹, Evelyne Viegas³

¹Institute for Web Science and Technologies, University of Koblenz-Landau, Germany

²The Software Languages Team, University of Koblenz-Landau, Germany

³Microsoft Research Redmond, US

Abstract. Coding against the semantic web can be quite difficult as the basic concepts of RDF data and programming languages differ greatly. Existing mappings from RDF to programming languages are mostly schema-centric. However, this can be problematic as many data sources lack schematic information. To alleviate this problem, we present a data centric approach that focuses on the properties of the instance data found in RDF and that lets a developer create types in his programming language by specifying properties that need to be present. This resembles a type definition rooted in description logics. We show how such a type definition can look like and demonstrate how a program using such type definitions can be written.

1 Introduction

Access to RDF data from within programs is difficult to realize since, (i) RDF follows a flexible and extensible data model, (ii) schema is often missing or incomplete, and (iii) data RDF type information is missing. In order to establish a robust access from a program to RDF data, a developer faces several challenges. Most of the data sources are defined externally and the developer has only a brief idea of what to find in this data source, therefore he has to explore the data first. Once explored, he has to deal with the impedance mismatch between how RDF types are structuring RDF data, compared to how code types are used in programming languages, [4, 2, 6, 1]. In response to these challenges, we have proposed LITEQ [5], a system that allows for the mapping of RDF schema information into programming language types.

However, using LITEQ in practice has shown that purely relying on RDF schema information for the mapping raises new issues. To alleviate these problems, we have implemented an property-based approach and include it into LITEQ as an alternative way of usage. Using this, a developer is able to create code types by listing properties that should be present in instances of this code type.

In this demo, we present an implementation of this approach¹ in F#. It supports developers in defining new code types by specifying their properties. This is aided by auto-completion mechanism, which computes its suggestions directly on the instance data without any need of a schema. The approach is intended as an extension to the LITEQ library also presented at the InUse-Track of ISWC 2014.

¹ <http://west.uni-koblenz.de/Research/systems/liteq>

2 The LITEQ approach

Typically, the integration of RDF data in a programming environment is a multi-step process. (1) The structure and content of the data source has to be explored, (2) the code types and their hierarchy has to be designed and implemented, then (3) the queries for the concrete data have to be defined, and finally (4) the data can be retrieved and mapped to the predefined code types.

The NPQL Approach: LITEQ provides an IDE integrated workflow to cope with all of these tasks. It implements NPQL, a novel path query language, to explore the data source, to define types based on schematic information and to query the data. Returned results of these queries are automatically typed in the programming language. All of this is aided by the autocompletion of the IDE. Figure 1 shows a typical LITEQ expression using an NPQL query in order to retrieve all `mo:MusicArtist` entities which actually have the `foaf:made` and `mo:biography` property defined. The result is returned as a set of objects of the created code type for `mo:MusicArtist`.

```
// Retrieve all musicArtists that have made something and that have a biography
let musicArtists = Store.NPQL().`mo:MusicArtist`.`<-`.`foaf:made`
    .`<-`.`mo:biography`.Extension
```

Fig. 1: Querying for all music artists that made at least one record and have a biography.

Property-based Type access with LITEQ: The example shown above has several problems. Additionally, schema-centric technique only provides code types for which a RDF type is defined in the schema. It is not possible to introduce more specific code types, e.g. if it is known that all entities of interest will have at least one `foaf:made` and `mo:biography` relation, one may like to reflect that by the returned code type. Lastly and most importantly, for all schema-centric access methods, like LITEQ, a more or less complete schema must be present, which is not always the case, especially in Linked Data. To cope with these problems of schema-centric approaches, we introduce the idea of a different data access/typing approach, property-based RDF access in a program [7].

Property-based Type declaration: The basic idea is very simple: (1) A code type is defined by a set of properties. (2) Extensionally, a code type represents a anonymous RDF type (or view on the data) which refers to the set of all entities which actually have all the properties defined in the code type. (3) Intensionally, the code type signature is given by the set of properties (for these it provides direct access). All other properties of an instance of this type can only be accessed indirectly in a program using the predicate name. Specifically, this means that a developer might define a type by just declaring a set of properties as the types signature (*Sig*), e.g. $Sig = \{foaf:made, mo:biography\}$. Our approach allows for two different ways to map such a property-based type to a corresponding code type. A flat-mapping which just maps to a `rdf:Resource` representation and only makes the properties explicitly accessible which are defined in the code type signature. Such an unnamed code type refers to the set of all entities sharing the properties in the code types signature (*Sig*), cf. 1.

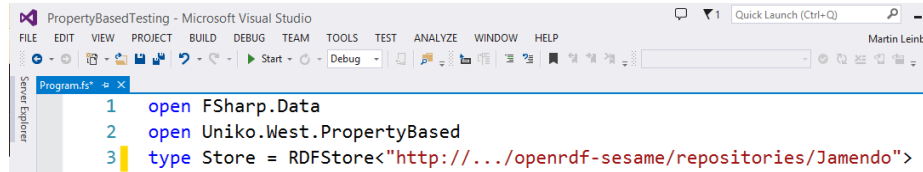
$$\text{unnamedType}_{Sig} \equiv \exists foaf:made \sqcap \exists mo:biography \quad (1)$$

The second option is named-mapping. Which allows to map the previously unnamed type to a given RDF type. This ensures separation if entities of distinguished types share properties, e.g. music artists and producers given the properties `foaf:made` and `mo:biography`. And to search for other properties stating that their domain is the specified type in order to provide a richer API. The named type is defined as the intersection of the unnamed type for the provided signature (*Sig*) and all entities of the given RDF type (`mo:MusicArtist`), cf. 2.

$$\text{namedType}_{(Sig, mo:MusicArtist)} \equiv \text{unnamedType}_{Sig} \sqcap mo:MusicArtist \quad (2)$$

Type-declaration in Practice In the following, we will give a brief overview of the new method of access RDF data in a programming environment:

(1) To use the library, the DLL must be referenced from the project. This allows opening the library namespace and creating a store object by passing a URL to a SPARQL endpoint, cf. Figure 2.



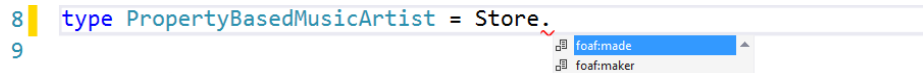
```

1 open FSharp.Data
2 open Uniko.West.PropertyBased
3 type Store = RDFStore<"http://.../openrdf-sesame/repositories/Jamendo">

```

Fig. 2: Creating the type representing the store.

(2) All properties contained in the SPARQL endpoint can be accessed from the store object (cf. Fig. 3).



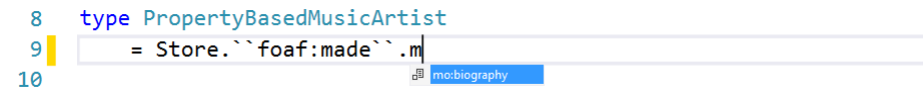
```

8 type PropertyBasedMusicArtist = Store.
9

```

Fig. 3: Defining types via property selection.

(3) Once a property has been chosen, the set of properties presented in the next step is restricted to those properties which have been observed in combination of the previously chosen properties, cf. Figure 4. Here `foaf:made` has already been chosen and the only co-occurring property `mo:biography` is presented for the next selection.



```

8 type PropertyBasedMusicArtist
9 = Store.`foaf:made`.m
10

```

Fig. 4: Refining a type by adding additional property constraints.

(4) Finally, the developer has to decide on how he likes the new type to be mapped. As mentioned in the previous section, he can choose for an unnamed representation or a named one, cf. Figure 5.

(5) If the developer decides for the named representation, he has to choose the RDF type, cf. Figure 6. In this case, only `mo:MusicArtist` instances contain the specified properties and therefore he can only chose this one RDF type.

The presented method of relying on instance information instead of the schema has a severe drawback when it comes to specifying return code types of properties in the programming language. As one cannot use schematic information, the only chance is

```

8 | type PropertyBasedMusicArtist
9 | = Store.`foaf:made`.`mo:biography`.
10 |
11 |

```



Fig. 5: Named and unnamed variants

```

11 | type MusicArtist
12 | = Store.`foaf:made`.`mo:biography`.Named.
13 |

```

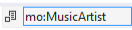


Fig. 6: Defining the named code type based on RDF type mo:MusicArtist.

to probe the instance set. However, for big instance sets, this process takes to long to be useful. The current prototype avoids this problem by only probing whether a property returns another RDF resource or a literal and types returned values accordingly.

3 Conclusion and Further Work

In this extended abstract, we presented a new way to work with RDF in a programming language - by defining types based on their properties. As an extension of LITEQ, this demo will focus on the new feature of property based type definition, as described in [8, 7] and discussed in [3].

Acknowledgements This work has been supported by Microsoft.

References

1. V. Eisenberg and Y. Kanza. Ruby on semantic web. In Serge Abiteboul, Klemens Böhm, Christoph Koch, and Kian-Lee Tan, editors, *ICDE2011*, pages 1324–1327. IEEE Computer Society, 2011.
2. L. Hart and P. Emery. OWL Full and UML 2.0 Compared. <http://uk.builder.com/whitepapers/0and39026692and60093347p-39001028qand00.htm>, 2004.
3. S. Homoceanu, P. Wille, and W. T. Balke. Proswip: Property-based data access for semantic web interactive programming. In *12th International Semantic Web Conference, ISWC 2013*, Sydney, Australia, 2013.
4. A. Kalyanpur, D. J. Pastor, S. Battle, and J. A. Padget. Automatic Mapping of OWL Ontologies into Java. In *SEKE2004*, 2004.
5. M. Leinberger, S. Scheglmann, R. Lämmel, S. Staab, M. Thimm, and E. Viegas. Semantic web application development with LITEQ. In *International Semantic Web Conference*, 2014.
6. T. Rahmani, D. Oberle, and M. Dahms. An adjustable transformation from owl to ecore. In *MoDELS2010*, volume 6395 of *LNCS*, pages 243–257. Springer, 2010.
7. S. Scheglmann and G. Gröner. Property-based Typing for RDF Data. In *PSW 2012, First Workshop on Programming the Semantic Web, Boston, Massachusetts, November 11th, 2012*, 2012.
8. S. Scheglmann, G. Gröner, S. Staab, and R. Lämmel. Incompleteness-aware programming with rdf data. In Evelyne Viegas, Karin Breitman, and Judith Bishop, editors, *DDFP*, pages 11–14. ACM, 2013.

From Tale to Speech: Ontology-based Emotion and Dialogue Annotation of Fairy Tales with a TTS Output

Christian Eisenreich¹, Jana Ott¹, Tonio Süßdorf¹, Christian Willms¹,
Thierry Declerck^{2,1}

¹ Saarland University, Computational Linguistics Department, D-66041 Saarbrücken, Germany

(eisenr|janao|tonios|cwillms)@coli.uni-saarland.de

² German Research Center for Artificial Intelligence (DFKI), Language Technology Lab, Stuhlsatzenhausweg 3, D-66123 Saarbrücken, Germany

thierry.declerck@dfki.de

Abstract. In this demo and poster paper, we describe the concept and implementation of an ontology-based storyteller for fairy tales. Its main functions are (i) annotating the tales by extracting timeline information, characters and dialogues with corresponding emotions expressed in the utterances, (ii) populating an existing ontology for fairy tales with the previously extracted information and (iii) using this ontology to generate a spoken version of the tales.

Common natural language processing technologies and resources, such as part-of-speech tagging, chunking and semantic networks have been successfully used for the implementation of the three tasks mentioned just above, including the integration of an open source text-to-speech system. The code of the system is publicly available.

Keywords: ontology, natural language processing, text-to-speech, semantic-network, fairy tale, storytelling

1 Introduction

The idea of developing an ontology-based storyteller for fairy tales was based on the consideration of two previous works in the field of narrative text processing. The first work is described in (Scheidel & Declerck, 2010), which is about an augmented Proppian¹ fairy tale markup language, called Apftml, which we extended according to the needs of our current work.

Our second starting point is described in (Declerck et al., 2012), which presents an ontology-based system that is able to detect and recognize the characters (participants) playing a role in a folktale. Our system combines and extends the results of

¹ From „Vladimir Yakovlevich Propp”, who was “a Soviet folklorist and scholar who analyzed the basic plot components of Russian folk tales to identify their simplest irreducible narrative elements.” (http://en.wikipedia.org/wiki/Vladimir_Propp)

those studies, adding the detection of dialogues and emotions in the tales and an ontology-driven Text-To-Speech (TTS) component that “reads” the tales, with individual voices for every character, including also a voice for the narrator, and taking into account the types of emotions detected during the textual processing of the tales.

To summarize: Our system first parses the input tale (in English or German) and extracts as much relevant information as possible on the characters – including their emotions -- and the events they are involved in. This provides us with an annotated version of the tale that is used for populating the ontology. The system finally uses the ontology and a robust and parameterizable TTS system to generate the speech output.

All the data of the system have been made available in a bitbucket repository (<https://bitbucket.org/ceisen/apftml2repo>), including documentation and related information².

2 Architecture of the System

Firstly, we use the Python NLTK³ and Pattern API⁴ to annotate the tale. Then we use the Java OWL-API⁵ to populate the ontology. And finally the Mary Text-To-Speech system⁶ is used to generate the speech output. Mary is an open-source, multilingual Text-to-Speech Synthesis platform, which is robust, easy to configure and allows us to extend our storyteller to more languages. The general architecture of the system is displayed below in **Fig. 1**.

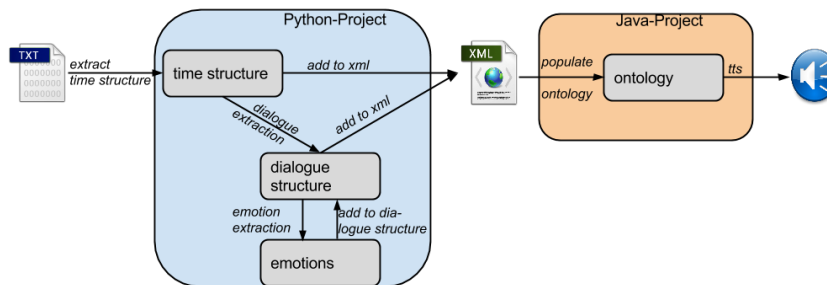


Fig. 1. The general architecture of the ontology-driven “Tale to Speech” system

² An example of the audio data generated for the tale “The Frog Prince” is available at https://bytebucket.org/ceisen/apftml2repo/raw/763c5eb533f09997e757ec61652310c742238384/example%20output/audio_output.mp3.

³ Natural Language Toolkit: <http://www.nltk.org/>. See also (Bird et al., 2009)

⁴ See (De Smedt & Daelemans, 2012).

⁵ See (Horridge & Bechhofer, 2011).

⁶ <http://mary.dfki.de/>. See also (Schröder Marc & Trouvain, 2003) or (Charfuelan & Steiner, 2013).

3 The Ontology Population

The ontology we use is an extension of the one presented in (Declerck et al., 2012), which describes basically family structures among human beings, but also a small list of extra-natural beings. In the extended version of the ontology we include also temporal information (basically for representing the mostly linear structure of the narrative) as well as dialogue structures, including the participants involved in the dialogues (sender(s) and receivers(s)), whereas we give special attention also to the narrator of the tale, since this “character” is also giving relevant information about the status of the characters in the tales, including their emotional state. Dialogues are synchronized with the linear narrative structure. Detected emotions are also included in the populated ontology, and are attached for the time being to utterances, and will be attached in the future to the characters directly. The Mary TTS system is accessing all this information in order to parameterize the voices that are attached to each detected characters.

4 A Gold Standard

In order to support evaluation of the automated annotation of fairy tales with our integrated set of tools 5 fairy tales have been manually annotated⁷. The tales are “The Frog Prince”, “The Town Musicians of Bremen”, “Die Bremer Stadt Musikanten” (the German original version), “The Magic Swan Geese” and “Rumpelstiltskin”.

The annotation examples show the different steps involved in the system: the text analysis, the temporal segmentation, the recognition of the characters and the dialogues they are involved in, the emotions that are attached to the utterances and delivered during speech the story in near real time.

5 Summary and Outlook

We have designed and implemented in the field of fairy tales an ontology-based emotion- and dialogue annotation system with speech output. The system provides robust results for the tested fairy tales. While the annotation and ontology population processes are working for both English and German texts, the TTS output is for the time being optimized for the English language.

Future work can deal with adding a graphical user interface, extending the parsing process for annotating tales in other languages and populating the ontology with more information, like the Proppian functions.

⁷ The manually annotated tales, together with the annotation schema, are available at <https://bitbucket.org/ceisen/apftml2repo/src/763c5eb533f09997e757ec61652310c742238384/soproworkspace/SoPro13Java/gold/?at=master>

6 References

1. Horridge Matthew and Bechhofer Sean (2011). The owl api: A java api for owl ontologies IOS Press, IOS Press volume 2 number 1, 11--12
2. Schröder Marc and Trouvain Jürgen (2003). The German text-to-speech synthesis system MARY: A tool for research, development and teaching. Springer: International Journal of Speech Technology, volume 6 number 4, 365—377.
3. Marcela Charfuelan and Ingmar Steiner (2013). Expressive speech synthesis in MARY TTS using audiobook data and EmotionML. ISCA: Proceedings of Interspeech 2013
4. Steven Bird, Ewan Klein, and Edward Loper (2009). Natural Language Processing with Python--- Analyzing Text with the Natural Language Toolkit.. O'Reilly Media, (<http://www.nltk.org/book/>)
5. Ekman Paul (1999). Emotions In T. Dalgleish and T. Power (Eds.) The Handbook of Cognition and Emotion Pp. 45-60. Sussex, UK: John Wiley & Sons, Ltd.
6. De Smedt, Tom and Daelemans, Walter (2012). Pattern for python. The Journal of Machine Learning Research, volume{13} nr.1 2063--2067
7. Scheidel Antonia and Declerck Thierry (2010). Apftml-augmented proppian fairy tale markup language. First International AMICUS Workshop on Automated Motif Discovery in Cultural Heritage and Scientific Communication Texts.. Szeged University, volume 10
8. Declerck Thierry, Koleva Nikolina and Krieger Hans-Ulrich (2012). Ontology-based incremental annotation of characters in folktales Association for Computational Linguistics. Proceedings of the 6th Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities, 30--34
9. Propp V.Y. *Morphology of the Folktale*. Leningrad, 1928; English: The Hague: Mouton, 1958; Austin: University of Texas Press, 1968.
10. Inderjeet Mani: Computational Modeling of Narrative. Synthesis Lectures on Human Language Technologies, Morgan & Claypool Publishers 2012.

BIOTEX: A system for Biomedical Terminology Extraction, Ranking, and Validation

Juan Antonio Lossio-Ventura¹, Clement Jonquet¹,
Mathieu Roche^{1,2}, and Maguelonne Teisseire^{1,2}

¹University of Montpellier 2, LIRMM, CNRS - Montpellier, France

²Irstea, CIRAD, TETIS - Montpellier, France

juan.lossio@lirmm.fr, jonquet@lirmm.fr,
mathieu.roche@cirad.fr, maguelonne.teisseire@teledetection.fr

Abstract. Term extraction is an essential task in domain knowledge acquisition. Although hundreds of terminologies and ontologies exist in the biomedical domain, the language evolves faster than our ability to formalize and catalog it. We may be interested in the terms and words explicitly used in our corpus in order to index or mine this corpus or just to enrich currently available terminologies and ontologies. Automatic term recognition and keyword extraction measures are widely used in biomedical text mining applications. We present BIOTEX, a Web application that implements state-of-the-art measures for automatic extraction of biomedical terms from free text in English and French.

1 Introduction

Within a corpus, there is different information to represent, with different communities to express that information. Therefore, the terminology and vocabulary is often very corpus specific and not explicitly defined. For instance in medical world, terms employed by lay users on a forum will necessarily differ from the vocabulary used by doctors in electronic health records. We thus intend to offer users an opportunity to automatically extract biomedical terms and use them for any natural language, indexing, knowledge extraction, or annotation purpose. Extracted terms can also be used to enrich biomedical ontologies or terminologies by offering new terms or synonyms to attach to existing defined classes. Automatic Term Extraction (ATE) methods are designed to automatically extract relevant terms from a given corpus.¹ Relevant terms are useful to gain further insight into the conceptual structure of a domain. In the biomedical domain, there is a substantial difference between existing resources (ontologies) in English and French. In English there are about 7 000 000 terms associated with about 6 000 000 concepts such as those in UMLS or BioPortal [7]. Whereas, in French there are only about 330 000 terms associated with about 160 000 concepts [6]. French ontologies therefore have to be populated and tool like BIOTEX will help for this task. Our project involves two main stages: (i) Biomedical term extraction, and (ii) Ontology population, in order to populate ontologies with the extracted terms.

¹ We refer to ATE when terms extracted are not previously defined in existing standard ontologies or terminologies. We refer to 'semantic annotation' when term extracted can be attached or match to an existing class (URI) such as in [8]. Both approaches are related to Named Entity Recognition (NER), which automatically extracts name of entities (disease, person, city).

In this paper, we present BIOTEX, an application that performs the first step. Given a text corpus, it extracts and ranks biomedical terms according to the selected state-of-the-art extraction measure. In addition, BIOTEX automatically validates terms that already exist in UMLS/MeSH-fr terminologies. We have presented different measures and performed comparative assessments in other publications [4, 5]. In this paper, we focus on the presentation of BIOTEX and the use cases it supports.

2 Related work and available extraction measures

Term extraction techniques can be divided into four broad categories: (i) **Linguistic approaches** attempt to recover terms via linguistic patterns [3]. (ii) **Statistical methods** focus on external evidence through contextual information. Similar methods, called Automatic Keyword Extraction (AKE), are geared towards extracting the most relevant words or phrases in a document. These measures, such as *Okapi BM25* and *TF-IDF*, can be used to automatically extract biomedical terms, as we proposed in [4]. These two measures are included in BIOTEX. (iii) **Machine Learning** is often designed for specific entity classes and thus integrate term extraction and term classification. (iv) **Hybrid methods**. Most approaches combine several methods (typically linguistic and statistically based) for the term extraction task. This is the case of *C-value* [2], a very popular measure specialized in multi-word and nested term extraction.

In [4], we proposed the new hybrid measures *F-TFIDF-C* and *F-OCapi*, which combine *C-value* with *TF-IDF* and *Okapi* respectively to extract terms and obtain better results than *C-value*. In [5], we propose *LIDF-value* measure based on linguistic and statistical information. We offer all of these measures within BIOTEX. Our measures were evaluated in terms of *precision* [4, 5] and obtained the best results over the top k extracted terms ($P@k$) on several corpora (LabTestOnline, GENIA, PubMed). For instance, on a GENIA corpus, *LIDF-value* achieved 82% for $P@100$, thus improving the *C-value* precision by 13%, and 66% for $P@2000$, with an improvement of 11%. BIOTEX allows users to assess the performances of measures with different corpora.

A detailed study of related work revealed that most existing systems implementing statistical methods are made to extract keywords and, to a lesser extent, to extract terminology from a text corpus. Indeed, most systems take a single text document as input, not a set of documents (as corpus), for which the *IDF* can be computed. Most systems are available only in English. Table 1 shows a quick comparison with *TerMine* (*C-value*), the most commonly used application, and *FlexiTerm*, the most recent one.

Table 1. Brief comparison of biomedical terminology extraction applications.

	<i>BioTex</i>	<i>TerMine</i>	<i>FlexiTerm</i>
<i>Languages</i>	en/fr	en	en
<i>Type of Application</i>	Desktop/Web	Web	Desktop
<i>License</i>	Open	Open	Open
<i>Processing Capacity</i>	No Limits / < 6 MB	< 2 MB	No Limits
<i>Possibility to save results</i>	XML	-	CSV
<i>POS tool</i>	TreeTagger	Genia/TreeTagger	Stanford POS
<i># of Implemented Measures</i>	8	1	1

<http://www.nactem.ac.uk/software/terminer/>

<http://users.cs.cf.ac.uk/I.Spasic/flexiterm/>

3 Implementation of BIOTEX

BIOTEX is an application for biomedical terminology extraction which offers several baselines and new measures to rank candidate terms for a given text corpus. BIOTEX can be used either as: (i) a Web application taking a text file as input, or (ii) as a Java library. When used as a Web application, it produces a file with a maximum of 1200 ranked candidate terms. Used as a Java library, it produces four files with ranked candidate terms found in the corpus, respectively, unigram, bigram, 3-gram and all the 4+ gram terms. BIOTEX supports two main use cases:

- (1) **Term extraction and ranking measures:** As illustrated by the Web application interface, Figure 1 (1), BIOTEX users can customize the workflow by changing the following parameters:
 - Choose the corpus *language* (i.e., English or French), and the *Part-of-Speech* (PoS) tagger to apply. Note that we tested three POS-tagger tools but currently only TreeTagger is available within BIOTEX.
 - Select a number of *patterns* to filter out the candidate terms (200 by default). Those reference patterns (e.g., noun-noun, noun-prep-noun, etc.) were built with terms taken from UMLS for English and MeSH-fr for French. They are ranked by frequency.
 - Select the *type of terms* to extract: all terms (i.e., single- and multi-word terms) or multi-word terms only.
 - Select the *ranking measures* to apply.
- (2) **Validation of candidate terms:** After the extraction process, BIOTEX automatically validates the extracted terms by using UMLS (Eng) & MeSH-fr (Fr). As illustrated in Figure 1 (2), these validated terms are displayed in green, specifying the used knowledge source and the others in red. Therefore, BIOTEX allows someone to easily distinguish the classes annotating the original corpus (in green) from the terms that maybe also considered relevant for their data, but need to be curated (in red). The last ones may be considered candidates for ontology enrichment.

4 Conclusions and Future Work

In this article, we present the BIOTEX application for biomedical terminology extraction. It is available for online testing and evaluation but can also be used in any program as a Java library (POS tagger not included). In contrast to other existing systems, this system allows us to analyze a French corpus, manually validate extracted terms and export the list of extracted terms. We hope that BIOTEX will be a valuable tool for the biomedical community. It is currently used in a couple of test-beds within the SIFR project (<http://www.lirmm.fr/sifr>). The application is available at <http://tubo.lirmm.fr/biotex/> along with a video demonstration <http://www.youtube.com/watch?v=EBbkZj7HcL8>. For our future validations, we will enrich our validation dictionaries with BioPortal [7] terms for English and CISMef [9] terms for French. In the future, we will offer disambiguation features using the Web to find the context in order to populate biomedical ontologies with the new extracted terms (red terms), while looking into the possibility of extracting relations [1] between new terms and already known terms.

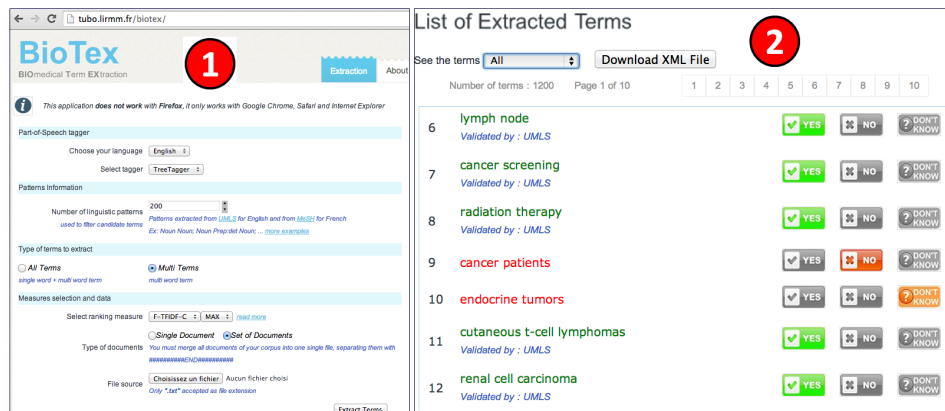


Fig. 1. (1) Interface: term extraction. (2) Interface: term validation. Users can export the results for offline processing.

Acknowledgments. This work was supported in part by the French National Research Agency under JCJC program, grant ANR-12-JS02-01001, as well as by University of Montpellier 2, CNRS, IBC of Montpellier project and the FINCYT program, Peru

References

1. Abacha, A. B., Zweigenbaum, P.: Automatic extraction of semantic relations between medical entities: a rule based approach. *Journal of Biomedical Semantics*, vol. 2 (2011)
2. Frantzi K., Ananiadou S., Mima, H.: Automatic recognition of multiword terms: the C-value/NC-value Method. *International Journal on Digital Libraries*, vol. 3, pp. 115-130, (2000)
3. Gaizauskas, R., Demetriou, G., Humphreys, K.: Term recognition, classification in biological science journal articles. *Proceeding of the Computational Terminology for Medical, Biological Applications Workshop of the 2 nd International Conference on NLP*, pp. 37-44 (2000)
4. Lossio-Ventura, J.A., Jonquet, C., Roche, M., Teisseire M.: Towards a Mixed Approach to Extract Biomedical Terms from Text Corpus. *International Journal of Knowledge Discovery in Bioinformatics, IGI Global*. vol. 4, pp. 1-15, Hershey, PA, USA (2014)
5. Lossio-Ventura, J.A., Jonquet, C., Roche, M., Teisseire M.: Yet another ranking function to automatic multi-word term extraction. *Proceedings of the 9th International Conference on Natural Language Processing (PolTAL'14)*, Springer LNAI. Warsaw, Poland (2014)
6. Neveol, A., Grosjean, J., Darmoni, S., Zweigenbaum, P.: Language Resources for French in the Biomedical Domain. *9th International Conference on Language Resources and Evaluation (LREC'14)*. Reykjavik, Iceland (2014)
7. Noy, N. F., Shah, N. H., Whetzel, P. L., Dai, B., Dorf, M., Griffith, N., Jonquet, C., Rubin, D. L., Storey, M., Chute, C.G., Musen, M. A.: BioPortal: ontologies and integrated data resources at the click of a mouse. *Nucleic acids research*, vol. 37(suppl 2), pp 170–173. (2009)
8. Jonquet, C., Shah, N.H., Youn, C.H., Callendar, Chris, Storey, M-A, Musen, M.A.: NCBO Annotator: Semantic Annotation of Biomedical Data. *8th International Semantic Web Conference, Poster and Demonstration Session Washington DC, USA* (2009)
9. Darmoni, S.J., Pereira, S., Sakji, S., Merabti, T., Prieur, E. Joubert, M., Thirion, B.: Multiple Terminologies in a Health Portal: Automatic Indexing and Information Retrieval. *12th Conference on Artificial Intelligence in Medicine, LNCS 5651*, pp.255-259, Verona, Italy (2009)

Visualizing and Animating Large-scale Spatiotemporal Data with ELBAR Explorer

Suvodeep Mazumdar¹ and Tomi Kauppinen^{2,3}

¹ Department of Computer Science,
University of Sheffield, 1 Portobello, S1 4DP, United Kingdom
`s.mazumdar@sheffield.ac.uk`

² Cognitive Systems Group, University of Bremen, Germany

³ Department of Media Technology, Aalto University School of Science, Finland
`tomi.kauppinen@uni-bremen.de`

Abstract. Visual exploration of data enables users and analysts observe interesting patterns that can trigger new research for further investigation. With the increasing availability of Linked Data, facilitating support for making sense of the data via visual exploration tools for hypothesis generation is critical. Time and space play important roles in this because of their ability to illustrate dynamicity, from a spatial context. Yet, Linked Data visualization approaches typically have not made efficient use of time and space together, apart from typical rather static multivisualization approaches and mashups. In this paper we demonstrate ELBAR explorer that visualizes a vast amount of scientific observational data about the Brazilian Amazon Rainforest. Our core contribution is a novel mechanism for animating between the different observed values, thus illustrating the observed changes themselves.

Keywords: Visual Analytics, Information Visualization, Linked Data

1 Introduction

Making sense of spatiotemporal data is a crucial step in providing insight for critical actionable decisions. Linked Data is no exception in this. With the increased availability of potentially interesting information the task is to support decision makers by illustrating significant patterns of data. This way data becomes narrative and can share a story [3].

In this paper we demonstrate the combined use of spatial and temporal aspects of Linked Data and illustrate how very heterogeneous phenomena can be illustrated over time and space. Our contribution is an explorer that takes spatiotemporal Linked Data as an input via SPARQL queries and enables exploration of variables via animations. To evaluate and illustrate the use of the tool we make use of openly published data about the Brazilian Amazon Rainforest. This data, including its economical, social and ecological dimensions, serves to show the potential of visualizing Linked Data by animations over time.

Section 2 explains both the ELBAR explorer and the spatiotemporal data we used for evaluation. Section 3 discusses the use of ELBAR explorer with a

concrete scenario. Section finishes the paper with concluding remarks and future work ideas.

2 Animating Large-scale Temporal Data with ELBAR

In this demonstration, ELBAR⁴ makes use of the openly available Linked Brazilian Amazon Rainforest Data⁵ [2] which captures and shares environmental observations (like deforestation) together with information about social phenomena (like amount of population) and economical phenomena (like market prices of products). The data has been aggregated to 25 km x 25 km grid cells [1] and extended with open governmental data and linked to DBPedia. ELBAR uses the paradigm of visual animations to illustrate changes over time on maps. The core idea is that employing such means to navigate multiple dimensions of data supports analysts in generating hypotheses for further investigation.

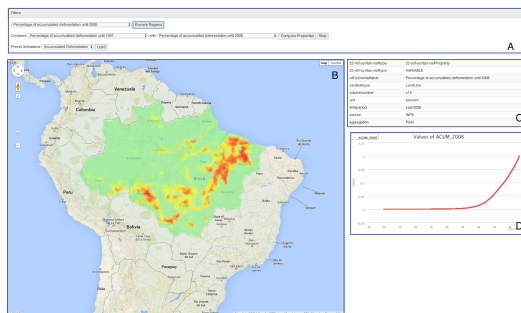


Fig. 1. Explorer for Linked Brazilian Amazon Rainforest (ELBAR). The interface contains four sections: A – Filters, B – Map, C – Info window, D – Graph

Fig. 1 presents a screenshot of the ELBAR explorer. The filters (Section A) provide mechanisms for selecting variables (e.g. deforestation rates) for further inspection. SPARQL queries are built from the interactions done by the users and sent to SPARQL endpoints. Results from the endpoint are processed and converted to visual elements on maps (Section B) and graphs (Section D)⁶. The relevant observations (as retrieved from a triple store) are then visually encoded and overlaid on a map, based on their spatial positions. The information window (Section C) is then updated with further information regarding the filter being selected. Clicking on visual elements of the graph (Section D) highlights the individual sections on the map.

⁴ A demo of ELBAR is presented at <http://linkedscience.org/demos/elbar>

⁵ <http://linkedscience.org/data/linked-brazilian-amazon-rainforest/>

⁶ Users are also presented with Preset Animations, that can be previously defined or extracted from the data

3 Scenario for the Use of ELBAR

3.1 Visualizing the Phenomena

Assuming a user would like to understand a certain phenomena over time (like deforestation) she/he will select a property (like deforestation rate) to visualise it on the map. The explorer then presents the corresponding values of the property, color-coded and overlaid on the map as well as a distribution of the values of the property as a graph. The graph (see the the bottom right of Figure 1) is interactive and supports mouse events such as zoom (by clicking and dragging to select a section in the graph) or left clicks.

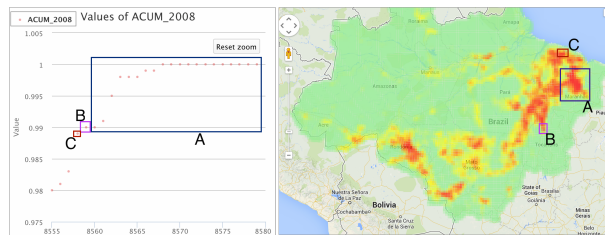


Fig. 2. User interactions on graphs translate to highlight visual elements on the map, aiding in providing context

Zooming the graph provides a finer grained view of the distribution to facilitating selection of individual data points. Clicking on the graph selects the respective section on the map and highlights it. Such interactions support understanding of how different phenomena are distributed. They also potentially illustrate their spatial proximity and patterns they form. The example illustrated in Figure 2 shows the distribution of a property (ACUM_2008 - accumulated percentage of deforestation in The Brazilian Amazon Rainforest until 2008) on the graph (left) and the map (right). The graph is then zoomed to the highest few points and then explored using mouse clicks. Clicking on individual datapoints indicate their spatial references by marking up the respective section on the map. Clicking on all the points (with a high value of ACUM_2008) on Section A (middle) highlights the respective section A on the map (right). However, with almost similar high values, the section B and C highlight sections in other areas.

3.2 Animating the Phenomena over Time

Comparing two properties for example (selected in Section A), accumulated deforestation in 1997 (ACUM_1997) and accumulated deforestation in 2008 (ACUM_2008) in such cases is a helpful feature that provides animated transitions to assist users observe change. The result is then parsed and the values are visually encoded into the relevant sections on the map, with a transition⁷ defined that

⁷ <https://github.com/mbostock/d3/wiki/Transitions>

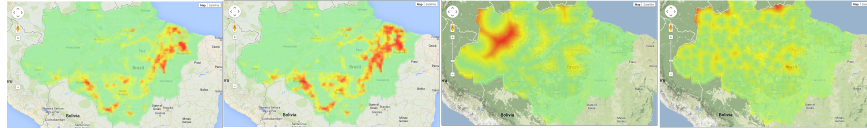


Fig. 3. Generating hypothesis for the high deforestation in the North East Amazon. 1: accumulated deforestation in 1997, 2: accumulated deforestation in 2008, 3: distance from the nearest road, 4: distance from nearest municipality.

iterates between the two visual encodings, thereby creating an animated effect. Since no other visual feature is altered in the user’s field of view apart from color encoding, the user can easily observe the evolution of deforestation. The analyst then attempts to understand why the North East Amazon contains the highest amount of deforestation. Different properties such as distance from nearest road (bottom left, Fig. 3) and distance from the nearest municipality (bottom right, Fig. 3) indicate the clear segregation of the North Eastern region, which could support an analyst to hypothesise that the remoteness of these areas help in illegal deforestation activities.

4 Conclusions

In this paper, we presented ELBAR explorer that employs generic visualization and animation techniques to support analysts and decision makers explore spatial and temporal data. We demonstrate the system and the architecture, along with a description of the data. This will be accompanied with a guided example of how such techniques can be used, and we also invite our demonstration attendees, both online and onsite, to build custom transitions.

Future work will include development and evaluation of ELBAR with different kinds of spatiotemporal data. Moreover, we investigate other novel mechanisms for exploring the spatiotemporal and topical aspects of Linked Data. We foresee the potential for the use of the community is the expansion of the background data, such as census data from individual municipalities and other authorities could support for gaining insight of social, economical and ecological processes.

References

1. de Espindola, G.M.: Spatiotemporal trends of land use change in the Brazilian Amazon. Ph.D. thesis, National Institute for Space Research (INPE), São José dos Campos, Brazil (2012)
2. Kauppinen, T., de Espindola, G., Jones, J., Sanchez, A., Gräler, B., Bartoschek, T.: Linked Brazilian Amazon Rainforest Data. *Semantic Web Journal* 5(2) (2014)
3. Segel, E., Heer, J.: Narrative visualization: Telling stories with data. *Visualization and Computer Graphics, IEEE Transactions on* 16(6), 1139–1148 (2010)

A Demonstration of Linked Data Source Discovery and Integration^{*}

Jason Slepicka, Chengye Yin, Pedro Szekely, and Craig A. Knoblock

University of Southern California
Information Sciences Institute and Department of Computer Science, USA
{knoblock, pszekely, slepicka}@isi.edu
{chengyey}@usc.edu

Abstract. The Linked Data cloud is an enormous repository of data, but it is difficult for users to find relevant data and integrate it into their datasets. Users can navigate datasets in the Linked Data cloud with ontologies, but they lack detailed characterization of datasets' contents. We present an approach that leverages R2RML mappings to characterize datasets. Our demonstration shows how users can easily create R2RML mappings for their datasets and then use these mappings to find data from the Linked Data cloud and integrate it into their datasets.

1 Introduction

The Linked Data cloud contains an enormous amount of data about many topics. Consider museums, which often have detailed data about their artworks but may only have sparse data about the artists who created them. Museums typically have tombstone data about artists (name, birth/death years, and places) but may lack biographies, influences, etc. Museums could use additional information about their artists in the Linked Data cloud and integrate it with their own to produce a richer, more complete dataset.

Our approach to this, built into our KARMA data integration system [8], uses R2RML mappings [7] to describe users' datasets and datasets in the Linked Data cloud. Today, datasets include, at best, a VoID description [1] with basic metadata, such as access method and vocabularies used. R2RML-style mappings could complement VoID with their schema-like nature by capturing the semantic structure of a dataset and characterize its subjects and properties accordingly with statistics or set approximations like Bloom filters. With this information, users can reason better about how a dataset might integrate with their own data.

R2RML was defined to specify mappings from relational DBs to RDF, but recent work [2] has proposed extensions to handle data types like CSV, JSON, XML and Web APIs. Consequently, it is reasonable to expect that more datasets in the Linked Data cloud could be published with R2RML-style descriptions.

In this demonstration we show how museum users can use KARMA to quickly define an R2RML mapping of a dataset (our previous work), use R2RML mappings

^{*} A video demonstration is available at <http://youtu.be/sr-XDBKeNCY>

from other datasets to find more information about artists in their dataset, and then augment their dataset with that information.

2 Datasets

For our demonstration we will integrate a CSV file containing 197 artists with Linked Data published by the Smithsonian American Art Museum (SAAM). In previous work [8], we mapped the SAAM dataset, including over 40,000 artworks and 8,000 artists to the CIDOC CRM ontology [3] using R2RML and made it accessible by a SPARQL endpoint, along with a repository for the R2RML mappings. The SAAM LOD here is a proxy for the Linked Data cloud to illustrate the vision of a Linked Data cloud populated with R2RML models.

3 Demonstration

We will show how a user can interactively model an artist dataset, discover the Smithsonian’s data for those artists, and then integrate the Smithsonian’s data.

Step 1: Modeling a New Source. The user begins by using KARMA’s existing capability to model the artists in the CSV file as `crm:E21_Person` in an R2RML mapping shown in Figure 1. KARMA can use this mapping to generate RDF, and can also compare it to retrieve other mappings, discovering new related sources that can be integrated with the artist dataset.

Step 2: Discovering Data Sources. The user then clicks on `E21_Person1` in the R2RML mapping and selects `Augment Data` to discover new data to integrate into artist records. KARMA retrieves R2RML mappings in its repository that describe `crm:E21_Person`, and uses these mappings to generate a candidate set of linked data sources to integrate, identifies meaningful object and data properties, and presents them to the user as illustrated in Figure 2. To help the users select properties to integrate, Karma uses Bloom filters to estimate the number of artists that have each of the properties listed in Figure 2.

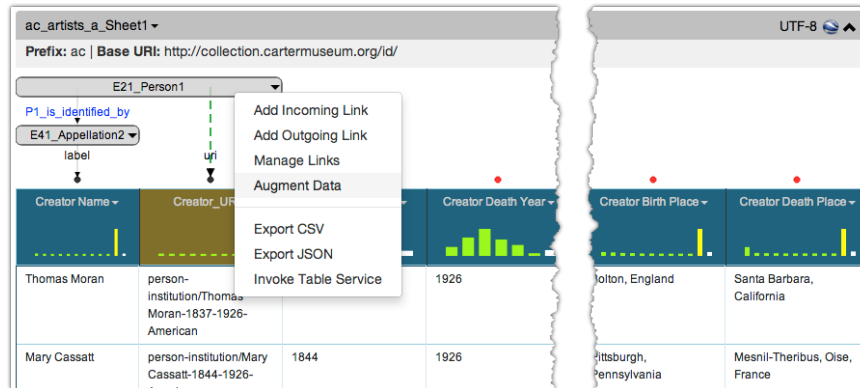


Fig. 1. A KARMA user creates an R2RML mapping for a CSV file of a museum’s artists’ biographical records and clicks ‘Augment Data’ to discover new data sources

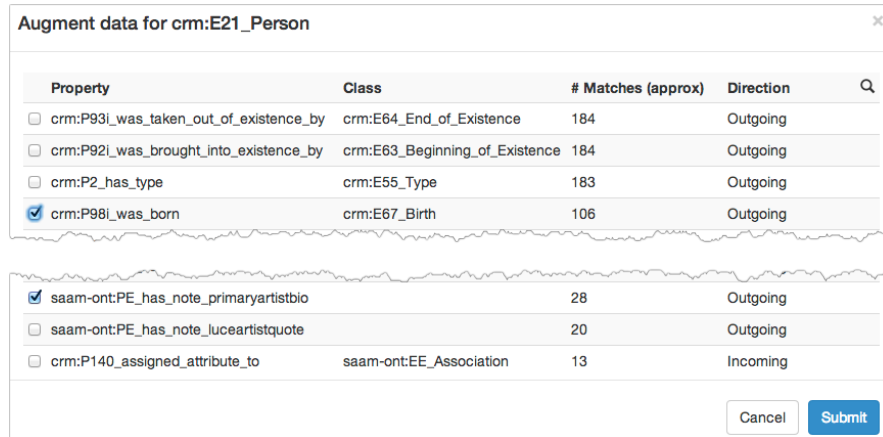


Fig. 2. A Karma user selects CIDOC CRM object and data properties discovered from other sources to augment crm:E21_Person

Step 3: Integrating Data Sources. The user selects the artist’s biography (for completeness) and birth (for validation). KARMA automatically constructs SPARQL queries to retrieve the data, integrates it into the worksheet, and augments the R2RML mapping accordingly (Figure 3). To support the integrated SPARQL queries, we generated owl:sameAs links between the artists in the CSV file and the Smithsonian dataset using LIMES [5] (we plan to integrate LIMES with KARMA to enable users to perform all integration steps within KARMA).

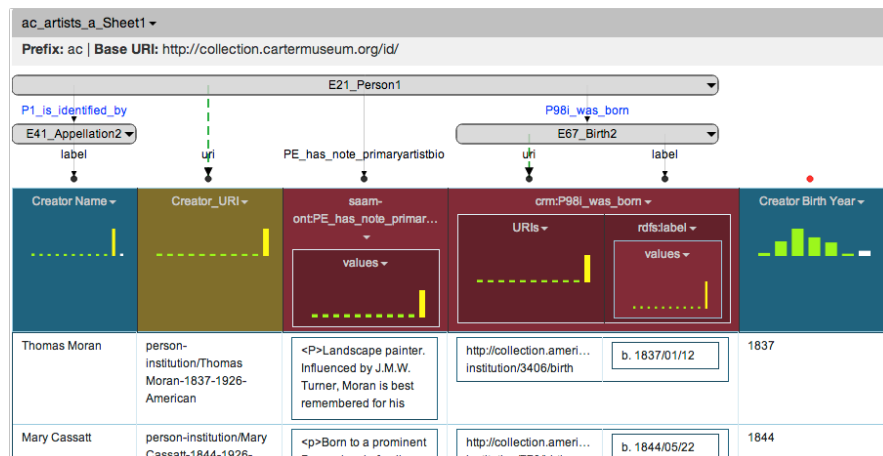


Fig. 3. A Karma user has integrated biographical data from the Smithsonian as new columns in their dataset. The columns contain artists’ biographies and birth dates.

4 Related Work and Conclusions

We see similarities in our approach with those used in relational database integration and semantic service composition. ORCHESTRA[4] starts, like R2RML, by aligning database tables to a schema graph. For integration, heuristics are used to translate keyword searches over the graph into join paths using its Q query system. However, these joins are not guaranteed to be semantically meaningful, unlike the integration paths KARMA finds using R2RML.

Platforms such as iServe[6] capture Linked Services and make them discoverable and queryable by annotating them with their Minimal Service Model. However, the past work on service discovery and composition only uses a semantic model of the inputs and outputs of the services. In contrast, KARMA service descriptions [9] also capture the relationship between the attributes, which allows us to automatically discover semantically meaningful joins.

By building on KARMA's ability to quickly model many source types, we demonstrate how a user can discover other linked data sources, select the desired attributes from those sources, and then integrate the data from those sources into their own dataset. Through this source discovery and integration, a user can transparently compose and join other sources and services in a semantically meaningful, interactive way that was not previously possible.

References

1. ALEXANDER, K., CYGANIAK, R., HAUSENBLAS, M., AND ZHAO, J. Describing linked datasets with the VoID vocabulary. W3C note, W3C, Mar. 2011.
2. DIMOU, A., SANDE, M. V., COLPAERT, P., MANNENS, E., AND DE WALLE, R. V. Extending R2RML to a source-independent mapping language for RDF. In *International Semantic Web Conference (Posters and Demos)* (2013), vol. 1035 of *CEUR Workshop Proceedings*, CEUR-WS.org, pp. 237–240.
3. DOERR, M. The CIDOC conceptual reference module: An ontological approach to semantic interoperability of metadata. *AI Mag.* 24, 3 (Sept. 2003), 75–92.
4. IVES, Z. G., GREEN, T. J., KARVOUNARAKIS, G., TAYLOR, N. E., TANNEN, V., TALUKDAR, P. P., JACOB, M., AND PEREIRA, F. The ORCHESTRA collaborative data sharing system. *ACM SIGMOD Record* 37, 3 (2008), 26–32.
5. NGOMO, A.-C. N., AND AUER, S. LIMES: a time-efficient approach for large-scale link discovery on the web of data. In *Proceedings of the Twenty-Second international joint conference on Artificial Intelligence* (2011), AAAI Press, pp. 2312–2317.
6. PEDRINACI, C., LIU, D., MALESHKOVA, M., LAMBERT, D., KOPECKY, J., AND DOMINGUE, J. iServe: a linked services publishing platform. In *CEUR workshop proceedings* (2010), vol. 596.
7. SUNDARA, S., CYGANIAK, R., AND DAS, S. R2RML: RDB to RDF mapping language. W3C recommendation, W3C, Sept. 2012.
8. SZEKELY, P., KNOBLOCK, C. A., YANG, F., ZHU, X., FINK, E., ALLEN, R., AND GOODLANDER, G. Connecting the Smithsonian American Art Museum to the Linked Data Cloud. In *Proceedings of the 10th ESWC* (2013).
9. TAHERIYAN, M., KNOBLOCK, C. A., SZEKELY, P., AND AMBITE, J. L. Semi-automatically modeling web APIs to create linked APIs. In *Proceedings of the ESWC 2012 Workshop on Linked APIs* (2012).

Developing Mobile Linked Data Applications

Oshani Seneviratne¹, Evan W. Patton^{2,1}, Daniela Miao¹, Fuming Shih¹, Weihua Li¹, Lalana Kagal¹, and Carlos Castillo³

¹ Massachusetts Institute of Technology

{oshanis, ewpatton, dmiao, fuming, wli17, lkagal}@csail.mit.edu

² Rensselaer Polytechnic Institute pattoe@rpi.edu

³ Qatar Computing Research Institute chatoc@acm.org

Abstract. With the rapid advancement of mobile technologies, users are generating and consuming a significant amount of data on their handheld devices. However, the lack of Linked Data tools for these devices has left much of the data unstructured and difficult to reuse and integrate with other datasets. We will demonstrate an application development framework that enables the easy development of mobile apps that generate and consume Linked Data. We also provide a set of easy-to-deploy Web Services to supplement functionality for mobile apps focused on crowdsourcing. We motivate our work by describing a real-world application of this framework, which is a disaster relief application that streams crowd-sourced reports in real time.

1 Introduction

Many developers are shifting their attention to the mobile world as smartphones are becoming the information hub for people’s daily lives [1]. The pervasiveness of smartphones has led to the ubiquitous consumption and generation of data on them. Smartphones can derive contextual information from their environment, enabling applications that provide great value both to individual users and to society. For example, context-aware applications can recommend products, services, or connections to others based on people’s surroundings. People can also use social applications to report their status and the situation around them in cases of emergencies, such as floods or earthquakes.

Most mobile applications create or consume data stored in standalone databases without the potential of being “interlinked” with data from other applications or organizations. The Web community has advocated the use of Linked Data technologies to address this data interoperability issue in Web-based applications. Although we have seen some success from content publishers in using or publishing Linked Data, few examples exist for mobile platforms [2].

Our goal is to bring support for Linked Data to mobile platforms by allowing developers to build mobile applications that consume and publish Linked Data automatically. We will demonstrate a framework that allows developers to accomplish this on Android devices through modular components, as well as cloud scripts aimed at enabling quick deployment of companion web services to interact with streams of Linked Data. We believe that this framework will reduce the burdens of mobile developers to work with Linked Data and open up many opportunities for building applications that interact with and contribute to the Linked Data world.



Fig. 1: **Consuming Linked Data.** *Left:* Logic used for obtaining the user’s location, constructing a SPARQL query, querying a remote endpoint, and displaying the retrieved results on a map. *Right:* Output as seen on the phone. ‘QueryButton’ in the blocks editor is the name of the button with the label ‘Get Landmarks near Current Location’ as seen on the phone.

2 Framework

Our framework¹ extends MIT App Inventor,² an open source, web-based environment for developing Android applications. Though primarily designed for pedagogical use, App Inventor has found great success in many domains and boasts about 87,000 users weekly, 1.9 million total users, and over 4.9 million applications ranging from tracking rainfall in Haiti³ to teaching U.S. Marines how to destroy weapons.⁴ App Inventor is structured around *components*: discrete blocks that provide functionality and user interface elements that developers connect together to create an application. App Inventor provides a designer view for laying out an application’s user interface and a blocks view where users define their program logic by connecting method blocks, associated with components, together (as shown in Fig. 1).

We will demonstrate several components that we’ve developed powered by Apache Jena⁵, including forms to generate Linked Data, maps for visualizing geographic data, web services for cloud messaging, web services for exploring Linked Data, and sensor components for context awareness. We will also demonstrate companion web services that provide cloud functionality for mobile phones such as providing messaging services to integrate streaming RDF content.

3 Use Case: WeReport

WeReport allows people to report (e.g. through a photo) the situation on the ground during an emergency, hence enhancing situational awareness with respect to the emergency [3]. The application demonstrates the capacity to generate and consume Linked Data, as well as integrate with different public datasets.

Consider a scenario where a hurricane has hit a city, and Joe, a volunteer citizen reporter, notices a series of potential hazards in his neighborhood, e.g. fallen trees blocking an intersection. With WeReport, Joe can take a picture of the hazard and upload it, along with a tag and description, to warn others in the area. An example of this report can be seen in Figure 2a.

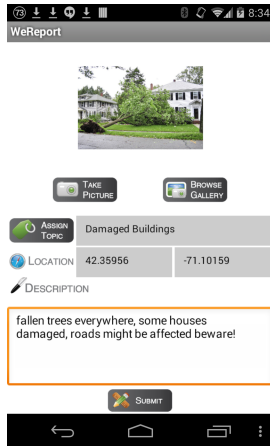
¹ <http://punya.appinventor.mit.edu>

² <http://appinventor.mit.edu/>

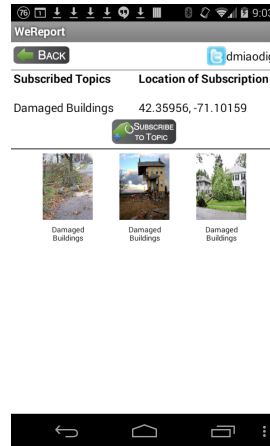
³ <http://www.webcitation.org/6PUXWVG0U>

⁴ <http://www.webcitation.org/6PUXZb7FM>

⁵ <https://jena.apache.org>



(a) Submitting Reports



(b) Browsing Reports

Fig. 2: WeReport application for Mobile Devices: an application that allows users to submit disaster reports to the cloud. Users can subscribe to reports close to a certain location on a topic such as ‘Fallen Trees’, and results are streamed in real-time to subscribers via push notifications.

When creating a report, WeReport consumes Linked Data via pre-defined SPARQL queries that search for popular landmarks near Joe’s location, so that those places can be tagged if hazardous. At the same time, the mobile application enables him to transparently produce structured data through our Linked Data-enhanced components. The generated data can be published to SPARQL 1.1 compliant endpoints, which enables data re-use by others.

WeReport also supports real-time streaming of user-submitted disaster reports. In this scenario, Bob and Anna are both relief workers heading towards the affected area. Using WeReport, they have subscribed to the hurricane disaster feed and they continuously receive push notifications as reports on the disaster area are submitted by users like Joe. While Bob wishes to be continuously notified on every newly submitted report, Anna chooses to receive an alert only when there are at least 3 reports of a certain type in a given location, a custom limit she had previously specified. Whenever Bob and Anna want to view the newest reports, WeReport provides a “Browse Reports” interface which automatically displays the received reports, as show in Figure 2b.

4 Related Work

There has been some work in Linked Data use for smartphones, however, they do not provide a comprehensive solution to mobile app development as our framework does. [4] propose a lightweight general-purpose RDF framework for Android to share application data inside the phone. Our framework focuses on obtaining external RDF sources and consuming them and not necessarily application data integration using RDF. *Spinel* [5] is a plug-in architecture and a set of web configuration tools for Android that enable new datasets to be added to mobile phones without programming. We provide both a plug-in architecture and support for adding new datasets using program blocks that can be configured easily. Another notable platform for creating mobile Linked Data applications is the “RDF on the go”, an RDF storage and query processor library for mobile devices [6]. Our framework not only allows storage and querying of RDF data, but more importantly makes it much more user

friendly to manipulate the data, be it contextual data from the phone or data from a remote endpoint.

Unlike applications such as Cinemappy [7] and DBPedia Mobile [8], which require developers to have extensive knowledge of both Linked Data and mobile programming, our target audience is Linked Data developers who want to develop mobile applications but do not have mobile programming experience.

5 Summary

Though mobile devices have become the primary computing and communication platform for users, the number of Linked Data apps available for these devices is insignificant because developing them is currently extremely time consuming.

In this demonstration, we show a mobile application platform that enables mobile Linked Data applications to be quickly and easily developed. Along with leading to applications that are able to leverage structured data, these mobile applications will help get around the classic “chicken-or-egg” situation by being both generators and consumers of Linked Data. The current implementation contains a few limitations including scalability, which our team will be working on. We will also be conducting further user studies to evaluate the usability and usefulness of the platform.

References

1. Tillmann, N., Moskal, M., de Halleux, J., Fahndrich, M., Bishop, J., Samuel, A., Xie, T.: The future of teaching programming is on mobile devices. In: Proceedings of the 17th ACM annual conference on Innovation and technology in computer science education, ACM (2012) 156–161
2. Ermilov, T., Khalili, A., Auer, S.: Ubiquitous Semantic Applications: A Systematic Literature Review. *International Journal on Semantic Web and Information Systems* **10**(1) (2014)
3. Vieweg, S.: Situational Awareness in Mass Emergency: A Behavioral and Linguistic Analysis of Microblogged Communications. PhD thesis, University of Colorado at Boulder (2012)
4. David, J., Euzenat, J., Rosoiu, M., et al.: Linked data from your pocket. In: Proc. 1st ESWC workshop on downscaling the semantic web. (2012) 6–13
5. Chang, K.S.P., Myers, B.A., Cahill, G.M., Simanta, S., Morris, E., Lewis, G.: A plug-in architecture for connecting to new data sources on mobile devices. In: Visual Languages and Human-Centric Computing (VL/HCC), 2013 IEEE Symposium on, IEEE (2013) 51–58
6. Le Phuoc, D., Parreira, J.X., Reynolds, V., Hauswirth, M.: RDF On the Go: RDF Storage and Query Processor for Mobile Devices. In: ISWC Posters&Demos. (2010) 12
7. Ostuni, V., Gentile, G., Noia, T., Mirizzi, R., Romito, D., Sciascio, E.: Mobile movie recommendations with linked data. In Cuzzocrea, A., Kittl, C., Simos, D., Weippl, E., Xu, L., eds.: Availability, Reliability, and Security in Information Systems and HCI. Volume 8127 of Lecture Notes in Computer Science. Springer Berlin Heidelberg (2013) 400–415
8. Passant, A.: Measuring Semantic Distance on Linking Data and Using it for Resources Recommendations. In: AAAI Spring Symposium: Linked Data Meets Artificial Intelligence. (2010)

A Visual Summary for Linked Open Data sources

Fabio Benedetti, Sonia Bergamaschi, Laura Po

Università di Modena e Reggio Emilia - Dipartimento di Ingegneria "Enzo Ferrari" - Italy
firstname.lastname@unimore.it

Abstract. In this paper we propose LODeX, a tool that produces a representative summary of a Linked open Data (LOD) source starting from scratch, thus supporting users in exploring and understanding the contents of a dataset. The tool takes in input the URL of a SPARQL endpoint and launches a set of pre-defined SPARQL queries, from the results of the queries it generates a visual summary of the source. The summary reports statistical and structural information of the LOD dataset and it can be browsed to focus on particular classes or to explore their properties and their use. LODeX was tested on the 137 public SPARQL endpoints contained in Data Hub (formerly CKAN)¹, one of the main Open Data catalogues. The statistical and structural information extraction was successfully performed on 107 sources, among these the most significant ones are included in the online version of the tool².

1 Introduction

The RDF Data Model plays a key role in the birth and continuous expansion of the Web of data since it allows to represent structured and semi-structured data. However, while the LOD cloud is still growing, we assist to a lack of tools able to produce a meaningful, high level representation of these datasets.

Quite a lot of portals catalog datasets that are available as LOD on the Web and permit users to perform keyword search over their list of sources. Nevertheless, when a user starts exploring in details an unknown LOD dataset, several issues arise: (1) the difficulty in finding documentation and, in particular, a high level description of classes and properties of the dataset; (2) the complexity of understanding the schema of the source, since there are no fixed modeling rules; (3) the effort to explore a source with a high number of instances; (4) the impossibility, for non skilled users, to write specific SPARQL queries in order to explore the content of the dataset.

To overcome the above problems, we devise LODeX, a tool able to automatically provide a high level summarization of a LOD dataset, including its inferred schema. It is composed by several algorithms that discern between intensional and extensional knowledge. Moreover, it handles the problem of long running queries, that are subject to timeout failures, by generating a pool of low complexity queries able to return the same information.

This work has been accomplished in the framework of a PhD program organized by the Global Grant Spinner 2013, and funded by the European Social Fund and the Emilia Romagna Region.

¹ <http://datahub.io>

² <http://dbgroup.unimo.it/lodex>

As presented in [3], the majority of the tools for data visualization is not able to provide a synthetic view of the data (instances) contained in a single source. Payola³ [4] and LOD Visualization⁴ [2] are two recent tools that exploits analysis functionalities for guiding the process of visualization. However, these tools always need some querying parameters to start the analysis of a LOD dataset. Conversely, LODeX neither requires any a priori knowledge of the dataset, nor asks users to set any parameters; it focuses on extracting the schema from a LOD endpoint and producing a summarized view of the concepts contained in the dataset.

The paper is structured as follows. Section 2 describes the architecture of LODeX, while a use case and demonstration scenario is described in Section 3. Conclusions and some ideas for future work are described in Section 4.

2 LODeX - Overview

LODeX aims to be totally automatic in the production of the schema summary.

Figure 1 depicts the architecture of LODeX. The tool is composed by three main processes: *Index Extraction*, *Post-processing* and *Visualization*. The goal of the first two steps is to automatically extract from a SPARQL endpoint the information needed to produce its schema summary, while the third step aims to produce a navigable view of schema summary for the users. For an easy reuse, all the contents extracted and processed by LODeX are stored in a NoSQL document database, since it allows a flexible representation of the indexes.

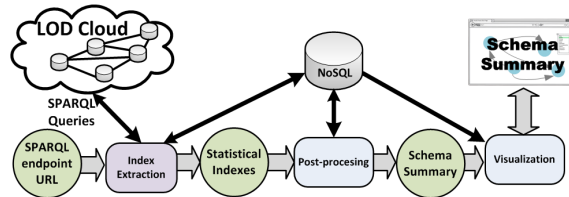


Fig. 1. LODeX Architecture

The **Index Extraction (IE)** takes as input the URL of a SPARQL endpoint and generates the queries needed to extract structural and statistical information about the source. Major details about the IE process can be found in [1]. The IE component has been designed in order to maximize the compatibility with LOD sources and minimize the costs in terms of time and computational complexity. The intensional and extensional knowledge are extracted and collected in a set of statistical indexes, stored in the NoSQL Database.

The **Post-processing (PP)** combines the information contained in the statistical indexes to produce the schema summary of a specific dataset. The summary is induced

³ <http://live.payola.cz/>

⁴ <http://lodvisualization.appspot.com/>

from the distribution of the instances in the dataset. The PP also collects synthetic information regarding the endpoint. Also the schema summary is stored in the NoSQL database.

The **Visualization** of the schema summary is performed through a web application written in Python that uses NoSQL database as backend. We used Data Driven Documents⁵ to create a visual representation of the dataset with which the user can interact to navigate the schema and discover the information that he/she is looking for.

The tool has been tested on the entire set of sources described in *SPARQL Endpoint Status*(SPARQLES)⁶, a specialized application that recursively monitors the availability of public SPARQL Endpoints contained in DataHub. At the time of our evaluation (May 2014), SPARQLES indicated that the 52% of SPARQL endpoints (244/469) were available and only the 13% of the endpoints presented a documentation, i.e. VOID and/or Service descriptions. LODeX was able to complete the extraction phase, thus building the visual summaries, for 107 LOD sources (78% of the 137 dataset that were compliant with the necessary SPARQL operators) that are now collected and shown in the online demo.

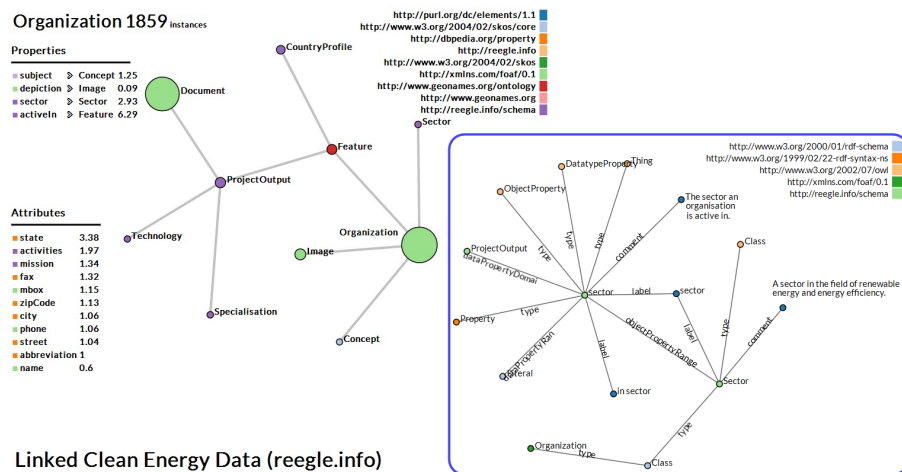


Fig.2. Visual Summary of the Linked Clean Energy Data source and a particular of the “Sector” property.

3 Use Case and Demonstration Scenario

We refer to an hypothetical use-case involving a company in the clean energy sector. The company has its own products and services and attempts to discover new information on renewable energy and energy efficiency in the country where it is located. While

⁵ <http://d3js.org/>

⁶ <http://sparqls.okfn.org/>

searching the key datasets in the energy field, the company will likely find the Linked Clean Energy Data dataset⁷. This dataset, composed of 60140 triples, is described as a “Comprehensive set of linked clean energy data including: policy and regulatory country profiles, key stakeholders, project outcome documents and a thesaurus on renewable, energy efficiency and climate change for public re-use”.

By using our application to explore this dataset (see Figure 2)⁸, the user can, at a glance, have the intuition of all the instantiated classes (the nodes in the graph) and the connections among them (the arcs), besides the number of instances defined for each class (reflected in the dimension of the node). Focusing on the color of the nodes in the graph, a user can understand which classes are defined by the provider of the source and which others are taken from external vocabularies (in this case we can see that some of the class definitions are acquired from Foaf, Geonames.org and Skos). By positioning the mouse on a node, more information about the class are shown (as depicted in Figure 2 on the left). Since classes are linked to each others by some properties, it is possible to explore the property details. Thus, by clicking on a property another visual representation of the intensional knowledge is shown (see the right part of Figure 2).

4 Conclusions and Future Work

This paper has shown how LODeX is able to provide a visual and navigable summary of a LOD dataset including its inferred schema starting from the URL of a SPARQL Endpoint. The result gained by LODeX could also be useful to enrich LOD sources’ documentation, since the schema summary can be easily translated with respect to a vocabulary and inserted into the LOD source. LODeX is currently limited to display the contents of a source proposing a graph. However, new developments are being implemented in order to facilitate the query definition by exploiting the visual summary.

References

1. F. Benedetti, S. Bergamaschi, and L. Po. Online index extraction from linked open data sources. To appear in Linked Data for Information Extraction (LD4IE) Workshop held at International Semantic Web Conference, 2014.
2. J. M. Brunetti, S. Auer, and R. Garca. The linked data visualization model. In *International Semantic Web Conference (Posters & Demos)*, 2012.
3. A.-S. Dadzie and M. Rowe. Approaches to visualising linked data: A survey. *Semantic Web*, 2(2):89–124, 2011.
4. J. Klímek, J. Helmich, and M. Nečaský. Payola: Collaborative linked data analysis and visualization framework. In *The Semantic Web: ESWC 2013 Satellite Events*, pages 147–151. Springer, 2013.

⁷ <http://data.reegle.info/>

⁸ The visual summary of this source is available at <http://dbgroup.unimo.it/lodex/157>

EasyESA: A Low-effort Infrastructure for Explicit Semantic Analysis

Danilo Carvalho^{1,2}, Çağatay Çallı³, André Freitas¹, Edward Curry¹

¹Insight Centre for Data Analytics, National University of Ireland, Galway

²PESC/COPPE, Federal University of Rio de Janeiro (UFRJ)

³Department of Computer Engineering, METU, Ankara

1 Introduction

Distributional semantic models (DSMs) are semantic models which are based on the statistical analysis of co-occurrences of words in large corpora. DSMs can be used in a wide spectrum of semantic applications including semantic search, question answering, paraphrase detection, word sense disambiguation, among others. The ability to automatically harvest meaning from unstructured heterogeneous data, its simplicity of use and the ability to build comprehensive semantic models are major strengths of distributional approaches.

The construction of distributional models, however, is dependent on processing large-scale corpora. The English version of Wikipedia 2014, for example, contains 44 GB of article data. The hardware and software infrastructure requirements necessary to process large-scale corpora bring high entry barriers for researchers and developers to start experimenting with distributional semantics. In order to reduce these barriers we developed EasyESA, a high-performance and easy-to-deploy distributional semantics framework and service which provides an Explicit Semantic Analysis (ESA) [4] infrastructure.

2 Explicit Semantic Analysis (ESA)

DSMs are represented as a *vector space model*, where each dimension represents a *context* \mathcal{C} for the linguistic context in which the *target word* occurs in a reference corpus. A *context* can be defined using documents, co-occurrence window sizes (number of neighbouring words or data elements) or syntactic features. The *distributional interpretation* of a target word is defined by a weighted vector of the contexts in which the word occurs, defining a geometric interpretation under a distributional vector space. The weights associated with the vectors are defined using an *associated weighting scheme* \mathcal{W} , which can recalibrate the relevance of more generic or discriminative contexts. A *semantic relatedness measure* \mathcal{S} between two words can be calculated by using different *similarity/distance* measures such as the *cosine similarity* or *Euclidean distance*.

In the Explicit Semantic Analysis DSM [4], Wikipedia is used as a reference corpus and the contexts are defined by each Wikipedia article. The weighting scheme is defined by TF/IDF (term frequency/inverse document frequency) and

the similarity measure by the *cosine similarity*. The interpretation vector of a term on ESA is a weighted vector of Wikipedia articles, which is called in the ESA model a concept vector.

A keyword query over the ESA semantic space returns the list of ranked articles titles, which define a concept vector associated with the query terms (where each vector component receives a relevance score). The approach supports the interpretation of small text fragments, where the final context vector is the centroid of the words' concept vectors. The ESA semantic relatedness measure between two terms is calculated by computing the cosine similarity between the concept vectors representing the interpretation of the two terms.

3 EasyESA

EasyESA consists of an open source platform that can be used as a remote service or can be deployed locally. The API consists of three services:

Semantic relatedness measure: Calculates the *semantic relatedness measure* between two terms. The semantic relatedness measure is a real number in the [0,1] interval, representing the degree of semantic proximity between two terms. Semantic relatedness measures are comparative measures and are useful when sets of terms are compared in relation to their semantic proximity. Semantic relatedness can be used for semantic matching in the context of the development of semantic systems such as question answering, text entailment, event matching and semantic search.

- *Example:* Request for the semantic relatedness measure between the words *wife* and *spouse*.
- *Service URL:* <http://vmdeb20.deri.ie:8890/esaservice?task=esa&term1=wife&term2=spouse>

Concept vector: Given a term, it returns the associated concept vector: a weighted vector of contexts (Wikipedia articles). The term can contain multiple words. The concept vectors can be used to build semantic indexes, which can be applied for semantic applications which depends on high performance semantic matching. An example of a semantic index built using ESA concept vectors is available in [1].

- *Example:* Request for the concept vector of the word *wife* with maximum dimensionality of 50.
- *Service URL:* <http://vmdeb20.deri.ie:8890/esaservice?task=vector&source=wife&limit=50>

Query explanation: Given two terms, returns the overlap between the concept vectors.

- *Example:* Request for the concept vector overlap between the words *wife* and *spouse* for concept vectors with 100 dimensions.
- *Service URL:* <http://vmdeb20.deri.ie:8890/esaservice?task=explain&term1=wife&term2=spouse&limit=100>

Mean request values are 0.055 ms for the semantic relatedness measure and 0.080 ms for the concept vector (500 dimensions) on an Intel Core i7 Quad Core 3770 3.40 GHz 32GB DDR3 RAM computer.

EasyESA was developed using Wikiprep-ESA¹ as a basis. The software is available as an open source tool at <http://easy-esa.org>. The improvements targeted the following contributions: (i) major performance improvements (fundamental for the application of distributional semantics in real applications which depends on 100s of requests per second); (ii) robust concurrent queries; (iii) RESTful service API; (iv) deployment of an online service infrastructure; (v) packaging and pre-processed files for easy deployment of a local ESA infrastructure. A detailed description of the improvements can be found at <http://easy-esa.org/improvements>.

4 Demonstrations

Two demonstration applications were built using EasyESA targeting to show the low effort involved in the use of distributional semantics in the context of different semantic tasks. In the demonstration, Wikipedia 2013 was used as a reference corpus. Videos of the running applications are available at: <http://treo.deri.ie/iswc2014demo>

Semantic Search: The first demonstration consists is the use of EasyESA for simulating a semantic search application. In this scenario users can enter a set of terms which can represent the searchable items (for example film genres). Each term associated with a genre has a distributional conceptual representation, i.e. is represented by a concept vector. Users can then enter a search term which has no lexical similarity to the indexed terms. The demonstration computes the semantic relatedness for each vector, ranking the results by their degree of semantic relatedness. In the example, the search query *'winston churchill'* returns the most likely film genres which are associated with the query. Genres *'war'*, *'documentary'*, and *'historical'* were the top most related terms. Figure 1 shows a screenshot of the interface of the example. The demonstration application can be accessed in: <http://vmdeb20.deri.ie/esa-demo/sensearch.html>.

Word Sense Disambiguation (WSD): In the second demonstration, EasyESA is used to perform a word sense disambiguation task (WSD). The user enters a sentence and then selects a word to get the correct sense from WordNet. The WSD application gets the *sentence context* (the words surrounding the target word) finds its associated context vector and computes the semantic relatedness measure in relation to the context vector of the *associated WordNet glosses* for each word sense available. The different senses are then ranked by their semantic relatedness values. In the examples there is no lexical overlap between the sentence context and the different WordNet glosses, with the distributional knowledge from Wikipedia filling the semantic gap between the context and the glosses. The demonstration application can be accessed in: <http://vmdeb20.deri.ie/esa-demo/sensedisambig.html>.

¹ <https://github.com/faraday/wikiprep-esa>

ESA Semantic Search Demonstrator

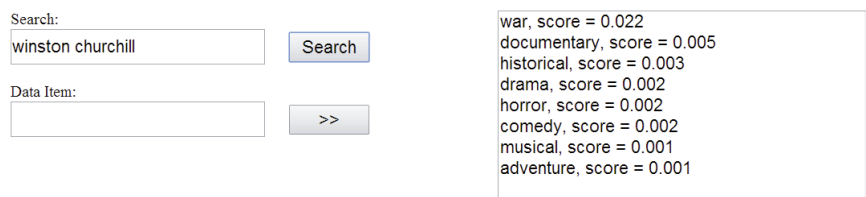


Fig. 1: Screenshot of the semantic search application.

5 Applications using EasyESA

While the demonstration focuses on providing easy to replicate applications for users to start experimenting with distributional semantics, more complex applications were built using EasyESA. Freitas et al [2] used EasyESA for terminology-level search over Linked Data vocabularies, achieving better performance in the semantic matching process when compared to WordNet-based query expansion approach. EasyESA was used in the Treo QA system [1], a schema-agnostic query approach over the Linked Data Web. The system uses a distributional semantics approach to match query terms to dataset elements, supporting schema-agnostic queries. Hasan & Curry [3] use EasyESA for semantically matching complex events from semantically heterogeneous data sources, in a real time scenario.

Acknowledgment: This publication was supported in part by Science Foundation Ireland (SFI) (Grant Number SFI/12/RC/2289) and by the Irish Research Council.

References

1. Freitas, A., Curry, E., Natural Language Queries over Heterogeneous Linked Data Graphs: A Distributional-Compositional Semantics Approach. *In Proc. of the 19th Intl. Conf. on Intelligent User Interfaces.* (2014).
2. Freitas, A., Curry, E., O’Riain, S., A Distributional Approach for Terminological Semantic Search on the Linked Data Web. *In Proc. of the 27th ACM Symposium On Applied Computing (SAC), Semantic Web and Applications (SWA).* (2012).
3. Hasan, S., Curry, E., Approximate Semantic Matching of Events for The Internet of Things,. *In ACM Transactions on Internet Technology (TOIT).* (2014).
4. Gabrilovich, E., Markovitch S., Computing semantic relatedness using Wikipedia-based explicit semantic analysis. *In Proc. of the 20th Intl. Joint Conf. on Artificial Intelligence,* 1606–1611. (2007).

LODHub - A Platform for Sharing and Analyzing large-scale Linked Open Data

Stefan Hagedorn and Kai-Uwe Sattler

Database & Information Systems Group,
Technische Universität Ilmenau, Germany
`{first.last}@tu-ilmenau.de`

Abstract. In this demo paper we describe the current prototype of our new platform LODHub that allows users to publish and share linked datasets. The platform further allows to run SPARQL queries and execute Pig scripts on these datasets to support users in their data processing and analysis tasks.

Keywords: Linked Open Data, data processing, data analysis, web platform

1 Introduction

Over the last years, the (Linked) Open Data movement has received a growing popularity. More and more public and government agencies as well as organizations publish data of common interest allowing to make use of it and building interesting applications. This is fostered by various hubs such as datahub.io, data.gov etc., which represent central registries for data sources.

However, creating added value from the published data requires typically to preprocess and clean the data, combine it with other data, and to create new datasets or build models. Results of such transformation and analysis tasks are twofold: first, the new datasets or models might be useful for other users in the form of curated datasets and second, the tasks could be reused for other datasets, too. Particularly, recent developments on big data technologies provide numerous useful building blocks for such an environment, e.g. scalable platforms like Hadoop or Spark or higher-level programming models and data flow languages like Pig or Jaql.

In [2] we argue that there is a need for a platform addressing these requirements by combining functionalities of Open Data management frameworks with an infrastructure for exploration, processing, and analytics over large collections of (linked) data sets while respecting access and privacy restrictions of the datasets.

In this demo we plan to show our initial results of building such a platform. Our LODHub platform provides services for uploading and sharing (RDF) datasets. Furthermore, it contains facilities to explore, query, and analyze datasets by integrating a SPARQL query processor as well as a visual data flow tool for designing and executing Pig scripts on the hosted datasets.

2 LODHub Services

The platform has several features that let users work with their datasets. However, it is currently only a prototype and some features are not completed yet.

2.1 Managing datasets

Upload Users can upload their datasets via the website. During the upload process, the user has the possibility to enter the dataset name, tags, and a short description. The tags can later be used to search in the users collection of datasets. An import feature that lets users upload datasets that are not in the linked data format (e.g., CSV files, Excel sheets, etc.) is not finished yet, but it is a work in progress.

Update During time, the contents of a dataset may change. These changes can be updated values for some particular statements, new additional statements, or removed statements. If the set of changes is small, one might consider it a new version of a dataset instead of a completely new one. To reflect this issue in LODHub, the platform supports versioning. This means it is possible to explicitly upload a new version of a dataset. By default, users will use the most recent version of a dataset. However, it is also possible to switch to an older version and, e.g., run queries on that version.

Collaboration The idea of LODHub is to allow users to work together on potentially large datasets. When a user uploads a new dataset, she or he becomes the owner of it and hence, has the permission to perform all operations on it. To also allow other users to work with this dataset, one can share the dataset to other users. With sharing we mean that the other users can see this dataset in their collection so that they are able to work with it. When sharing a dataset, the owner can choose what permission the other users should have on this dataset: *Read* allows to query the dataset, *Write* is like Read access plus the permission to upload new versions, *Share* is the permission to share the dataset to other users.

2.2 Querying datasets

Working with datasets means to run queries on them to find the needed information. However, there are different types of queries that users need to execute. On the one hand, there are the rather short ad-hoc queries that can be run directly on the datasets. On the other hand there are the data-intensive transformation and analysis tasks. LODHub supports both types of workload by providing two ways to formulate the queries.

Ad-hoc queries Since LODHub was designed around linked data, the main query language used on the platform is SPARQL. SPARQL queries can be used to instantly formulate a query on one dataset or even the union of many datasets. The user is presented a text input area where she or he can enter the query. The query is then directly executed by the underlying RDF framework (in our case Jena) and the result triples are presented on the website.

Analytical tasks Writing SPARQL queries can be cumbersome and requires the users to learn the language. Furthermore, there may be complex tasks that are too difficult to express in SPARQL, e.g., data transformation steps, or complex operators that are not even available in SPARQL. In this case, it is easier to formulate the query in a script language where data is processed in a streaming fashion to achieve high throughput.

To achieve a low entrance barrier, we provide a graphical editor that lets users create queries via drag and drop. Users can choose between several predefined operators which they can then connect to model the data flow between the operators. For each operator, its parameters like filter conditions, join attributes, or projection columns can be set individually. Thus, users intuitively build an operator tree, without having to care about language specific rules.

Each operator is translated into a Pig script statement. Pig, being a framework on top of Apache Hadoop, allows a distributed execution of the query and to load the data from a distributed file system (e.g., HDFS). Hence, this approach does not use the generated indexes from the used RDF framework. However, the data flow approach allows a high parallelization in a cluster environment.

Currently, the graphical editor produces Pig scripts only. However, the editor was designed so that it will be possible to also generate other languages from the graph. Thus, in a future version the editor will be able to generate traditional SPARQL queries and maybe other languages, too.

Data exploration To help people to understand the content of datasets and to find useful datasets that may contribute to the users question, LODHub allows to visualize how datasets are interlinked with each other, i.e., how many objects of a dataset occur as subjects in another dataset (and vice versa).

3 Architecture

The platform was written using the Play! framework¹. Play's integration of Akka allows to easily run the application in a cluster environment. However, in the current development phase we concentrate on a single machine, but plan to distribute the application in a cluster to achieve better load distribution.

To store the datasets, we use the Jena TDB framework². The SPARQL queries on these datasets are directly passed to the Jena library which then

¹ <http://www.playframework.com/>

² <http://jena.apache.org/documentation/tdb/>

evaluates the query. The Pig scripts are currently executed in `local` mode, i.e., they are not distributed to a cluster. However, this is just a configuration step, so that the application can be run in a cluster environment without having to change a lot of code.

The modular design of the application will enable us to easily replace the RDF store with another one or even to install a new store along with the others. Thus, we could use for example our fast CameLOD [1] for analytical SPARQL queries that have to read a massive amount of data, while transactional queries are still handled by the Jena TDB store.

4 Demo

In our demo we will show the features that were described in the previous section. Users will be able to upload datasets, update existing ones, and to execute queries on the datasets. For the queries they can either type in traditional SPARQL queries or use our graphical editor to define a data flow and then generate a Pig script from the created graph.

A short demonstration of the current status of the platform can be found in this video:

<http://youtu.be/m4kKiBrw2m4>

In this demonstration we used several datasets which contain information about events, their location, date, and an URL for more information, as well as datasets containing information about cities and the country they belong to.

After an introduction of the dashboard that is the user's entry point to all actions, we run a SPARQL query with a `GROUP BY` and `HAVING` clause to find the subjects that have the same predicate two or more times.

Next, we show how to create more complex analytical data processing tasks using the graphical editor to create the Pig scripts. In this editor we create a data flow using a `LOAD`, `GROUP BY`, and a `PROJECTION` operator. For each operator, we enter the necessary parameters. `LOAD`: the path of the file to load and the schema, `GROUP BY`: the grouping column, and `PROJECTION`: the column names to project. After uploading a new dataset we create a second Pig script that uses a special `MATERIALIZER` operator. This operator allows to materialize the results of a Pig script into a new dataset that is immediately available to the user and can be used just like a normal dataset.

At the end of the demo video we show how to visualize the interlinks between selected datasets.

References

1. Hagedorn, S., Sattler, K.U.: Efficient parallel processing of analytical queries on linked data. In: OTM, pp. 452–469 (Sept 2013)
2. Hagedorn, S., Sattler, K.U.: Lodhub - a platform for sharing and integrated processing of linked open data. In: In proceeding of: 5th International Workshop on Data Engineering Meets the Semantic Web. pp. 260–262. IEEE (March 2014)

LOD4AR: Exploring Linked Open Data with a Mobile Augmented Reality Web Application

Silviu Vert, Bogdan Dragulescu and Radu VasIU

Multimedia Research Centre, Politehnica University Timisoara, Timisoara, Romania
{silviu.vert, bogdan.dragulescu, radu.vasiu}@cm.upt.ro

Abstract. There is a vast amount of linked open data published nowadays on the web, ranging from user-generated data to government public data. This data needs visualization and exploration tools which people can use to make sense of the data and turn it into useful and used information. Several such tools have been proposed in the past, but they are designed mostly for specialists. Augmented reality has recently emerged as an interactive medium for exploring information in the real world and is well-suited for non-specialists. We propose a demo of a mobile augmented reality application that runs in the browser and consumes and displays linked open data in a friendly manner on top of the surroundings of the user.

1 Introduction

Due to the vast amount of data that was published in the latest years, linked data has become a popular research field. The Linking Open Data cloud¹ has grown from 12 datasets in 2007 to 295 datasets in 2011. One of the major research areas is related to the consumption of linked data by browsers designed specifically for this task. These browsing and visualization tools are required by tech users and lay users equally to easily retrieve information from the Web of Data. A recent survey [1] categorized these tools into text browsers and visualization-based browsers, with the latter being more suited for lay users, but also being the ones that are fewer and need more tweaking. However, browsers that make use of large quantity of linked (open) data and are well-suited for specific tasks are still under-researched.

Mobile augmented reality applications have recently emerged as interactive and usable tools for exploration of the surrounding world of a user [2]. We propose this medium as a suitable form of exploring geo-based linked data. This approach is not without its research challenges, mainly geodata integration, data quality assessment and provenance and trust issues [3]. We present a demo of a mobile augmented reality application that helps tourists to get a sense of the unfamiliar surroundings based on popular linked open data content sources that are integrated for this purpose. The current version of the demo is available online.²

¹ <http://lod-cloud.net/>

² <http://dev.cm.upt.ro/ar/>

2 Overview of the implementation

The application implements the crawling/data warehousing pattern. Figure 1 highlights the steps needed to build an augmented reality application that uses data from multiple sources.

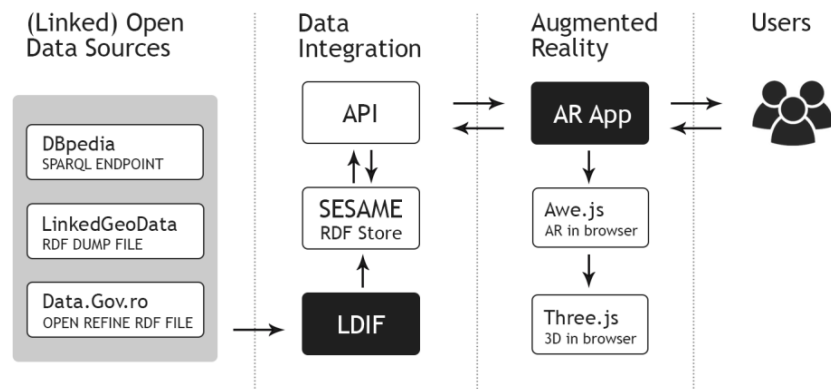


Fig. 1. Overview of the flow of information in the application

In the first step, a list of data sources is identified with the purpose of being used in an augmented reality application. The data sources have to contain information regarding places with geographic coordinates, with labels and descriptions, ideally in multiple languages, and with mapping links and specific categories.

In the second step, the data from these multiple data sources is collected and mapped to a consistent vocabulary. The identity resolution is resolved for different URIs addressing the same resource, a data quality assessment is provided, the merged data is stored into a RDF store and an API is provided for the AR application to obtain the desired information in JSON format.

The mobile augmented reality application, the third step, is browser based so there is no need for the user to download a standalone application from the store. Given the detected geolocation of the user, the application displays a set of Points of Interest (POIs) in the immediate vicinity of the user. The information about the POIs is consumed from the triple store, via the above-named API.

The first two steps are described in section three and the AR application in section four.

3 Linked Open Data Integration

In order to build a dataset usable in an augmented reality application, first of all we identified possible data sources that satisfied the requirements described above. The

datasets chosen for this demo application are extracted from DBpedia.org, LinkedGeoData.org and the Romanian Government Open Data portal³.

In order to collect, map and integrate the chosen datasets we used the powerful Linked Data Integration Framework (LDIF) [4]. From DBpedia we collected information relevant to the authors hometown, Timisoara, using the SPARQL Import module from LDIF. In the case of LinkedGeoData the SPARQL Import is unusable because the query cannot be limited to a geographic area. The solution was to generate a dump file for the desired area and using the Triple/Quad Dump Import module to load the data in LDIF. From the Romanian Government Open Data portal we used the museums dataset available only in CSV format. The Open Refine tool has been used to clean and convert the dataset into RDF format and the Triple/Quad Dump Import module to load the data in LDIF.

Data translation was carried out to obtain a consistent list of categories and to build geometry values, if missing. Identity resolution was employed using the Levenshtein distance with a threshold of 1 on labels and comparing the distance between POIs with a threshold of 100 meters. For quality assessment the metric was time closeness; this implies that only the most recent data is used in the data fusion step.

The resulted dataset was outputted by LDIF in an OpenRDF Sesame RDFS store and contained 25,000 statements for the city of Timisoara. To allow the AR application to query the data, an API was built that can query triples from the RDF store and return them in JSON format.

4 The Augmented Reality Web Application

The augmented solution chosen is a browser-based one. Modern web technologies (HTML, CSS, Javascript) combined with the current capabilities of mobile devices that are available in the browser (geolocation, camera, WebGL) have given rise to such solutions. Our demo uses the recently launched awe.js library,⁴ which builds an augmented reality layer on top of the three.js library, a 3D library working in the browser. Awe.js library is advertised to work both with location and marker based AR, on the latest versions of Chrome and Firefox on Android, as well as on devices such as Oculus Rift, Leap Motion and Google Glass. We successfully tested our web application so far on Chrome v35, Firefox v30 and Opera v22 on a Nexus 4 smartphone with Android 4.4.

The web application queries asynchronously the Sesame server (through a proxy, to overcome the same-origin policy) to get the required POIs, based on the location of the user, an area around it, where to search the points, and the desired category of interest. It then processes the list of retrieved points to place 3D pinpoints into the space, based on the category the POI belongs to and on the distance from the user to that POI. On touching a certain POI, the user is presented with a short snippet of information, which he can expand to read more about that place. Figure 2 shows some screenshots of the application.

³ <http://data.gov.ro>

⁴ <https://github.com/buildar/awe.js>

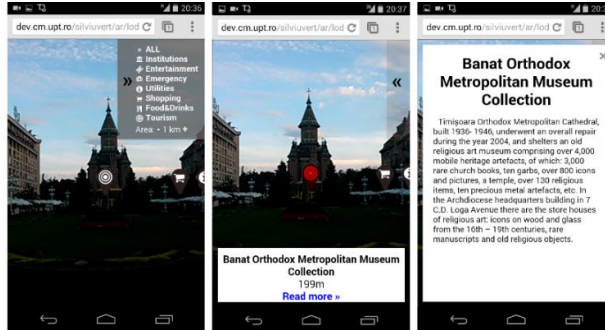


Fig. 2. Screenshots of the mobile browser application. On the left, a screenshot of the menu and multiple display of POIs. In the middle, the selected POI and a short snippet of information. On the right, the extended information for the same POI.

5 Conclusions and Future Work

In this demo paper we proposed a mobile augmented reality application that runs in the browser and consumes and displays linked open data. To accomplish this we used three open datasets, a powerful linked data integration framework, LDIF, an OpenRDF Sesame RDFS store and we built an API for extracting data from the store and delivering it to the mobile augmented reality browser application built on top of awe.js. As future work we intend to include: submenus (to be able to further refine the selection of the POIs), UI improvement, additional datasets, improved data quality.

Acknowledgements. This work was partially supported by the strategic grant POSDRU/159/1.5/S/137070 (2014) of the Ministry of National Education, Romania, co-financed by the European Social Fund – Investing in People, within the Sectoral Operational Programme Human Resources Development 2007-2013.

References

1. Dadzie, A.-S., Rowe, M.: Approaches to visualising linked data: A survey. *Semantic Web*, 2, 89–124 (2011).
2. Kounavis, C.D., Kasimati, A.E., Zamani, E.D.: Enhancing the Tourism Experience through Mobile Augmented Reality: Challenges and Prospects. *Int. J. Eng. Bus. Manag.* 4, (2012).
3. Vert, S., Vasii, R.: Relevant Aspects for the Integration of Linked Data in Mobile Augmented Reality Applications for Tourism. *The 20th International Conference on Information and Software Technologies (ICIST 2014)*, Druskininkai, Lithuania October 9 (2014) (accepted).
4. Schultz, A., Matteini, A., Isele, R., Mendes, P.N., Bizer, C., Becker, C.: LDIF-A Framework for Large-Scale Linked Data Integration. *21st International World Wide Web Conference (WWW 2012)*, Developers Track, Lyon, France (2012).

PLANET: Query Plan Visualizer for Shipping Policies against Single SPARQL Endpoints

Maribel Acosta¹, Maria-Esther Vidal², Fabian Flöck¹,
Simon Castillo², and Andreas Harth¹

¹ Institute AIFB, Karlsruhe Institute of Technology, Germany
{maribel.acosta, fabian.floeck, harth}@kit.edu

² Universidad Simón Bolívar, Venezuela
{mvidal, scastillo}@ldc.usb.ve

Abstract. Shipping policies allow for deciding whether a query should be executed at the server, the client or distributed among these two. Given the limitations of public SPARQL endpoints, selecting appropriate shipping plans is crucial for successful query executions without harming the endpoint performance. We present PLANET, a query plan visualizer for shipping strategies against a single SPARQL endpoint. We demonstrate the performance of the shipping policies followed by existing SPARQL query engines. Attendees will observe the effects of executing different shipping plans against a given endpoint.

1 Introduction and Overview

In the context of the Web of Data, endpoints are acknowledged as promising SPARQL server infrastructures to access a wide variety of Linked Data sets. Nevertheless, recent studies reveal high variance in the behavior of public SPARQL endpoints, depending on the queries posed against them [3]. One of the factors that impact on the endpoint performance is the type of shipping policy followed to execute a query.

Shipping policies [4] define the way that the workload of executing a query is distributed among servers and clients. Query-shipping policies conduct the execution of query operators at the server, while plans following data-shipping exploit the capacity of the client and execute the query operators locally. In contrast, hybrid approaches pose sub-queries and operators according to the complexity of the queries, and the server workload and availability. Current SPARQL query engines implement different policies. For example, FedX [5] implements a query-shipping strategy, executing the whole query at the endpoint, when the federation is comprised by one endpoint. ANAPSID [2] usually follows a hybrid-shipping strategy, it locally gathers the results of star-shaped sub-queries executed by the endpoint. To showcase an adaptive hybrid approach alongside FedX and ANAPSID, we also demonstrate SHEPHERD [1], an endpoint-tailored SPARQL client-server query processor that aims for reducing the endpoint workload and benefits the generation of hybrid shipping plans.

Analyzing the shipping policies followed to execute a query provides the basis not only to understand the behavior of an endpoint, but also allows for the development of endpoint-aware query processing techniques that preserve endpoint resources. The goal of the work presented here is to assist data consumers or data providers in understanding the effects of posing different shipping plans against an existing public SPARQL

endpoint. We introduce PLANET, a query plan visualizer for shipping strategies that provides an intuitive overview of the plan structure, the used shipping strategies as well as key metrics to understand the behavior of different engines when executing a query. PLANET is designed to shed light on the distribution of operator execution between client and server, which is crucial for investigating the type of plans that may lead to severe under-performance of endpoints. Attendees will observe the impact of different shipping strategies when queries are posed against a single endpoint. The demo is published at <http://km.aifb.kit.edu/sites/planet/>.

2 The PLANET Architecture

PLANET invokes SPARQL query engines that implement different shipping strategies. In this work, we studied the query-shipping plans produced by FedX, and the hybrid shipping plans of ANAPSID and SHEPHERD. The plan retrieved from each engine is processed by PLANET’s query plan parser, which translates the plans into JSON structures to encode the visualization data that will be consumed by the rendering module. Currently PLANET is able to parse plans generated by engines that use the Sesame¹ framework, or the ANAPSID or SHEPHERD internal structures.

The rendering module uses the “Collapsible Tree” layout of the D3.js JavaScript library² to generate the visualizations of plans produced by the SPARQL query processing engines. Figures 1(a) and 1(b) show snapshots of plans rendered by PLANET. Plan operator nodes are filled with different colors to distinguish whether the operator is executed by the engine (locally) or at the server (remotely), which allows to easily identify the type of shipping strategy followed in each plan.

The plan descriptor reports a set of metrics characterizing the shipping plans. Execution performance is measured by the execution time of the query and the number of results produced. In addition, the metric *hybrid ratio* measures the quantitative relation between the SPARQL operators executed at the local engine and the ones executed at the endpoint. The hybrid ratio of a plan p is calculated as follows:

$$hybridRatio(p) = \frac{clientOp(p) \cdot serverOp(p)}{totalOp(p) \cdot \max\{clientOp(p), serverOp(p)\}}$$

where $clientOp(p)$, $serverOp(p)$ stand for the number of operations executed for plan p at the client and server, respectively, and $totalOp(p) = clientOp(p) + serverOp(p)$. Note that the hybrid ratio for plans following data- or query-shipping strategies is zero.

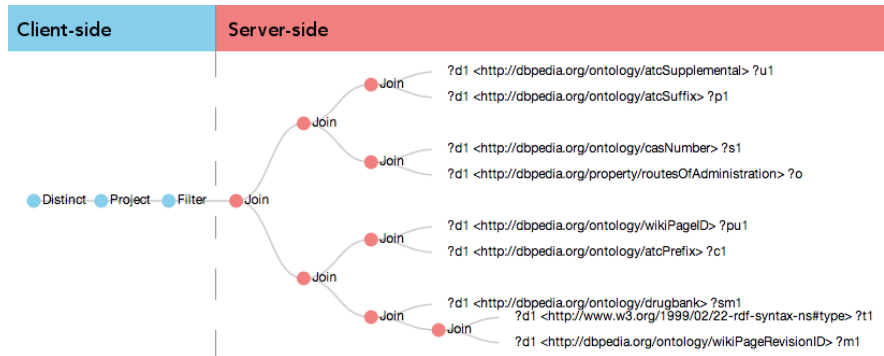
The output of PLANET is a set of plan visualizations and a summary report with the metrics computed for each plan.

3 Demonstration of Use Cases

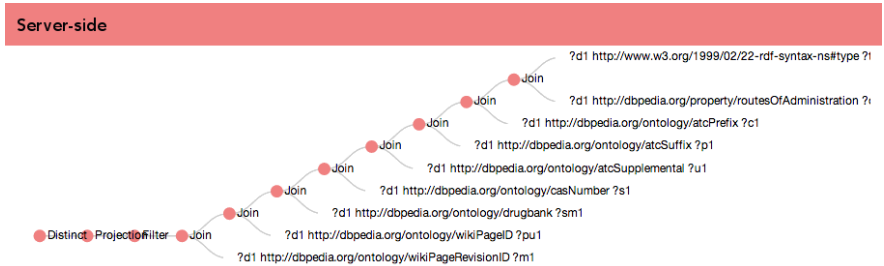
As an illustrating example, consider the following query included in our online demo: GP Query 2 against the DBpedia endpoint, comprised of 9 triple patterns and one FILTER operator. Figure 1 reports on the plans depicted by PLANET for the previous query. The plan reported in Figure 1(a) follows a hybrid shipping strategy where the quantitative relation between the SPARQL operators executed locally and the ones

¹ <http://www.openrdf.org/>

² <http://d3js.org/>



(a) Hybrid Shipping Plan. Sub-query under red bar is posed against DBpedia endpoint; Operators under blue bar are executed at the client side



(b) Query Shipping Plan. The whole query is posed to the DBpedia endpoint

Fig. 1. Plans against the DBpedia endpoint. Blue circles represent operators executed locally; red circles correspond to operators that will be posed against the endpoint

posed against the DBpedia endpoint or *hybrid ratio* is 0.27; the execution time is 0.63 secs. and one tuple is retrieved. On the other hand, when the query shipping-based plan presented in Figure 1(b) is executed, the execution time is 1.44 secs. and no answer is received. Finally, if the query is executed directly against the endpoint, the answer is effectively retrieved but the execution time is 6.91 secs. For the three different engines, we can observe that the combined performance of engine and endpoint deteriorates as the number of operators posed against the endpoint increases.

In congruence with the previous example result, the following research questions arise: (i) is the observed behavior due to limitations of the endpoints? Or (ii) is this behavior caused by the shipping plan followed during query execution? As part of this demo, we will visualize characteristics of different plans and public endpoints that provide evidence enabling to answer our research questions. By this time, we have performed a comprehensive study of the execution of 70 queries against seven different public SPARQL endpoints. We selected the query targets from the list of endpoints monitored by the SPARQLES tool [3] and classified them in quartiles. These included two high-performant endpoints (Top25% Quartile), two medium-performant ((25%; 50%] Quartile), and three second-least performant endpoints ((50%;75%] Quartile). We crafted ten SPARQL queries for each endpoint; five are composed of basic graph pat-

terns (BGP queries), and the others comprise any SPARQL graph pattern (GP queries). Attendees of the demo have the possibility of analyze the results of executing these queries currently loaded in the system, or visualize the plans of their own SPARQL queries. Query plans are computed on-the-fly while the reported results were computed off-line to facilitate demonstration. We will demonstrate the following use cases:

Effects of Shipping Policies in BGP Queries. We show that in setups as the one reported in Figure 1 and where endpoints receive large number of concurrent request per day, i.e., the endpoint is in the (50%;75%] quartile, hybrid-shipping policies can reduce execution time and the results surpasses the 79% of the answers retrieved by query-shipping plans. For high- and medium-performant endpoints, there is a trade-off between execution time and size of retrieved answers. Nevertheless, in all the queries the effectiveness of the endpoints is increased by up to one order of magnitude, i.e., the number of answers produced per second following a hybrid-shipping plan can be up 20 times the number of answers produced by a query-shipping plan.

Effects of Shipping Policies in GP Queries. We observed that for highly work-loaded endpoints, 90% of hybrid-shipping plans reduce execution time by up two orders of magnitude. Hybrid plans generated by SHEPHERD achieved the highest performance on DBpedia. For endpoints in other quartiles, a competitive performance between hybrid- and query-shipping plans is observed, but hybrid plan performance never significantly deteriorates. This suggests that hybrid-shipping policies are appropriate to achieve reasonable performance while shifting load from the server to the client.

4 Conclusions

PLANET visualizes the impact of shipping policies and provide the basis for the understanding of the conditions that benefit the implementation of hybrid shipping plans, e.g., attendees will be able to observe that for non-selective queries hybrid plans significantly outperforms the others. Thus, PLANET facilitates the analysis of the behavior of public endpoints, as well as, the development of scalable real-world client-server applications against single SPARQL endpoints.

Acknowledgements

The authors acknowledge the support of the European Community's Seventh Framework Programme FP7-ICT-2011-7 (XLike, Grant 288342).

References

1. M. Acosta, M.-E. Vidal, F. Flock, S. Castillo, C. Buil-Aranda, and A. Harth. Shepherd: A shipping-based query processor to enhance sparql endpoint performance. In *ISWC Poster Track*, 2014.
2. M. Acosta, M.-E. Vidal, T. Lampo, J. Castillo, and E. Ruckhaus. Anapsid: an adaptive query processing engine for SPARQL endpoints. In *ISWC*, pages 18–34, 2011.
3. C. B. Aranda, A. Hogan, J. Umbrich, and P.-Y. Vandenbussche. SPARQL web-querying infrastructure: Ready for action? In *ISWC*, pages 277–293, 2013.
4. M. J. Franklin, B. T. Jónsson, and D. Kossmann. Performance tradeoffs for client-server query processing. In *SIGMOD Conference*, pages 149–160, 1996.
5. A. Schwarte, P. Haase, K. Hose, R. Schenkel, and M. Schmidt. Fedx: Optimization techniques for federated query processing on linked data. In *ISWC*, pages 601–616, 2011.

High Performance Linked Data Processing for Virtual Reality Environments

Felix Leif Keppmann¹, Tobias Käfer¹, Steffen Stadtmüller¹, René Schubotz²,
and Andreas Harth¹

¹ Karlsruhe Institute of Technology (KIT)
{felix.leif.keppmann, tobias.kaefer, steffen.stadtmueller,
andreas.harth}@kit.edu
² Airbus Group
rene.schubotz@eads.net

1 Introduction

The success of Linked Data (LD) [1] has enabled an environment in which application data can easily be enriched by the abundance of available information on the Web. Many recent approaches of the Linked Data community go beyond the mere exposure of static data and propose the combination of Linked Data and Representational State Transfer (REST) [3, 5, 7] to enable dynamic systems. However, in highly dynamic environments, where near real-time data integration and processing with high update frequencies are required, the perceived overhead of Linked Data query processing and stateless communication pattern often prevents the adoption of resource state exchange-oriented systems.

Nevertheless, in our demonstration, we show a Virtual Reality (VR) information system that leverages the REST principles and the integration capabilities of LD. We specifically chose a VR setting, because it requires very low latency [2] in order to enable a natural interaction of the user with the system. Our system consists of loosely coupled components [4] as implicated by REST, and provides an interactive experience by seamlessly integrating existing LD sources from the Web as well as high dynamic body tracking data in a VR environment.

We show how sensor data exposed as LD, can be processed with high update frequencies and be rendered in a VR environment. Constantly evaluated queries are employed to realise both gesture recognition and collision detection of objects in the VR. Derived actions like data retrieval from the Web and the subsequent integration of the retrieved data with the sensor data are performed on-the-fly. With our system we contribute by demonstrating:

- the applicability of Linked Data in a VR environment
- the feasibility of a REST-based distributed system with high frequencies
- the capability to execute high frequency on-the-fly declarative integration of Linked Data in a REST environment

In the following we present the experience provided by our demonstration from both a user’s and a technological point of view (Section 2). Afterwards, we elaborate on the underlying technologies (Section 3) and conclude shortly at the end (Section 4).



Fig. 1. Depth Video with Skeleton Tracking and Demo System Visualization

2 Demonstration

Our demonstration system let the user experience an interactive, integrated and responsive Virtual Reality. The system displays an avatar representing the user and information from various sources integrated on-the-fly in response to user commands. In our system, the user inputs commands to the system via Natural Interaction (NI), e.g. by interacting with the system via movements and gestures. The user in our set-up stands in front of a motion tracking sensor that tracks all parts of the body. As the user moves, for example, a hand to form a specific pose, the system executes a particular action. Figure 1 shows both, a visualization of the depth video input data (including skeleton tracking) and the VR which the user is remote controlling via NI.³

The user is represented in the VR by a human-like avatar (blue in Figure 1). Each recognized joint of the user’s skeleton is mapped to the avatar, e.g. a knee of the user is mapped to knee of the avatar and moved accordingly. Instead of walking on a floor, the avatar is placed on a map (in Figure 1 the city of Karlsruhe in Germany) which will move in a specific direction if the avatar steps on the corresponding border of the map. Further, Points of Interest (PoIs) are visualized on the map, e.g. important buildings or upcoming concerts in the area (represented by red arrows in Figure 1). Via gestures of the user at a PoI, more detailed information will be requested, integrated on-the-fly and displayed in the VR. Thereby, a user is able to navigate through the map by walking and requesting additional information with gestures.

Beside the user experience of our system, we demonstrate from a technological point of view, loosely coupled components representing several internal and external data sources and sinks with a variety of exposed update frequencies, which are integrated on-the-fly in a declarative manner. All data sources and data sinks expose their data and functionality as LD via REST interfaces. The data sources include sources of a relatively static nature, e.g. map data or PoIs, and highly dynamic data sources, e.g. the sensor data of the body-tracking video sensor. Nevertheless, all sources and sinks are integrated via one component without the need for programmatic hard-coded integration of the interfaces.

³ More screenshots and videos of the demonstration system are available at: <http://purl.org/NET/HighPerfLDVR>

In particular, a set of rules defined in a declarative rule language specifies the data flow between sources and sinks. A corresponding engine handles the query processing, evaluates the rule set in each cycle and executes low level tasks, e.g. retrieving data from the REST interface of the video sensor and a PoIs service, transformation and integration in the target schema and updating or creating resources in the REST interface of the visualization. Currently, our demonstration system processes data with a frequency of 20 Hz under high load up to 28 Hz under low load, which is close to the ideal 30 Hz update frequency at which the body tracking sensor updates.

3 Technology

We use Linked Data-Fu (LD-Fu) [6] as a central component for integration of data sources and sinks. LD-Fu is both a declarative rule language and an engine to handle the execution of programs consisting of these rules. The LD-Fu engine is able to interact with LD resources via REST interfaces. Rules defined in a program are evaluated and the interactions defined in the rule body are executed if the conditions are met. These interactions can include the retrieval of data from LD resources, the integration of data and the manipulation of LD resources. We developed the engine as multi-threaded forward-chaining data processor, thereby supporting fast parallel data processing with reasoning capabilities.

In our demonstration system LD-Fu handles 1) the data flow between body tracking of the depth sensor and visualization on the screen or wall, 2) collision detection in the interaction of the user with virtual objects, 3) execution of actions as result of the user interaction and 4) the integration of additional data sources on the web, e.g. information about PoIs or map data. All these interactions are defined as rule sets in LD-Fu programs.

With Natural Interaction via REST (NIREST) we expose body tracking data as LD via a REST interface. It utilizes data extracted from depth video sensors, e.g. Microsoft Kinect⁴ devices. The position information of recognized people in the depth video, their skeleton data and device metadata are exposed as LD resources on the REST interface and are updated at a frequency of 30 Hz. We developed NIREST in Java as application container that encapsulates all required components and is deployable on application servers. The application is build on top of the OpenNI⁵ framework and the NiTE middleware. OpenNI is as an Open Source framework providing low-level access to colour video, depth video and audio data of sensor devices. NiTE acts as middleware on top of OpenNI and provides body, skeleton and hand tracking.

We use NIREST in our demonstration system as 1) high dynamic data source for 2) body positions and for 3) the position of skeleton joint points of all people in front of the sensor. LD-Fu programs use this data via the REST interface for collision detection and visualizations in the virtual reality.

⁴ <http://www.microsoft.com/en-us/kinectforwindows/>

⁵ <https://github.com/opencv/opencv>

Our user interface is based on jMonkey⁶, an Open Source 3D engine for the development of games in Java. We combine jMonkey with a REST interface to expose data about objects in the 3D scene graph using LD resources. A scene graph is a data structure in VR engines which represents all objects in the VR as well as their interrelations. The LD interface on top of jMonkey allows for the retrieval and modification of data about the 3D scene graph nodes. In our demonstration we employ the visualization based on jMonkey as user interface which can be displayed on a monitor or on a wall using a beamer.

4 Conclusion

With our system we demonstrate data integration in a highly dynamic environment using Semantic Web technologies. We applied successfully the LD and REST paradigms that facilitate on-the-fly declarative data integration in our VR scenario. Moreover, the user will be part of the demonstration system and be able to control the integration and visualization of information via natural interaction all programmed using a declarative rule language.

Acknowledgements

This work was supported by the German Ministry of Education and Research (BMBF) within the projects ARVIDA (FKZ 01IM13001G) and Software-Campus (FKZ 01IS12051) and by the EU within the projects i-VISION (GA #605550) and PlanetData (GA #257641).

References

1. Bizer, C., Heath, T., Berners-Lee, T.: Linked Data – The Story So Far. *International Journal on Semantic Web and Information Systems* 5(3), 1–22 (2009)
2. (ITU), I.T.U.: End-user multimedia QoS categories. ITU-T Recommendation G.1010, International Telecommunication Union (ITU) (2001)
3. Krummenacher, R., Norton, B., Marte, A.: Towards Linked Open Services and Processes. In: *Proceedings of the Future Internet Symposium*. Springer Berlin Heidelberg (2010)
4. Pautasso, C., Wilde, E.: Why is the Web Loosely Coupled? A Multi-Faceted Metric for Service Design. In: *Proceedings of the International World Wide Web Conference* (2009)
5. Speiser, S., Harth, A.: Integrating Linked Data and Services with Linked Data Services. In: *Proceedings of the Extended Semantic Web Conference* (2011)
6. Stadtmüller, S., Speiser, S., Harth, A., Studer, R.: Data-Fu: A Language and an Interpreter for Interaction with Read/Write Linked Data. In: *Proceedings of the International World Wide Web Conference* (2013)
7. Verborgh, R., Steiner, T., van Deursen, D., van de Walle, R., Gabarró Vallès, J.: Efficient Runtime Service Discovery and Consumption with Hyperlinked RESTdesc. In: *Proceedings of the International Conference on Next Generation Web Services Practices* (2011)

⁶ <http://jmonkeyengine.org/>

Analyzing Relative Incompleteness of Movie Descriptions in the Web of Data: A Case Study

Wancheng Yuan¹, Elena Demidova², Stefan Dietze², Xuan Zhou¹

¹DEKE Lab, MOE. Renmin University of China. Beijing, China
wancheng.yuan@ruc.edu.cn, zhou.xuan@outlook.com

²L3S Research Center and Leibniz University of Hanover, Germany
{demidova, dietze}@L3S.de

1 Introduction and Approach

In the context of Linked Open Data (LOD) [3], datasets are published or updated frequently, constantly changing the landscape of the Linked Data Cloud. In this paper we present a case study, investigating relative incompleteness among sub-graphs of three Linked Open Data (LOD) datasets (DBpedia (dbpedia.org), Freebase (www.freebase.com), LinkedMDB (www.linkedmdb.com)) and propose measures for relative data incompleteness in LOD. The study provides insights into the level of accuracy and actual conflicts between different LOD datasets in a particular domain (movies). In addition, we investigate the impact of the neighbourhood size (i.e. path length) under consideration, to better understand the reliability of cross-dataset links.

Fig. 1 presents an example of relative incompleteness in the representation of movie entity “Holy Smoke!” in DBpedia and Freebase. In this example, the difference between the actor sets indicates the “Movie.Actor” property might be incomplete. As we do not know the exact complete set of actors and the noise observed in linked datasets interferes completeness estimation, we call this phenomenon relative incompleteness. If we follow the “Movie.Cinematographer” link in the data graphs of the two datasets, we can observe further relative incompleteness in its “birthPlace” property.

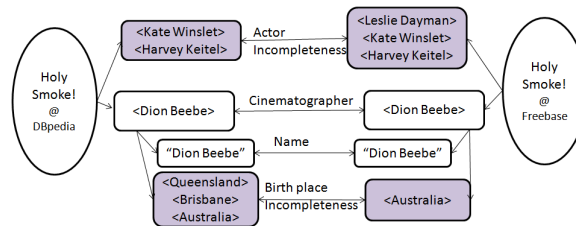


Fig. 1. Representation of the movie “Holy Smoke!” in DBpedia and Freebase.

In this paper we discuss incompleteness related measures that can be obtained by pairwise dataset comparisons exploiting entity co-resolution across

these datasets and apply these measures in a case study. The basis for the proposed measures is the assumption that dataset specific differences in representation of equivalent entities, and in particular the values of multi-value properties, can provide valuable insights into the relative incompleteness of these datasets. To facilitate discovery of relative incompleteness, we assume correct schema mapping and make use of known equivalent entities. In the context of LOD, absolute incompleteness is difficult to judge, as it is difficult to obtain the ground truth of absolute completeness. Therefore, we choose to estimate relative incompleteness of the properties by following paths of limited lengths in the data graphs.

By i^{th} -**Order property**, we mean a property that can be reached from the target entity through a path of length i in the data graph. For instance, “Movie.Actor” is a 1st Order property in Fig. 1, while “Movie.Cinematograher.Name” is a 2nd Order property. Then, we define i^{th} -Order Value Incompleteness as follows:

i^{th} -Order Value Incompleteness (D_x, D_y, P) between the pair of datasets D_x, D_y with respect to a i^{th} -Order multi-value property P is the proportion of entities in D_x and D_y having different values in P .

As P is a multi-value property, the difference on P usually indicates that at least one of the datasets does not provide sufficient information on P . In Fig. 1, we observe a 2nd Order Value Incompleteness in the “Movie.Cinematographer.birthPlace” property. To determine equivalent values, we rely on direct value comparisons and identity links.

Considering the LOD cloud as a large interlinked knowledge graph, relative incompleteness of data across different datasets is a crucial and often under-investigated issue. Relative incompleteness can result e.g. from extraction errors, lack of source maintenance [4], imprecise identity links [2] as well as incompatibilities in schemas and their interpretation (as we observed in this study). In the literature, detection and resolution of data inconsistency has been studied in the context data fusion [1]. However, the corresponding methods for the assessment of LOD datasets are underdeveloped. The measures proposed in this paper can help judging relative agreement of datasets on certain properties and thus support source selection. E.g. these statistics can support identification of sources with the highest relative agreement as well as the sources containing complementary information, dependent on the particular scenario.

2 A Case Study

Datasets and Schemas: We used the latest version of three datasets from LOD - LinkedMDB (LMDB), DBpedia and Freebase. The LMDB dataset contains eight concepts about movies, such as *Movie*, *Actor* and *Country*, and more than 200,000 records. The DBpedia and Freebase datasets contain around 150,000 and 1,000,000 movie records respectively. To perform the study, we randomly selected 200 Movie and 200 Actor entities shared between these datasets. To establish the relationship of entities across the three datasets, we obtained the existing interlinking information (i.e., the *owl:sameAs* predicate) of the Movie

entities across all the three datasets as well as the Actor entities in the DBpedia and Freebase. We manually established schema mappings between the Movie and Actor concepts and their properties among the datasets.

Evaluation Results: We computed the 1st and 2nd Order Value Incompleteness for each property in each pair of datasets. Table 1 presents an aggregated 1st and 2nd Order Value Incompleteness results for the Movie and Actor entities. In this result, if there is a single property that is incomplete on an entity, we would count this entity as incomplete. As we can see in Table 1, the relative incompleteness in the DBpedia/Freebase pair reaches 100% in the first order and 89% in the second order, meaning that all the Movie entities in the datasets are affected by incompleteness issues. The overall 1st Order Incompleteness of the Movie entities in the other dataset pairs is also pretty high, e.g., 70% for LMDB/Freebase and 56% for LMDB/DBpedia.

Table 1. Aggregated Incompleteness for Movie and Actor Entities

Datasets	Movie 1 st O. Incompleteness	Movie 2 nd O. Incompleteness	Actor 1 st O. Incompleteness
LMDB/DBpedia	0.56	n/a	n/a
LMDB/Freebase	0.70	n/a	n/a
DBpedia/Freebase	1.00	0.89	0.76

Table 2 presents the details of the evaluation for each property. As we can see in the Table 2, the highest relative incompleteness among all datasets is observed in the DBpedia/Freebase pair on the property Actor, whose incompleteness is 73%. This is because DBpedia tends to include only key people in a movie, whereas Freebase tends include more complete actor lists. For example, for the movie “Stand by Me”, DBpedia lists only five actors: Wil Wheaton, Kiefer Sutherland, River Phoenix, Corey Feldman, and Jerry O’Connell. The “starring” property of Freebase includes many more actor names such as Gary Riley, Bradley Gregg, Frances Lee McCain, etc. We also observed that the “starring” property sometimes mixes actor and character names in a movie. For example, for “Stand by Me”, it includes characters Teddy Duchamp and Waitress. Regarding the LMDB/Freebase, the incompleteness on the properties Producer, Release Date and Actor are 30%, 26% and 19% respectively. LMDB/DBpedia shows a similar distribution, i.e. 29%, 11% and 15%, on the same properties.

Table 2. 1st Order Value Incompleteness of Selected Movie Properties

Dataset	Release Date	Country	Language	Actor	Director	Writer	Editor	Producer
LMDB/DBpedia	0.11	0.02	0.16	0.15	0.02	n/a	n/a	0.29
LMDB/Freebase	0.26	0.15	0.24	0.19	0.02	n/a	n/a	0.30
DBpedia/Freebase	0.21	0.12	0.25	0.73	0.04	0.25	0.08	0.36

An exemplary evaluation performed on the Actor entities indicates a similar tendency as for the Movie type, with 76% incompleteness in the first order in the DBpedia/Freebase pair. While Actor entities always agree on the names and very often on the birth dates (which makes us think that the existing interlinking of Actor entities was established using these properties), they frequently disagree on the property of *birthPlace*. This is because the values of property *birthPlace* from DBpedia are much more detailed than those from Freebase. DBpedia typically includes a country name in an address, whereas Freebase does not. For example, the place of birth of the person “Len Wiseman” from DBpedia is “Fremont, California, United States”, while that from Freebase is “Fremont, California”. As a result, we observe an increased incompleteness in the property *birthPlace*. Interestingly, the property *deathPlace* is much less incomplete, as most actors listed in these databases are still alive (we regard null values as incomparable).

3 Conclusions and Outlook

In this paper we presented measures to automatically evaluate relative incompleteness in linked datasets and applied these measures in a case study. From the experiment performed using three linked datasets in the movie domain we can conclude that incompleteness is a very common phenomenon in these datasets, and its number increases significantly with increasing order, i.e. increase of investigated entity neighbourhood. The main causes of relative incompleteness observed during our experiment are due to different interpretations of properties in the datasets. Our method of classification and identification of quality issues provides not only insights into the level of agreement between datasets but also into the overall quality of datasets. In future work we intend to extend these approaches to infer knowledge about the correctness and agreement of schemas.

Acknowledgments

This work was partially funded by the NSFC Project No. 61272138, ERC under ALEXANDRIA (ERC 339233), the COST Action IC1302 (KEYSTONE) and 973 Program Project of China (Grant Nr.: 2012CB316205).

References

1. X. L. Dong and F. Naumann. Data fusion: resolving data conflicts for integration. *Proc. VLDB Endow.*, 2(2):1654–1655, 2009.
2. H. Halpin, P. J. Hayes, J. P. McCusker, D. L. McGuinness, and H. S. Thompson. When owl: sameas isn’t the same: An analysis of identity in linked data. In *Proc. of the 9th International Semantic Web Conference, ISWC 2010, Shanghai*, 2010.
3. T. Heath and C. Bizer. *Linked Data: Evolving the Web into a Global Data Space (1st edition)*. *Synthesis Lectures on the Semantic Web: Theory and Technology*, 1:1, 1-136. Morgan & Claypool, 2011.
4. P. N. Mendes, H. Mühleisen, and C. Bizer. Sieve: linked data quality assessment and fusion. In *Proc. of the 2012 Joint EDBT/ICDT Workshops, Berlin*, 2012.

A Semantic Metadata Generator for Web Pages Based on Keyphrase Extraction

Dario De Nart, Carlo Tasso, Dante Degl'Innocenti

Artificial Intelligence Lab
Department of Mathematics and Computer Science
University of Udine, Italy
{dario.denart,carlo.tasso}@uniud.it, dante.deglinnocenti@spes.uniud.it

Abstract. The annotation of documents and web pages with semantic metadata is an activity that can greatly increase the accuracy of Information Retrieval and Personalization systems, but the growing amount of text data available is too large for an extensive manual process. On the other hand, automatic keyphrase generation and wikification can significantly support this activity. In this demonstration we present a system that automatically extracts keyphrases, identifies candidate DBpedia entities, and returns as output a set of RDF triples compliant with the Opengraph and the Schema.org vocabularies.

1 Introduction

In the last few years we have witnessed the rapid growth of the Semantic Web and all its related technologies, in particular the ones that allow the embedding of semantic data inside the HTML markup of Web pages, such as RDFa. Recent studies highlight how a significant part of the most visited pages of the Web is annotated with semantic data and this number is expected to grow in the near future. However, up to now, the majority of such metadata is manually authored and maintained by the owners of the pages, especially those associated with textual content (such as articles and blog posts). Keyphrase Extraction (herein KPE) and Wikification can greatly ease this task, by identifying automatically relevant concepts in the text and Wikipedia/DBpedia entities to be linked. In this demonstration we propose a system for semantic metadata generation based on a knowledge-based KPE and Wikification phase and a subsequent rule-based translation of extracted knowledge into RDF ¹. Generated metadata adhere to the Opengraph and the Schema.org vocabularies which currently are, according to a recent study [2], wide-spread on the Web.

2 Related Work

Several authors in the literature have already addressed the problem of extracting keyphrases (herein KPs) from natural language documents and a wide range of

¹ A live demo of the system can be found at <http://goo.gl/beKJu5> and can be accessed by logging as user “guest” with password “guest”

approaches have been proposed. The authors of [11] identify four types of KPE strategies:

- *Simple Statistical Approaches*: mostly unsupervised techniques, considering word frequency, TF-IDF or word co-occurrence [8].
- *Linguistic Approaches*: techniques relying on linguistic knowledge to identify KPs. Proposed methods include lexical analysis [1], syntactic analysis [4], and discourse analysis [6].
- *Machine Learning Approaches*: techniques based on machine learning algorithms such as Naive Bayes classifiers and SVM. Systems such as KEA [10], LAKE [3], and GenEx [9] belong to this category.
- *Other Approaches*: other strategies exist which do not fit into one of the above categories, mostly hybrid approaches combining two or more of the above techniques. Among others, heuristic approaches based on knowledge-based criteria [7] have been proposed.

Automatic semantic data generation from natural language text has already been investigated as well and several knowledge extraction systems already exist [5], such as OpenCalais², AIDA³, Apache Stanbol⁴, and NERD⁵.

3 System Overview

The proposed system includes three main modules: a Domain Independent KPE module (herein DIKPE), a KP Inference module (KPIM), and a RDF Triple Builder (RTB). Our KPE technique exploits a knowledge-based strategy. After a candidate KP generation stage, candidate KPs are selected according to various features including statistic (such as word frequency), linguistic (part of speech analysis), meta-knowledge based (life span in the text, first and last occurrence, and presence of specific tags), and external-knowledge based (existence of a match with a DBpedia entity) ones. Such features correspond to different kinds of knowledge that are involved in the process of recognizing relevant entities in a text. Most of such features are language-independent and the modular architecture of DIKPE allows an easy substitution of language-dependent components, making our framework language-independent. Currently English and Italian languages are supported.

The result of this KPE phase is a set of relevant KPs including DBpedia matches, hence providing a partial wikification of the text. Such knowledge is used by the KPIM for a further step of KP generation, in which a new set of potentially relevant KPs not included in the text is inferred exploiting the link structure of DBpedia. Properties such as *type* and *subject* are considered in order to discover concepts possibly related to the text. Finally, the extracted and the inferred KPs are used by the RTB to build a set of Opengraph and Schema.org triples. Due to

² <http://www.opencalais.com/>

³ www.mpi-inf.mpg.de/yago-naga/aida/

⁴ <https://stanbol.apache.org/>

⁵ <http://nerd.eurecom.fr/>

the simplicity of the adopted vocabularies, this task is performed in a rule-based way. The rdf fragment to be generated, in fact, is considered by the RTB as a template to fill according to the data provided by the DIKPE and the KPIM.

4 Evaluation and Conclusions

In order to support and validate our approach several experiments have been performed. Due to the early stage of development of the system and being the KP generation the critical component of the systems, testing efforts were focused on assessing the quality of generated KPs. The DIKPE module was benchmarked against the KEA algorithm on a set of 215 English documents labelled with keyphrases generated by the authors and by additional experts. For each document, the KP sets returned by the two compared systems were matched against the set of human generated KPs. Each time a machine-generated KP matched a human-generated KP, it was considered a correct KP; the number of correct KPs generated for each document was then averaged over the whole data set. Various machine-generated KP set sizes were tested. As shown in Table 1, the DIKPE system significantly outperformed the KEA baseline. A user evaluation

Table 1. Performance of DIKPE compared to KEA.

Extracted Keyphrases	Average number of correct KPs	
	KEA	DIKpE
7	2.05	3.86
15	2.95	5.29
20	3.08	5.92

of the perceived quality of generated KPs was also performed: a set of 50 articles was annotated and a pool of experts of various ages and gender was asked to assess the quality of generated metadata. Table 2 shows the results of the user evaluation.

Table 2. User evaluation of generated keyphrases.

Evaluation	Frequency
Good	56,28%
Too Generic	14,72%
Too Specific	2,27%
Incomplete	9,85%
Not Relevant	9,85%
Meaningless	7,03%

Evaluation is, however, still ongoing: an extensive benchmark with more complex Knowledge Extraction systems is planned, as well as further enhancements

such as inclusion of more complex vocabularies and integration with the Apache Stanbol framework.

References

1. Barker, K., Cornacchia, N.: Using noun phrase heads to extract document keyphrases. In: *Advances in Artificial Intelligence*, pp. 40–52. Springer (2000)
2. Bizer, C., Eckert, K., Meusel, R., Mühleisen, H., Schuhmacher, M., Völker, J.: Deployment of rdfa, microdata, and microformats on the web—a quantitative analysis. In: *The Semantic Web—ISWC 2013*, pp. 17–32. Springer (2013)
3. DAvanzo, E., Magnini, B., Vallin, A.: Keyphrase extraction for summarization purposes: The lake system at duc-2004. In: *Proceedings of the 2004 document understanding conference* (2004)
4. Fagan, J.: Automatic phrase indexing for document retrieval. In: *Proceedings of the 10th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. pp. 91–101. SIGIR '87, ACM, New York, NY, USA (1987), <http://doi.acm.org/10.1145/42005.42016>
5. Gangemi, A.: A comparison of knowledge extraction tools for the semantic web. In: *The Semantic Web: Semantics and Big Data*, pp. 351–366. Springer (2013)
6. Krapivin, M., Marchese, M., Yadrantsau, A., Liang, Y.: Unsupervised key-phrases extraction from scientific papers using domain and linguistic knowledge. In: *Digital Information Management, 2008. ICDIM 2008. Third International Conference on*. pp. 105–112 (Nov 2008)
7. Liu, Z., Li, P., Zheng, Y., Sun, M.: Clustering to find exemplar terms for keyphrase extraction. In: *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1 - Volume 1*. pp. 257–266. EMNLP '09, Association for Computational Linguistics, Stroudsburg, PA, USA (2009), <http://dl.acm.org/citation.cfm?id=1699510.1699544>
8. Matsuo, Y., Ishizuka, M.: Keyword extraction from a single document using word co-occurrence statistical information. *International Journal on Artificial Intelligence Tools* 13(01), 157–169 (2004)
9. Turney, P.D.: Learning algorithms for keyphrase extraction. *Information Retrieval* 2(4), 303–336 (2000)
10. Witten, I.H., Paynter, G.W., Frank, E., Gutwin, C., Nevill-Manning, C.G.: Kea: Practical automatic keyphrase extraction. In: *Proceedings of the fourth ACM conference on Digital libraries*. pp. 254–255. ACM (1999)
11. Zhang, C.: Automatic keyword extraction from documents using conditional random fields. *Journal of Computational Information Systems* 4(3), 1169–1180 (2008), <http://eprints.rclis.org/handle/10760/12305>

A Multilingual SPARQL-Based Retrieval Interface for Cultural Heritage Objects

Mariana Damova¹ and Dana Dannélls² and Ramona Enache³

¹ Mozaika, Bulgaria

mariana.damova@mozajka.co

² Språkbanken, University of Gothenburg

dana.dannells@svenska.gu.se

³ Department of Computer Science and Engineering, University of Gothenburg

ramona.enache@cse.gu.se

1 Introduction

In this paper we present a multilingual SPARQL-based [1] retrieval interface for querying cultural heritage data in natural language (NL). The presented system offers an elegant grammar-based approach which is based on Grammatical Framework (GF) [2], a grammar formalism supporting multilingual applications. Using GF, we are able to present a cross-language SPARQL grammar covering 15 languages and a cross-language retrieval interface that uses this grammar for interacting with the Semantic Web⁴. To our knowledge, this is the first implementation of SPARQL generation and parsing via GF that is published as a knowledge representation infrastructure-based prototype.

Querying the Semantic Web in natural language, more specifically, using English to formulate SPARQL queries with the help of controlled natural language (CNL) syntax has been developed before [3,4]. Such approaches, based on verbalization methods are adequate for English, but in a multilingual setting where major challenges such as lexical and structural gaps become prominent [5], grammar-based approaches are preferable. The work presented here complements the method proposed by Lopez et al. [6] in that it faces the challenges in realizing NL in real world systems, not only in English, but also in multiple languages.

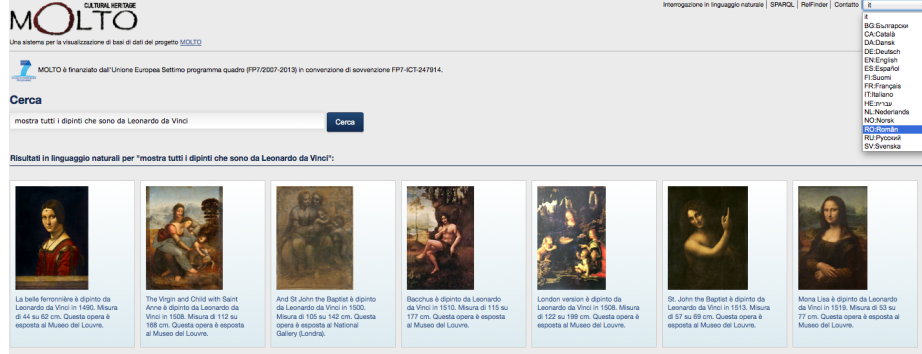
2 An Interface for Multilingual Queries

Our system follows the approach of the Museum Reason-able View (MRV) of Linked Open Data (LOD) [7]. It provides a unified access to the cultural heritage sources including LOD from DBpedia,⁵ among other sources.

⁴ A demo is available from the following page: <http://museum.ontotext.com/>

⁵ <http://dbpedia.org>

Fig. 1. Demo of the natural language query “show all paintings that are by Leonardo da Vinci” in Italian.



The query grammar for this data covers the nine central classes: title, painter, type, colour, size, year, material, museum, place and the major properties describing the relationship between them: `hasCreationDate`, `fromTimePeriodValue`, `toTimePeriodValue`, `hasMaterial`, `hasTitle`, `hasDimension`, `hasCurrentLocation`, `hasColour`. The set of SPARQL queries we cover include the famous five WH questions: *who*, *where*, *when*, *how*, *what*. Table 1 shows some NL queries and their mappings to query variables in SPARQL.

NL Query	SPARQL
Where is Mona Lisa located?	<code>:hasCurrentLocation ?location</code>
What are the colours of Mona Lisa?	<code>:hasColour ?colour</code>
Who painted Mona Lisa?	<code>:createdBy ?painter</code>
When was Mona Lisa painted?	<code>:hasCreationDate ?crdat</code>
How many paintings were painted by Leonardo da Vinci?	<code>?(count(distinct ?painting) as ?count)</code>

Table 1. Queries and query variables

The NL to SPARQL mapping is implemented as a transformation table, which could be extended to cover larger syntactic question variations.

The grammar has a modular structure with three main components: (1) lexicon modules covering ontology classes and properties; (2) data module covering ontology instances; and (3) query module covering NL questions and SPARQL query patterns. It supports NL queries in 15 languages, including: Bulgarian, Finnish, Norwegian, Catalan, French, Romanian, Danish, Hebrew, Russian, Dutch, Italian, Spanish, English, German and Swedish. The system relies on GF grammars, treating SPARQL as yet another language. In the same manner as NL generation, SPARQL patterns are encoded as grammar rules. Because of this compact representation within the same grammar, we can achieve parallel translations between any pair of the 15 languages and SPARQL.

The grammar-based interface provides a mechanism to formulate a query in any of the 15 languages, translate it to SPARQL and view the answers in any of those languages. The answers can be displayed as natural language descriptions or as triples. The latter can then be navigated as linked data. The browsing of the triples can be carried on continuously; by clicking on one of the triples listed in the answers, a new SPARQL query is launched and the results are generated as natural language text via the same grammar-based interface or as triples.

Fig. 2. Example of the query “who painted Guernica?” in 15 languages and in SPARQL.

Bul: кой нарисова Guernica	SPARQL: PREFIX painting: <http://spraakbanken.gu.se/rdf/owl/painting.owl#>
Cat: per qui és Guernica	PREFIX rdf:
Dan: hvem malede Guernica	<http://www.w3.org/1999/02/22-rdf-syntax-ns#>
Dut: wie schilderde Guernica	PREFIX rdfs:
Eng: who painted Guernica	<http://www.w3.org/2000/01/rdf-schema#>
Fin: kenen maalaama on Guernica	SELECT distinct ?author
Fre: par qui est Guernica	WHERE {
Ger: wer malte Guernica	?painting rdf:type painting:Painting .
Heb: מי צייר את גרניקה	?painting painting:createdBy ?painter . ?painter rdfs:label ?author .
Ita: da chi è dipinto Guernica	?painting rdfs:label ?title
Nor: hvem malte Guernica	FILTER (str(?title)= "Guernica") . }
Ron: cine a pictat Guernica	
Rus: кто нарисовал Guernica	
Spa: quién pintó Guernica	
Swe: vem målade Guernica	

3 Evaluation

Following previous question answering over linked data (QALD) evaluation challenges [5], we divided the evaluation into three parts, each focusing on a specific aspect: (1) user satisfaction, i.e. how many queries were answered; (2) correctness; and (3) coverage, how the system scales up.

For the first parts of the evaluation, we considered a number of random queries in 7 languages and counted the number of corrections that 1-2 native informants would make to the original queries. The results of the evaluation showed that the amount of suggested corrections is relatively low for the majority of the evaluated languages. The overall correctness of the generated queries seem to be representative and acceptable, at least among the users who participated in the evaluation.

Regarding coverage, the grammar allows for paraphrasing most of the question patterns, which sums up, on average to 3 paraphrases per construction in the English grammar. The number of alternatives varies across languages, but

the average across languages ranges between 2 and 3 paraphrases per construction. In addition, the 112 basic query patterns from the query grammar can be combined with logical operators, in order to obtain more complex queries, which sums up to 1159 query patterns that the grammar covers, including WH and Yes/No questions. The additions needed to build the query grammar in order for it to scale up are small, given that the other resources are in place. Also for building the query grammar for a given language, no more than 150 lines of code are needed. This process can be done semi-automatically.

4 Conclusions

We introduce a novel approach to multilingual interaction with the Semantic Web content via GF grammars. The method has been successfully demonstrated for the cultural heritage domain and could subsequently be implemented for other domains or scaled up in terms of languages or content coverage. The main contribution with respect to current state-of-the-art approaches is SPARQL support and question answering in 15 languages.

Acknowledgment

This work was supported by MOLTO European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement FP7-ICT-247914. The authors would like to acknowledge the Centre for Language Technology in Gothenburg.

References

1. Garlik, S.H., Andy, S.: SPARQL 1.1 Query Language. (March 2013) W3C Recommendation.
2. Ranta, A.: Grammatical Framework: Programming with Multilingual Grammars. CSLI Studies in Computational Linguistics. CSLI, Stanford (2011)
3. Ferré, S.: SQUALL: A controlled natural language for querying and updating RDF graphs. In: CNL. (2012) 11–25
4. Ngonga Ngomo, A.C., Bühmann, L., Unger, C., Lehmann, J., Gerber., D.: Sorry, I don't speak SPARQL — translating SPARQL queries into natural language. In: Proceedings of WWW. (2013)
5. Walter, S., Unger, C., Cimiano, P., Bär, D.: Evaluation of a Layered Approach to Question Answering over Linked Data. In: International Semantic Web Conference (2). (2012) 362–374
6. Lopez, V., Fernández, M., Motta, E., Stieler, N.: Poweraqua: Supporting users in querying and exploring the semantic web. *Semantic Web* **3**(3) (2012) 249–265
7. Damova, M., Dannélls, D.: Reason-able View of Linked Data for cultural heritage. In: Proceedings of the third International Conference on Software, Services and Semantic Technologies (S3T). (2011)

Extending Tagging Ontologies with Domain Specific Knowledge

Frederic Font¹, Sergio Oramas¹, György Fazekas², and Xavier Serra¹

¹Music Technology Group, Universitat Pompeu Fabra, Barcelona, Spain

²Centre for Digital Music, Queen Mary University of London, London, UK
{name.surname}@upf.edu, gyorgy.fazekas@eecs.qmul.ac.uk

Abstract. Currently proposed tagging ontologies are mostly focused on the definition of a common schema for representing the agents involved in a tagging process. In this paper we describe preliminary research around the idea of extending tagging ontologies by incorporating some domain specific class definitions and relations. We illustrate our idea with a particular use case where a tag recommendation system is driven by such an ontology. Besides our use case, we believe that such extended tagging ontologies can bring more meaningful structure into folksonomies and improve browsing and organisation functionalities of online platforms relying on tagging systems.

Keywords: Tagging ontology, Tag recommendation, Folksonomy, Freesound

1 Introduction

Tagging systems are extensively used in online sharing sites as a means of content browsing and organisation. In general, tagging systems allow users to annotate resources with free-form textual labels chosen by the users of the system. The resulting set of associations between tags, users and resources that arise in tagging systems is known as a folksonomy. Folksonomies suffer from a number of well-known issues including tag scarcity, ambiguities with synonymy and polysemy, typographical errors, the use of user-specific naming conventions, or even the use of different languages [1]. Despite these issues, folksonomies have succeeded in providing basic organisation and browsing functionalities to online sharing sites. However, their unstructured nature makes it difficult to allow more advanced capabilities such as hierarchical browsing or faceted searching.

In order to bring some structure to folksonomies, some studies have focused on the analysis of folksonomies to automatically derive structured or semi-structured representations of the knowledge of the domain, typically in the form of lightweight ontologies or hierarchical taxonomies [2–4]. However, these methods still tend to require significant amount of manual effort to provide meaningful representations. Some other studies have proposed modelling folksonomies and the tagging process using ontologies [5]. These ontologies are focused on defining a common schema for the agents involved in a tagging process. Current tagging ontologies may enhance interoperability between folksonomies, but do not generally provide ways of structuring a folksonomy with domain-specific knowledge.

In this paper, we present some preliminary research on extending a tagging ontology by including the possibility to represent the semantics of a specific domain. The generic idea is presented and discussed in Sec. 2. In Sec. 3 we describe a practical application in a real-world tagging system where the tagging ontology is used to drive a tag recommendation system. Finally, in Sec. 4, we discuss about possible future directions.

2 Extending a tagging ontology

Our starting point for the extension of the tagging ontology is the Modular Unified Tagging Ontology (MUTO) [5]. In the core of the MUTO ontology, the `muto:Tagging` class is defined which supports several relations to indicate, among others, a resource that is tagged (`muto:hasResource` of type `rdfs:Resource`), the tag assigned to the resource (`muto:hasTag` of type `muto:Tag`), and the user that made the tag assignment (`muto:hasCreator` of type `sioc:UserAccount`).

We propose to extend the tagging ontology in two ways. First, we add a number of subclasses to the `muto:Tag` class which can be used instead of `muto:Tag` (right side of Fig. 1). These subclasses represent different tag categories (i.e. with a narrower scope than the generic `muto:Tag` class), similarly to the idea of TagSet introduced in the SCOT ontology [6], but in a semantic sense. A tag category represents a broad concept that groups a set of tags that share some semantic characteristics related to the specific domain. The same principle is applied to resources, and a number of `rdfs:Resource` subclasses are defined (left side of Fig. 1). Resource subclasses (or resource categories) are used to organise resources into groups with a narrower scope than the general `rdfs:Resource` class. The particular definition of tag and resource categories would depend on the particular application domain of the extended tagging ontology (an example is given below). Also, in the diagram of Fig. 1, both tag and resource subclasses are only shown as a flat hierarchy, but more complex class structures could be explored. Moreover, existing domain ontologies and taxonomies may be reused to extend the tagging ontology.

Second, we propose to extend the tagging ontology by adding object properties to model semantic relations among tag categories and resource categories (dashed lines in Fig. 1). These object properties are useful to, for example, model dependencies between categories of tags and resources. The specific meaning of these semantic relations would also depend on the particular application domain of the extended tagging ontology. In addition to semantic relations between tag and resource categories, and given that the `muto:Tag` class inherits from the Simple Knowledge Organization System (SKOS) [7] class `skos:Concept`, semantic relations between tag individuals can be also modelled [5].

3 Use case: tag recommendation in Freesound

We applied an extended tagging ontology as described above in a tag recommendation task in the context of Freesound, an online collaborative sound database with more than 200,000 uploaded sounds and 3,8 million registered users [8]. In

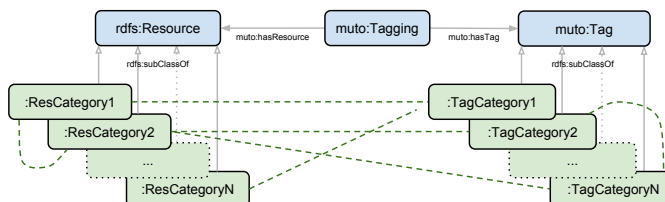


Fig. 1. Diagram of the extended parts of the tagging ontology.

previous work by the authors, a tag recommendation system was proposed which, given a set of input tags, is able to suggest other potentially relevant tags [9]. The system is based on the construction of five tag-tag similarity matrices tailored to five manually defined and rather generic audio categories (e.g. “Music”, “Effects”, etc.). The recommendation system uses a classifier to automatically predict one of these five categories depending on the input tags, and then uses the corresponding tag-tag similarity matrix for the recommendation process.

To improve that recommendation system, we used the extended tagging ontology to model the folksonomy and include some domain specific knowledge. On the one side, we extended the tagging ontology by adding 5 resource subclasses corresponding to the 5 sound categories mentioned above (e.g. `:EffectsSound`). Moreover, we defined 26 tag subclasses that are intended to group the tags in categories according to the type of information that they describe about sounds (i.e. grouped in audio properties). These include categories like “instrument”, “microphone”, “chord”, “material”, or “action” (e.g. `:InstrumentTag`). On the other side, we extended the ontology by defining several object properties that relate resource and tag categories. These object properties indicate that a particular tag category is relevant for one or more resource categories. For example, `:InstrumentTag` is relevant for `:MusicSound` audio category, and this is indicated with a `:hasInstrument` object property that relates instrument tag category with music resource category. Furthermore, we populated the extended ontology by manually classifying the 500 most used tags in Freesound into one of the 26 defined tag categories and added these tags as individuals (instances) of the corresponding tag category. This last step was necessary to bootstrap the tag recommendation system (see below).

Using this ontology we can extend the tag recommendation system in a way that, given the audio category detected by the classifier and the object properties that relate resource and tag categories, we can guide the annotation process by suggesting tag categories that are relevant for a particular sound. For example, for a sound belonging to the resource category `:MusicSound`, we can suggest tag categories like `:InstrumentTag` or `:TempoTag`, which are particularly relevant for musical sounds. Once tag categories are suggested, users can click on them and get a list of tag recommendations for every category. This list is obtained by computing the intersection of the tags provided by the aforementioned recommendation system (based on the tag-tag similarity matrix), with those that have been manually introduced in the ontology as tag instances of the selected tag category. See Fig. 2 for an screenshot of a prototype interface for that system.

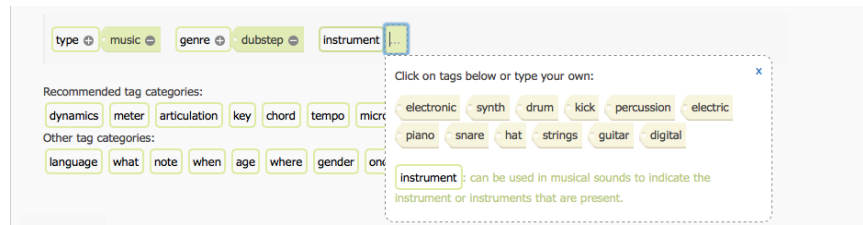


Fig. 2. Screenshot of the interface of a prototype tag recommendation system driven by the extended tagging ontology.

4 Conclusions

In this paper we have shown some preliminary research on extending current tagging ontologies with structured knowledge specific to the domain of application of a tagging system. By incorporating domain specific knowledge in tagging ontologies, we expect to be able to bring some semantically-meaningful structure into folksonomies. We have illustrated the idea with a use case in the context of an audio clip sharing site where a tag recommendation system is driven by an extended tagging ontology. Formal evaluation of the ontology-driven tag recommendation system is planned for future work. Besides the described use case, we think that using extended tagging ontologies can improve other aspects of online platforms relying on tagging systems such as browsing and organisation functionalities. The main limitation for such improvements is the population of the ontology. In our use case, we use a manually populated ontology to bootstrap the recommender, but the tagging system could further populate the ontology by learning new “tag individuals-tag category” relations when users annotate new sounds. Furthermore, other knowledge extraction techniques could be used to automatically populate the ontology with information coming from other user-generated data (e.g. in our case could be sound comments or textual descriptions), and even from external data sources from linked open data.

References

1. H. Halpin, V. Robu, and H. Shepard, “The dynamics and semantics of collaborative tagging,” in *Proceedings of the 1st Semantic Authoring and Annotation Workshop*, pp. 1–21, 2006.
2. P. Mika, “Ontologies are us: A unified model of social networks and semantics,” *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 5, pp. 5–15, Mar. 2007.
3. P. Heymann and H. Garcia-Molina, “Collaborative Creation of Communal Hierarchical Taxonomies in Social Tagging Systems,” tech. rep., 2006.
4. F. Limpens, F. L. Gandon, and M. Buffa, “Linking folksonomies and ontologies for supporting knowledge sharing: a state of the art,” 2009.
5. S. Lohmann, P. Díaz, and I. Aedo, “MUTO: the modular unified tagging ontology,” *Proceedings of the 7th International Conference on Semantic Systems - I-Semantics '11*, pp. 95–104, 2011.
6. H. L. Kim, S. Scerri, J. G. Breslin, S. Decker, and H. G. Kim, “The State of the Art in Tag Ontologies : A Semantic Model for Tagging and Folksonomies,” pp. 128–137, 2008.
7. SKOS: Simple Knowledge Organization System. <http://www.w3.org/TR/skos-reference>.
8. F. Font, G. Roma, and X. Serra, “Freesound Technical Demo,” in *Proceedings of the 21st ACM Conference on Multimedia (ACM MM 13)*, pp. 411–412, 2013.
9. F. Font, J. Serrà, and X. Serra, “Class-based tag recommendation and user-based evaluation in online audio clip sharing,” *Journal on Knowledge Based Systems*, 2014, 10.1016/j.knosys.2014.06.003.

Disambiguating Web Tables using Partial Data

Ziqi Zhang

Department of Computer Science, University of Sheffield, UK
z.zhang@dcs.shef.ac.uk

Abstract. This work addresses disambiguating Web tables - annotating content cells with named entities and table columns with semantic type information. Contrary to state-of-the-art that builds features based on the entire table content, this work uses a method that starts by annotating table columns using automatically selected *partial* data (i.e., a sample), then using the type information to guide content cell disambiguation. Different sample selection methods are introduced and tested to show that they contribute to higher accuracy in cell disambiguation, comparable accuracy in column type annotation with reduced computation.

1 Introduction

Enabling machines to effectively and efficiently access the increasing amount of tabular data on the Web remains a major challenge to the Semantic Web, as the classic indexing, search and NLP techniques fail to address the underlying semantics carried by tabular structures [1, 2]. This has sparked increasing interest in research on semantic Table Interpretation, which deals with semantically annotating tabular data such as shown in Figure 1. This work focuses specifically on annotating table columns that contain named entity mentions with semantic type information (column classification), and linking content cells in these columns with named entities from knowledge bases (cell disambiguation). Existing work follows a typical workflow involving 1) retrieving candidates (e.g., named entities, concepts) from the knowledge base, 2) constructing features of candidates, and 3) applying inference to choose the best candidates. One key limitation is that they adopt an *exhaustive* strategy to build the candidate space for inference. In particular, annotating table columns depends on candidate entities from *all* cells in the column [1, 2]. However, for human cognition this is unnecessary. For example, one does not need to read the entire table shown in Figure 1 - which may contain over a hundred rows - to label the three columns. Being able to make such inference using *partial* (as opposed to the entire table) or *sample* data can improve the efficiency of the task as the first two phases can cost up to 99% of computation time [1].

Sample driven Table Interpretation opens up several challenges. The first is defining a sample with respect to each task. The second is determining the optimal size of the sample with respect to varying sizes of tables. The third is choosing the optimal sample entries, since a skewed sample may damage accuracy. Our previous work in [5] has proposed TableMiner to address the first two challenges. This work adapts TableMiner to explore the third challenge. A number of sample selection techniques are introduced and experiments show that they can further improve cell disambiguation accuracy and in the column type annotation task, contribute to reduction in computation with comparable learning accuracy.

2 Related Work

An increasing number of work has been carried out in semantic Table Interpretation, such as Venetis et al. [3] that uses a maximum likelihood model, Limaye et al. [1] that uses a joint inference model, and Mulwad et al. [2] that uses joint inference with semantic message passing. These methods differ in terms of the inference models, features and background knowledge

bases used. All these methods are, as discussed earlier, ‘exhaustive’ as they require features built based on all content cells in order to annotate table columns. Zwicklbauer et al. [6] is the first method that annotates a table column using a sample of the column. However, the sample is arbitrarily chosen.

NE-column		
Name	Area	Prefecture
Trichonida	96,513	Aetolia-Acarnania
Yliki	22,731	Boeotia
Amvrakia	13,619	Aetolia-Acarnania
Lysimachia	13,200	Aetolia-Acarnania
...

Fig. 1. Lakes in Central Greece

3 Methodology

TableMiner is previously described in [5]. It disambiguates named entity columns in a table in two phases. The *first phase* creates preliminary annotations by using a sample of a column to classify the column in an iterative, incremental algorithm shown in Algorithm 1. In each iteration, a content cell $T_{i,j}$ drawn from a column T_j is disambiguated (output $E_{i,j}$). Then the concepts associated with the winning entity are gathered to create a set of candidate concepts for the column, C_j . Candidate concepts are scored and their score can change at each iteration due to newly disambiguated content cells adding re-enforcing evidence. At the end of each iteration, C_j from the current iteration is compared with the previous. If scores of candidate concepts are little changed (convergence, see [5] for a method for detection), then column classification is considered to be stable and the highest scoring candidates are (C_j^+) chosen to annotate the column. The *second phase* begins by disambiguating the remaining cells (part I), this time using the type information for the column to limit candidate entity space to those belonging to the type only. This may revise C_j for the column, either adding new elements, or resetting scores of existing ones and possibly causing the winning concept for the column to change. In this case, the next part of the second phase (part II) repeats the disambiguation and classification operations on the entire column, while using the new C_j^+ as constraints to restrict candidate entity space. This procedure repeats until C_j^+ and the winning entity in each cell stabilizes (i.e., no change).

Modified TableMiner For the purpose of this study, TableMiner is modified (TM_{mod}) to contain only the *first phase* and *part I* of the *second phase*. In other words, we do not revise the column classification results obtained from sample data. Therefore TM_{mod} may only use a fraction of a column’s data to classify the column, which reduces computation overhead compared to classic ‘exhaustive’ methods.

Sample selection The choice of the sample can affect learning in TM_{mod} in two ways. While the *size* of the sample is dealt with by the convergence measure described in [5], here we address the issue of selecting the *suitable* sample entries to ensure learning accuracy. Since column classification depends on the disambiguated cells in the sample, we hypothesize that high accuracy of cell disambiguation contributes to high accuracy

in column classification. And we further hypothesize that higher accuracy of content cell disambiguation can be achieved by 1) richer feature representation, and 2) less ambiguous names (i.e., if a name is used by only one or very few named entities). Therefore, we propose three methods to compute a score of each content cell in a column, then rank them by the score before running Algorithm 1 (i.e., input T_j will contain content cells the order of which is re-arranged based on the scores).

One-sense-per-discourse (*ospd*) First and foremost, we make the hypothesis of ‘one-sense-per-discourse’ in table context, that if an NE-column is not the subject column of the table (e.g., the first column in Figure 1 is a subject column), then cells containing the same text content are extremely likely to express the same meaning¹. Thus to apply *ospd* we firstly re-arrange cells in a column by putting those containing duplicate text content adjacent to each other. Next, when disambiguating a content cell, the feature representation of the cell concatenates the row context of the cell, and that of any adjacent cells with the same text content (e.g., in Table 1 we assume ‘Aetolia-Acarnania’ on the three rows to have the same meaning, and build a single feature representation by concatenating all the three rows). Effectively this creates a richer feature representation for cells whose content re-occur across a table.

Feature size (*fs*) With *fs*, we firstly apply *ospd*, then rank cells in a column by the size of their feature representation as determined by the number of tokens in a bag-of-words representation. This would allow TM_{mod} to start with cells that potentially have the largest - hence ‘richest’ - feature representation in Algorithm 1.

Name length (*nl*) With *nl*, we count the number of words in the cell text content to be disambiguated and rank cells by this number - name length (e.g., in Table 1 ‘Aetolia-Acarnania’ has two words and will be disambiguated before ‘Boeotia’). *nl* merely relies on the name length of a cell content and does not apply *ospd*. The idea is that longer names are less likely to be ambiguous.

4 Evaluation and Conclusion

We evaluate the proposed methods of sample selection using two datasets: LimayeAll and Limaye200². LimayeAll contains over 6000 tables and is used for evaluating content cell disambiguation. Limaye200 contains a subset 200 tables from LimayeAll with

¹ Due to space limitation, details are omitted but can be found in [3, 4]

² [4], currently under transparent review.

Algorithm 1 Sample based classification

```

1: Input:  $T_j$ ;  $C_j \leftarrow \emptyset$ 
2: for all cell  $T_{i,j}$  in  $T_j$  do
3:    $prevC_j \leftarrow C_j$ 
4:    $E_{i,j} \leftarrow \text{disambiguate}(T_{i,j})$ 
5:    $C_j \leftarrow \text{updateclass}(C_j, E_{i,j})$ 
6:   if  $\text{convergence}(C_j, prevC_j)$  then
7:     break
8:   end if
9: end for

```

Cell disambiguation (LimayeAll)				Column classification (Limaye200)			
TM_{mod}	TM_{mod}^{ospd}	TM_{mod}^{fs}	TM_{mod}^{nl}	TM_{mod}	TM_{mod}^{ospd}	TM_{mod}^{fs}	TM_{mod}^{nl}
0.809	0.812	0.812	0.813	0.723	0.719	0.721	0.723

Table 1. Cell disambiguation and column classification accuracy in F1.

columns manually annotated with Freebase concepts, and used for evaluating column classification. As a **baseline**, TM_{mod} without any sample selection techniques is used. It simply chooses cells from a column in their original order in Algorithm 1. This is compared against TM_{mod}^{ospd} , which applies *ospd* to non-subject NE-columns, preserves the original order but disambiguates groups of cells containing the same text content; TM_{mod}^{fs} that applies *ospd* to non-subject NE-columns then prioritizes cells that potentially have richer feature representation; and TM_{mod}^{nl} that prioritizes cells containing longer text content. Results on both datasets are shown in Table 1. It suggests that, compared against TM_{mod} , the sample selection techniques can enhance the accuracy of cell disambiguation marginally. In the column classification task however, they do not add benefits in terms of accuracy. By analyzing the computation overhead in terms of the automatically determined sample size in each table, it shows that the sample selection techniques have reducing the amount of data to be processed in column classification. As an example, TM_{mod} converges on average after processing 58% of cells in a table column, i.e., it manages to classify a table column using a sample size of 58% of the total number of cells in that column. TM_{mod}^{ospd} reduces this to 53%, for TM_{mod}^{fs} 52% and for TM_{mod}^{nl} 58% (unchanged). This may contribute to noticeable reduction in CPU time since the construction of feature space (including querying knowledge bases) for each data unit consumes over 90% of computation time [1]. To summarize, it has been shown that, by using sample selection techniques, it is possible to semantically annotate Web tables in a more efficient way, achieving comparable or even higher learning accuracy depending on tasks.

Acknowledgement: Part of this work is carried out in the LODIE project (Linked Open Data Information Extraction), funded by EPSRC (EP/J019488/1).

References

1. Limaye, G., Sarawagi, S., Chakrabarti, S.: Annotating and searching web tables using entities, types and relationships. Proceedings of the VLDB Endowment 3(1-2), 1338–1347 (2010)
2. Mulwad, V., Finin, T., Joshi, A.: Semantic message passing for generating linked data from tables. In: International Semantic Web Conference (1). pp. 363–378. Springer (2013)
3. Venetis, P., Halevy, A., Madhavan, J., Paşca, M., Shen, W., Wu, F., Miao, G., Wu, C.: Recovering semantics of tables on the web. Proc. of VLDB Endowment 4(9), 528–538 (Jun 2011)
4. Zhang, Z.: Start small, build complete: Effective and efficient semantic table interpretation using tableminer. In: The Semantic Web Journal (under reviewer, #668-1878) (2014)
5. Zhang, Z.: Towards efficient and effective semantic table interpretation. In: To appear in: ISWC2014 (2014)
6. Zwicklbauer, S., Einsiedler, C., Granitzer, M., Seifert, C.: Towards disambiguating web tables. In: International Semantic Web Conference (Posters & Demos). pp. 205–208 (2013)

On Linking Heterogeneous Dataset Collections

Mayank Kejriwal and Daniel P. Miranker

University of Texas at Austin
{kejriwal,miranker}@cs.utexas.edu

Abstract. Link discovery is the problem of linking entities between two or more datasets, based on some (possibly unknown) specification. A blocking scheme is a one-to-many mapping from entities to blocks. Blocking methods avoid $O(n^2)$ comparisons by clustering entities into blocks, and limiting the evaluation of link specifications to entity pairs within blocks. Current link-discovery blocking methods explicitly assume that two RDF datasets are provided as input, and need to be linked. In this paper, we assume instead that two heterogeneous dataset collections, comprising arbitrary numbers of RDF and tabular datasets, are provided as input. We show that data model heterogeneity can be addressed by representing RDF datasets as *property tables*. We also propose an unsupervised technique called *dataset mapping* that maps datasets from one collection to the other, and is shown to be compatible with existing clustering methods. Dataset mapping is empirically evaluated on three real-world test collections ranging over government and constitutional domains, and shown to improve two established baselines.

Keywords: Heterogeneous Blocking, Instance Matching, Link Discovery

With the advent of Linked Data, discovering links between entities emerged as an active research area [2]. Given a link specification, a naive approach would discover links by conducting $O(n^2)$ comparisons on the set of n entities. In the *Entity Resolution* (ER) community, a preprocessing technique called *blocking* mitigates full pairwise comparisons by clustering entities into blocks. Only entities within blocks are paired and compared. ER is critical in data integration systems [1]. In the Semantic Web, the problem has received attention as scalably discovering *owl:sameAs* links between RDF datasets [5].

In the Big Data era, *scalability* and *heterogeneity* are essential components of systems and hence, practical requirements for real-world link discovery. Scalability is addressed by blocking, but current work assumes that the *dataset pairs* between which entities are to be linked are provided. In other words, datasets A and B are input to the pipeline, and entities in A need to be linked to entities in B . Investigations in some important real-world domains show that pairs of dataset *collections* also need to undergo linking. Each collection is a *set* of datasets. An example is government data. Recent government efforts have led to release of public data as batches of files, both across related domains and time, as

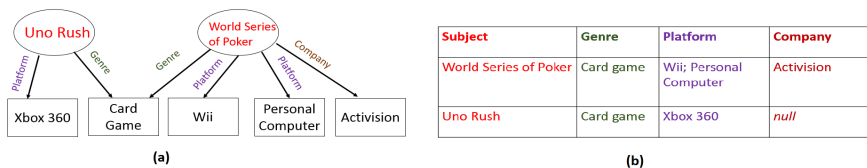


Fig. 1. The property table representation. For subjects that don’t have a property, the reserved keyword *null* is entered. ; is a reserved delimiter that allows each field value to have set semantics.

one of our real-world test sets demonstrates. Thus, there are (at least) two scalability issues: at the collection level, and at the dataset level. That is, datasets in one collection first need to be mapped to datasets in the second collection, after which a blocking scheme is learned and applied on each mapped pair. The problem of blocking two collections is exacerbated by *data model heterogeneity*, where some datasets are RDF and the others are tabular.

We note that data model heterogeneity has larger implications, since it also applies in the standard case where two datasets are provided, but one is RDF and the other, tabular. In recent years, the growth of both Linked Open Data and the Deep Web have been extensively documented. Datasets in the former are in RDF, while datasets in the latter are typically relational. Because of data model heterogeneity, both communities have adopted different techniques for performing link discovery (typically called *record linkage* in the relational community). There is a clear motivation, therefore, in addressing this particular type of heterogeneity, since it would enable significant cross-fertilization between both communities. We will show an example of this empirically.

The intuition behind our proposed solution to data model heterogeneity is to represent the RDF dataset as an *information-preserving table*, not as a set of triples or a directed graph. The literature shows that such a table has previously been proposed as a *physical* data structure, for efficient implementation of triple stores [6]. An example of this table, called a *property table*, is shown in Figure 1. We note that this is the first application of property tables as logical data structures in the link-discovery context. The table is information-preserving because the original set of triples can be reconstructed from the table.

Note that the property table builds a *schema* (in the form of a set of properties) for the RDF file, regardless of whether it has accompanying RDFS or OWL metadata. Thus, it applies to arbitrary files on Linked Open Data. Secondly, numerous techniques in relational data integration can handle datasets with different schemas (called *structural heterogeneity*). By representing RDF datasets in the input collections as property tables, data model heterogeneity is reduced to structural heterogeneity in the tabular domain.

Figure 2 shows the overall framework of link-discovery. The first step, proposed in this paper for collections, is called the *dataset mapping* step. It takes two collections A and B of heterogeneous datasets as input and produces a set

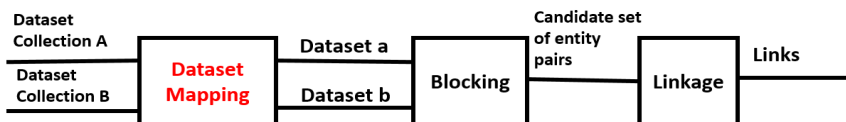


Fig. 2. The overall link-discovery framework. Dataset mapping is our contribution.

of mappings between datasets. Let such a mapping be (a, b) where $a \in A, b \in B$. For *each* such mapping, the subsequent blocking process is invoked. Blocking has been extensively researched, with even the least expensive blocking methods having complexity $O(n)$, where n is the total number of entities in the input datasets. Blocking generates a *candidate* set of entity pairs, Γ , with $|\Gamma| \ll O(n^2)$. Thus, blocking provides complexity improvements over brute-force linkage. To understand the savings of dataset mapping, assume that each collection contains q datasets, and each dataset contains n entities. Without dataset mapping, any blocking method would be at least $O(qn)$. With mapping, there would be q instances of complexity $O(n)$ each. Since Γ depends heavily on n , the savings carry over to the final quadratic process (but which cannot be quantified without assumptions about the blocking process). We empirically demonstrate these gains. An added benefit is that there is now scope for parallelization.

The mapping process itself relies on *document similarity* measures developed in the information retrieval community, by representing each dataset as bag of tokens. Intuitively, mapped datasets should have relatively high document similarity to each other. Empirically, we found a tailored version of cosine similarity to work best. Many packages exist for efficiently computing it. Computing similarities between all pairs of datasets, we get a $|A| \times |B|$ matrix. A straightforward approach would use a threshold to output many-many mappings, or a graph bipartite matcher to output one-one mappings. The former requires a parameter specification, while the latter is cubic ($O(q^3)$). Therefore, we opted for a *dominating strategy*, which can be computed in the same time it takes to build the matrix. Namely, a mapping (a, b) is chosen if the score in the cell of (a, b) *dominates*, that is, it is the highest in its constituent row and column. This has the advantage of being conservative against false positives. The method applies even when $|A| \neq |B|$. In our experiments, we used cosine document similarity combined with the dominating strategy.

Experiments: Some results are demonstrated in Figure 3. We use three real-world test cases. The first two test cases (a and b in the figure) comprise RDF dataset collections describing court cases decided in Colombia and Venezuela respectively, along with Constitution articles. The third test set consists of ten US government budget dataset collections from 2009 to 2013¹. Other such collections can also be observed on the same website, providing motivation for dataset mapping. We have released publicly available datasets on a single page² with

¹ <http://www.pewstates.org/research/reports/>

² <https://sites.google.com/a/utexas.edu/mayank-kejriwal/datasets>

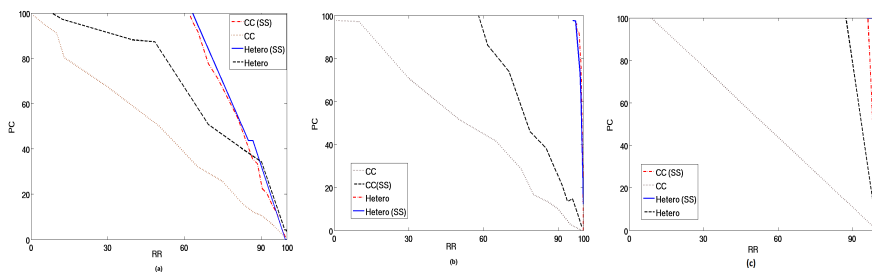


Fig. 3. *RR* (Reduction Ratio) quantifies *efficiency* and is given by $1 - |\Gamma|/|\Omega|$, with Ω the set of all $O(n^2)$ entity pairs, while *PC* (Pairs Completeness) measures *recall* of Γ with respect to the ground-truth set, $\Omega_m (\subseteq \Omega)$. *SS* indicates if dataset mapping (equivalently denoted *Source Selection*) was used.

ground-truths. We used two popular methods as baselines, a state-of-the-art unsupervised clustering method called *Canopy Clustering* (*CC* in figure) [4] as well as an extended feature-selection based blocking method (*Hetero* in figure) [3]. The gains produced by dataset mapping are particularly large on *CC*. More importantly, we found that the dataset mapping algorithm was able to deduce the correct mappings without introducing false positives or negatives, and with run-time negligible compared to the subsequent blocking procedures.

Future Work: We continue to investigate dataset mapping, including other document similarity measures, task domains and mapping strategies. We are also investigating supervised versions of the problem, particularly in cases where token overlap is low. Finally, we are investigating the property table further.

References

1. P. Christen. *Data matching: concepts and techniques for record linkage, entity resolution, and duplicate detection*. Springer, 2012.
2. R. Isele and C. Bizer. Learning expressive linkage rules using genetic programming. *Proceedings of the VLDB Endowment*, 5(11):1638–1649, 2012.
3. M. Kejriwal and D. P. Miranker. An unsupervised algorithm for learning blocking schemes. In *Data Mining, 2013. ICDM'13. Thirteenth International Conference on*. IEEE, 2013.
4. A. McCallum, K. Nigam, and L. H. Ungar. Efficient clustering of high-dimensional data sets with application to reference matching. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 169–178. ACM, 2000.
5. F. Scharffe, Y. Liu, and C. Zhou. Rdf-ai: an architecture for rdf datasets matching, fusion and interlink. In *Proc. IJCAI 2009 workshop on Identity, reference, and knowledge representation (IR-KR), Pasadena (CA US)*, 2009.
6. K. Wilkinson, C. Sayers, H. A. Kuno, D. Reynolds, et al. Efficient rdf storage and retrieval in jena2. In *SWDB*, volume 3, pages 131–150, 2003.

Scientific data as RDF with Arrays: Tight integration of SciSPARQL queries into MATLAB

Andrej Andrejev¹, Xueming He¹, Tore Risch¹

¹ Uppsala DataBase Laboratory (UDBL), Department of Information Technology,
Uppsala University, Box 337, SE-751 05 Uppsala, Sweden.
{Andrej.Andrejev, Tore.Risch}@it.uu.se, emilyhexueming@hotmail.com

Abstract. We present an integrated solution for storing and querying scientific data and metadata, using MATLAB environment as client front-end and our prototype DBMS on the server. We use RDF for experiment metadata, and numeric arrays for the rest. Our extension of SPARQL supports array operations and extensibility with foreign functions.

1 Introduction

In many branches of science and engineering, researchers accumulate large amounts of experimental data [3,4] and use widely recognized (de-facto standard) libraries of algorithms to analyze and refine that data. Tools such as MATLAB or similar serve as integrated environments that provide basic file management, extensibility with algorithmic libraries, visualization and debugging tools, and are generally oriented towards single-user scenario.

What is typically missing is the infrastructure for storing the descriptions of experiments, including parameters, terminology mappings, provenance records and other kinds of metadata. At best, this information is stored in a set of variables in the same files that contain large numeric arrays of experimental data, and thus is prone to duplication and hard to update. We have addressed this problem in our previous work [2] utilizing the Semantic Web approach for storing both data and metadata, and using Scientific SPARQL query language [1], that extends SPARQL queries with numeric array operations and external user-defined functions. The goal of SciSPARQL is to provide uniform query access to both metadata about the experiments and the massive experimental data itself, as illustrated by table 1. Section 3 gives more detailed account of SciSPARQL features.

SciSPARQL is supported by our software prototype - SSDM (Scientific SPARQL Database Manager [1,2]), a database management system (DBMS) for storing and querying data originating from scientific experiments. SSDM provides scalable storage representation of RDF and numeric multidimensional arrays.

Table 1. Comparison of data processing domains of MATLAB, SPARQL and SciSPARQL

	MATLAB	SPARQL	SciSPARQL
Metadata		✓	✓
Scientific data including Arrays	✓		✓

In this work, we demonstrate a client-server architecture featuring (i) **SSDM server**: the centralized storage for both experiment metadata (as RDF) and arrays

stored in binary files linked from the RDF dataset and (ii) **MSL**: a MATLAB extension that allows to establish connections to SSDM server, run SPARQL queries and updates directly from MATLAB interpreter, and access the query result sets.

We show that the data is shipped from the server only on demand. Also, the conversion of numeric array data between native MATLAB format and internal SSDM representation only takes place if non-MATLAB function going to access the array, or, more typically, a certain range within the array.

For this demo¹ we have deployed SSDM server on a Linux machine to store RDF datasets in-memory and array data in binary .mat files [5], which is currently a de-facto standard. (This provides the same speed for reading and processing array data as it would be while using MATLAB alone) The demo script is run on the client machine inside MATLAB interpreter.

2 MATLAB-SciSPARQL Link

The extension to MATLAB includes two main classes: *Connection* and *Scan*, and additional classes used to represent RDF types on MATLAB client side, e.g. URIs and typed literals. An additional class *MatProxy* is used to represent (on the client side) an array stored in a .mat file on the server.

Connection encapsulates a connection to SSDM server, including methods for

- executing SciSPARQL queries and obtaining a result as a *Scan*,
- executing non-query SciSPARQL statements, e.g. updates and function definitions, apart from inserting RDF triples into the dataset on the server
- defining URI prefixes to be used both on client and server side,
- shipping MATLAB arrays from client to the server,
- managing data persistence on the server.

Scan encapsulates a result set of the query. The data is not physically retrieved, stored or shipped anywhere before it is explicitly accessed as a row in the scan. *Scan* includes methods for iterating through the result sets of SciSPARQL queries: the arrays and scalar numbers become represented by MATLAB arrays and numbers, other RDF values get represented by the wrapper objects defined in MSL.

As we show in the demo, the user can easily create MATLAB routines to convert (partially or entirely) the data from the *Scan* into the desired representation, e.g. for visualization.

3 Scientific SPARQL

We have extended SPARQL language to query and update RDF datasets extended with arrays. SciSPARQL [1] includes

- extensions for declaratively specifying element access, slicing, projection and transposition operations over numeric arrays of arbitrary dimensionality,
- a library of array aggregation functions, that are performed on the server in order to reduce the amount of data shipped to the client,
- extensibility with user-defined foreign functions, allowing to make use of existing computational libraries.

¹ The demo script is available at <http://www.it.uu.se/research/group/udbl/SciSPARQL/demo3/>

SciSPARQL is designed to handle both metadata (stored or viewed as RDF) and large numeric data to be accessed in the uniform way: by the same query, from the same dataset.

One important feature of SciSPARQL is ability to define *SciSPARQL functional views*, essentially, the named parameterized queries (or, similarly, updates). These can be used in other queries, or called directly from MATLAB client with parameters provided as MATLAB values. The conversion of values from MATLAB to RDF is performed automatically on the client.

4 SSDM Server and Array Proxy Objects

Scientific SPARQL Database Manager is designed for storing RDF data and numeric multidimensional arrays, working either as in-memory DBMS, or with a help of SQL-based [2], or any other interfaced back-end storage. In this demo SSDM server is configured to store RDF triples in-memory, and array data as managed directory of native .mat files. Reading and writing .MAT files on the server side is done via freely distributed MATLAB MCR libraries.

To save a snapshot of RDF dataset linking to the arrays stored in .mat files, `save()` SciSPARQL directive can be sent via the connection. The server can be restarted with a named image, and continue to function as in-memory DBMS.

The main purpose of SSDM server is to process SciSPARQL queries and updates. As part of an update, a `store()` function can be called from the client. A MATLAB value (e.g. numeric multidimensional array) will be shipped to the server as a binary .mat file, and saved under server-managed name in the server file system. The *Array Proxy* object pointing to the value in that .mat file will be returned to the client, and used as a replacement for the actual array e.g. as a parameter to SciSPARQL queries and updates. Once stored in RDF dataset, *Array Proxy* serves as a link from metadata RDF graph to the numeric data stored externally in a .mat file.

If the file is already on the server, and its location is known (maybe, due to some convention among the users), an alternative `link()` function can be used to obtain an equivalent *Array Proxy* object.

When SciSPARQL query involves slicing, element access, projection or array aggregate operations on an array represented by *Array Proxy*, the SSDM server reads the specified part of the array stored in file into SSDM internal array representation (thus performing slicing, projection or element access), does any further processing (e.g. applying array aggregate functions, like "sum of all columns"), and ships the resulting, typically, much smaller array to MATLAB client, where it is converted back to MATLAB representation. It is also possible to do slicing and projection operations within the native .mat array representation, when no further processing by SSDM is planned.

One of the possible workflows involving arrays is shown on Fig. 1-2. First, a MATLAB array *A* is created on the client. A call to `store()` function ships it to the server and returns an *Array Proxy* object. This object is used in RDF triples sent to SSDM while populating RDF graph describing the experiment.

At the query phase (Fig. 2), a subset of *A* (that is now stored on the server in a .mat file) is selected, fed to `array_sum()` aggregate function, and the result (a single number) is shipped back to the client for post-processing and visualization.

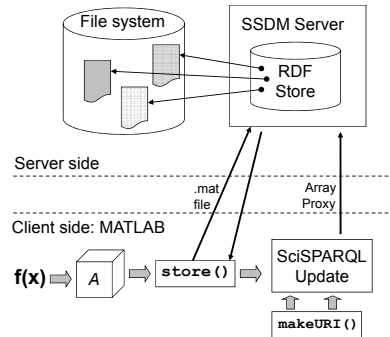


Fig. 1. Storing client-generated data and metadata on SSDM server.

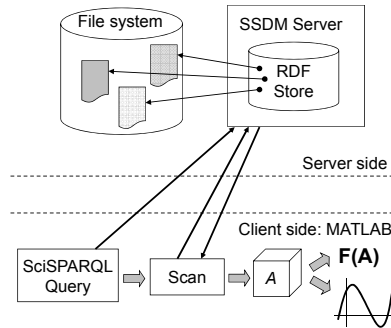


Fig. 2. Querying data and metadata on SSDM server from MATLAB client.

5. Conclusion

The use of standard query languages for bringing the remotely stored data into the computational environments is becoming increasingly popular as the data gets bigger and more distributed. MATLAB already has facility to execute SQL, and R statistical environment recently gained a simple SPARQL package [6]. We take the next step, by providing extensions to the standard query techniques, to make the database connections even more useful and efficient.

The approach with linking to the data instead of copying and storing it locally is beneficial, as the creation of the RDF graph to represent metadata takes negligibly small time compared to copying the massive data described by this RDF graph. There is a number of efficient binary storage formats around, and our approach can be easily extended to any of them, as long as it is possible to address stored data in terms of string or symbolic identifiers, and read specified parts of the arrays.

The main benefit, however, is integrating Semantic Web metadata management approach (RDF and SPARQL) into an environment that misses it so obviously. The MATLAB users can now take advantage of remote and centralized repositories for both massive numeric data and metadata, send queries that combine them both, retrieve exactly as much data as required for the task, and do any further processing the way they already do.

References

1. A.Andrejev and T.Risch. Scientific SPARQL: Semantic web queries over scientific data. In International Workshop on Data Engineering Meets the Semantic Web, ICDE'12
2. A.Andrejev, S.Toor, A.Hellander, S.Holmgren, and T.Risch: Scientific Analysis by Queries in Extended SPARQL over a Scalable e-Science Data Store, In e-Science'13
3. M. Stonebraker, J. Becla, D. J. DeWitt, K.-T. Lim, D. Maier, O. Ratzesberger, and S. B. Zdonik. Requirements for science data bases and scidb. In CIDR '09.
4. E. Soroush, M. Balazinska, and D. L. Wang. Arraystore: a storage manager for complex parallel array processing. In SIGMOD '11
5. http://www.mathworks.se/help/pdf_doc/matlab/matfile_format.pdf
6. <http://cran.r-project.org/web/packages/SPARQL/index.html>

Measuring Similarity in Ontologies: A new family of measures

Tahani Alsubait, Bijan Parsia, and Uli Sattler

School of Computer Science, The University of Manchester, United Kingdom
{alsubait,bparsia,sattler}@cs.man.ac.uk

1 Introduction

Similarity measurement is important for numerous applications. Be it classical information retrieval, clustering, ontology matching or various other applications. It is also known that similarity measurement is difficult. This can be easily seen by looking at the several attempts that have been made to develop similarity measures, see for example [2, 4]. The problem is also well-founded in psychology and a number of psychological models of similarity have been already developed, see for example [3]. Rather than adopting a psychological model for similarity as a foundation, we noticed that some existing similarity measures for ontologies are ad-hoc and unprincipled. In addition, there is still a need for similarity measures which are applicable to expressive Description Logics (DLs) (i.e., beyond \mathcal{EL}) and which are terminological (i.e., do not require an *ABox*). To address these requirements, we have developed a new family of similarity measures which are founded on the feature-based psychological model [3]. The individual measures vary in their accuracy/computational cost based on which *features* they consider.

To date, there has been no thorough empirical investigation of similarity measures. This has motivated us to carry out two separate empirical studies. First, we compare the new measures along with some existing measures against a gold-standard. Second, we examine the practicality of using the new measures over an independently motivated corpus of ontologies (BioPortal library) which contains over 300 ontologies. We also examine whether cheap measures can be an approximation of some more computationally expensive measures. In addition, we explore what could possibly could wrong when using a cheap similarity measure.

2 A new family of similarity measures

The new measures are based on Jaccard’s similarity coefficient which has been proved to be a proper metric (i.e., satisfies the properties: equivalence closure, symmetry and triangle inequality). Jaccard’s coefficient, which maps similarity to a value in the range [0,1], is defined as follows (for sets of “features” A', B' of A, B , i.e., subsumers of A and B):

$$J(A, B) = \frac{|(A' \cap B')|}{|(A' \cup B')|}$$

We aim at similarity measures for general OWL ontologies and thus a naive implementation of this approach would be trivialised because a concept has infinitely many subsumers. To overcome this, we present refinements for the similarity function in which we do not count all subsumers but consider subsumers from a set of (possibly complex) concepts of a concept language \mathcal{L} . Let C and D be concepts, let \mathcal{O} be an ontology and let \mathcal{L} be a concept language. We set:

$$\begin{aligned} S(C, \mathcal{O}, \mathcal{L}) &= \{D \in \mathcal{L}(\tilde{\mathcal{O}}) \mid \mathcal{O} \models C \sqsubseteq D\} \\ \text{Com}(C, D, \mathcal{O}, \mathcal{L}) &= S(C, \mathcal{O}, \mathcal{L}) \cap S(D, \mathcal{O}, \mathcal{L}) \\ \text{Union}(C, D, \mathcal{O}, \mathcal{L}) &= S(C, \mathcal{O}, \mathcal{L}) \cup S(D, \mathcal{O}, \mathcal{L}) \\ \text{Sim}(C, D, \mathcal{O}, \mathcal{L}) &= \frac{|\text{Com}(C, D, \mathcal{O}, \mathcal{L})|}{|\text{Union}(C, D, \mathcal{O}, \mathcal{L})|} \end{aligned}$$

To design a new measure, it remains to specify the set \mathcal{L} . For example:

$$\begin{aligned} \text{AtomicSim}(C, D) &= \text{Sim}(C, D, \mathcal{O}, \mathcal{L}_{\text{Atomic}}(\tilde{\mathcal{O}})), \text{ and } \mathcal{L}_{\text{Atomic}}(\tilde{\mathcal{O}}) = \tilde{\mathcal{O}} \cap N_C. \\ \text{SubSim}(C, D) &= \text{Sim}(C, D, \mathcal{O}, \mathcal{L}_{\text{Sub}}(\tilde{\mathcal{O}})), \text{ and } \mathcal{L}_{\text{Sub}}(\tilde{\mathcal{O}}) = \text{Sub}(\mathcal{O}). \\ \text{GrSim}(C, D) &= \text{Sim}(C, D, \mathcal{O}, \mathcal{L}_G(\tilde{\mathcal{O}})), \text{ and } \mathcal{L}_G(\tilde{\mathcal{O}}) = \{E \mid E \in \text{Sub}(\mathcal{O}) \\ &\text{ or } E = \exists r.F, \text{ for some } r \in \tilde{\mathcal{O}} \cap N_R \text{ and } F \in \text{Sub}(\mathcal{O})\}. \end{aligned}$$

where $\tilde{\mathcal{O}}$ is the signature of \mathcal{O} , N_C is the set of concept names and $\text{Sub}(\mathcal{O})$ is the set of concept expressions in \mathcal{O} . The rationale of $\text{SubSim}(\cdot)$ is that it provides similarity measurements that are sensitive to the modeller's focus. To capture more possible subsumers, one can use $\text{GrSim}(\cdot)$ for which the grammar can be extended easily.

3 Approximations of similarity measures

Some measures might be practically inefficient due to the large number of candidate subsumers. For this reason, it would be nice if we can examine whether a “cheap” measure can be a good approximation for a more expensive one.

Definition 1 *Given two similarity functions $\text{Sim}(\cdot)$, $\text{Sim}'(\cdot)$, we say that:*

- $\text{Sim}'(\cdot)$ preserves the order of $\text{Sim}(\cdot)$ if $\forall A_1, B_1, A_2, B_2 \in \tilde{\mathcal{O}}: \text{Sim}(A_1, B_1) \leq \text{Sim}(A_2, B_2) \implies \text{Sim}'(A_1, B_1) \leq \text{Sim}'(A_2, B_2)$.
- $\text{Sim}'(\cdot)$ approximates $\text{Sim}(\cdot)$ from above if $\forall A, B \in \tilde{\mathcal{O}}: \text{Sim}(A, B) \leq \text{Sim}'(A, B)$.
- $\text{Sim}'(\cdot)$ approximates $\text{Sim}(\cdot)$ from below if $\forall A, B \in \tilde{\mathcal{O}}: \text{Sim}(A, B) \geq \text{Sim}'(A, B)$.

Consider $\text{AtomicSim}(\cdot)$ and $\text{SubSim}(\cdot)$. The first thing to notice is that the set of candidate subsumers for the first measure is actually a subset of the set of candidate subsumers for the second measure ($\tilde{\mathcal{O}} \cap N_C \subseteq \text{Sub}(\mathcal{O})$). However, we need to notice also that the number of entailed subsumers in the two cases need not to be proportionally related. Hence, the above examples of similarity measures are, theoretically, non-approximations of each other.

4 Empirical evaluation

We carry out a comparison between the three measures $GrSim(\cdot)$, $SubSim(\cdot)$ and $AtomicSim(\cdot)$ against human similarity judgments. We also include two existing similarity measures in this comparison (Rada [2] and Wu & Palmer [4]). We also study in detail the behaviour of our new family of measures in practice. $GrSim(\cdot)$ is considered as the expensive and most precise measure in this study.

To study the relation between the different measures *in practice*, we examine the following properties: order-preservation, approximation from above/below and correlation (using Pearson’s coefficient).

4.1 Experimental set-up

Part 1: Comparison against a gold-standard The similarity of 19 SNOMED-CT concept pairs was calculated using the three methods along with Rada [2] and Wu & Palmer [4] measures. We compare these similarities to human judgments taken from the Pedersen et al.[1] test set.

Part 2: Cheap vs. expensive measures A snapshot of BioPortal from November 2012 was used as a corpus. It contains a total of 293 ontologies. We excluded 86 ontologies which have only atomic subsumptions as for such ontologies the behaviour of the considered measures will be identical, i.e., we already know that $AtomicSim(\cdot)$ is good and cheap. Due to the large number of classes and difficulty of spotting interesting patterns by eye, we calculated the pairwise similarity for a sample of concepts from the corpus. The size of the sample is 1,843 concepts with 99% confidence level. To ensure that the sample encompasses concepts with different characteristics, we picked 14 concepts from each ontology. The selection was not purely random. Instead, we picked 2 random concepts and for each random concept we picked some neighbour concepts.

4.2 Results

How good is the expensive measure? Not surprisingly, $GrSim$ and $SubSim$ had the highest correlation values with experts’ similarity (Pearson’s correlation coefficient $r = 0.87, p < 0.001$). Secondly comes $AtomicSim$ with $r = 0.86$. Finally comes Wu & Palmer then Rada with $r = 0.81$ and $r = 0.64$ respectively. Figure 1 shows the similarity curves for the 6 measures used in this comparison. The new measures along with Wu & Palmer measure preserve the order of human similarity more often than Rada measure. They mostly underestimated similarity whereas the Rada measure was mostly overestimating human similarity.

Cost of the expensive measure The average time per ontology taken to calculate grammar-based similarities was 2.3 minutes (standard deviation $\sigma = 10.6$ minutes, median $m = 0.9$ seconds) and the maximum time was 93 minutes for the Neglected Tropical Disease Ontology which is a *SRIQ* ontology with 1237 logical axioms, 252 concepts and 99 object properties. For this ontology, the cost of $AtomicSim(\cdot)$ was only 15.545 sec and 15.549 sec for $SubSim(\cdot)$. 9 out of 196 ontologies took over 1 hour to be processed. One thing to note about these ontologies is the high number of logical axioms and object properties. Clearly, $GrSim(\cdot)$ is far more costly than the other two measures. This is why we want to know how good/bad a cheaper measure can be.

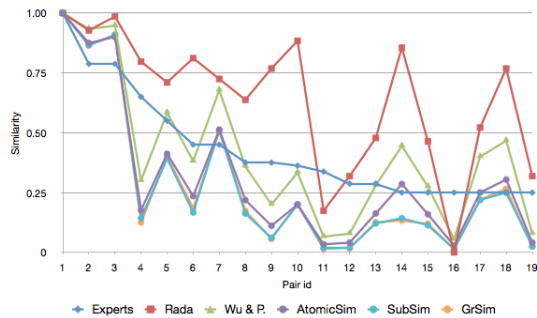


Fig. 1: 6 Curves of similarity for 19 SNOMED clinical terms

How good is a cheap measure? Although we have excluded all ontologies with only atomic subsumptions from the study, in 12% of the ontologies the three measures were perfectly correlated ($r = 1, p < 0.001$). These perfect correlations indicate that, in some cases, the benefit of using an expensive measure is totally neglectable.

AtomicSim(·) and *SubSim*(·) did not preserve the order of *GrSim*(·) in 80% and 73% of the ontologies respectively. Also, they were not approximations from above nor from below in 72% and 64% of the ontologies respectively.

Take a look at the African Traditional Medicine ontology in Figure 2. *SubSim*(·) is 100% order-preserving while *AtomicSim*(·) is only 99% order-preserving.

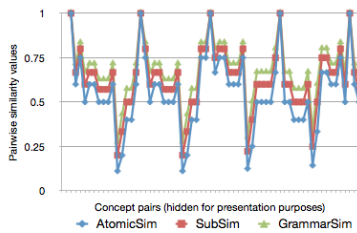


Fig. 2: African Traditional Medicine

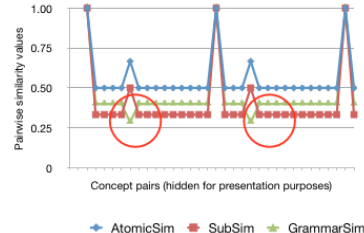


Fig. 3: Platynereis Stage

Note also the Platynereis Stage Ontology in Figure 3 in which both *AtomicSim*(·) and *SubSim*(·) are 75% order-preserving. However, *AtomicSim*(·) was 100% approximating from above while *SubSim*(·) was 85% approximating from below.

References

1. T. Pedersen, S. Pakhomov, S. Patwardhan, and C. Chute. Measures of semantic similarity and relatedness in the biomedical domain. *Journal of Biomedical Informatics*, 30(3):288–299, 2007.
2. R. Rada, H. Mili, E. Bicknell, and M. Blettner. Development and application of a metric on semantic nets. In *IEEE Transaction on Systems, Man, and Cybernetics*, volume 19, page 1730, 1989.
3. A. Tversky. Features of similarity. *Psychological Review by the American Psychological Association, Inc.*, 84(4), July 1977.
4. Z. Wu and MS. Palmer. Verb semantics and lexical selection. In *Proceedings of the 32nd. Annual Meeting of the Association for Computational Linguistics (ACL 1994)*, page 133138, 1994.

Towards Combining Machine Learning with Attribute Exploration for Ontology Refinement

Jedrzej Potoniec¹, Sebastian Rudolph², and Agnieszka Ławrynowicz¹

¹ Institute of Computing Science, Poznan University of Technology, Poland
{jpotoniec, alawrynowicz}@cs.put.poznan.pl

² Technische Universität Dresden, Germany sebastian.rudolph@tu-dresden.de

Abstract. We propose a new method for knowledge acquisition and ontology refinement for the Semantic Web utilizing Linked Data available through remote SPARQL endpoints. This method is based on combination of the attribute exploration algorithm from formal concept analysis and the active learning approach from machine learning.

1 Introduction

Knowledge acquisition is a process of capturing knowledge, typically from a human expert, and thus it concerns all systems and environments where that kind of knowledge is required. It is also said to be major bottleneck in development of intelligent systems due to its difficulty and time requirements. Of course the Semantic Web, as an area concerned with structured and precise representation of information, has to deal with exactly the same issue.

Since the early days of the Semantic Web, building ontologies has been a difficult and laborious task. Frequently people trying to express complex knowledge do not know how to perform this task properly. Mistakes come from difficulty in understanding the complex logic formalism supporting OWL.

Frequently an ontology engineer would start collecting vocabulary and requirements for an ontology, structuralize the vocabulary and later specify more complex dependencies [6]. We propose a solution to support knowledge acquisition for ontology construction. Especially we address the last part of the process, where some basic knowledge is already gathered and more complex dependencies are to be specified. We aim to answer the question *how to extend an ontology with meaningful, valid and non-trivial axioms taking into consideration available data and user workload?*

2 Related work

For knowledge acquisition for ontology development many approaches have been proposed so far. The most basic ones are ontology editors supporting ontology development, such as *Protégé*³. In addition to that, there are methodologies helpful in ontologies development, such as the one proposed in *NeOn* [6].

³ <http://protege.stanford.edu/>

In [1,5] applications of attribute exploration algorithm from formal concept analysis to ontology development have been proposed. [1] describes how to discover subsumptions between conjunction of classes and [5] extends it to properties' domains and ranges.

In [3] the idea of learning ontologies purely from Linked Data by means of discovering association rules is presented. [2] presents a methodology for manually building and populating domain ontologies from Linked Data.

3 Approach

The proposed approach is to support the user during attribute exploration by means of machine learning (ML). The ML algorithm's task is to answer simple, non-interesting questions posed by the attribute exploration algorithm and leave for the user only these questions which are non-trivial to answer.

Input of our proposed algorithm is an ontology \mathcal{O} , a partial context derived from it, and two thresholds θ_a and θ_r . They are, respectively, thresholds for accepting and rejecting an implication and have to be manually chosen w.r.t. the used ML algorithm. Result of the algorithm is a set of probably valid implications, which can be transformed into subsumptions for extending the ontology. A detailed description of the algorithm is presented below. For sake of clarity its description treats the attribute exploration algorithm as a black box, which provides the next implication to consider.

1. Generate implication $L \rightarrow R$ by means of the attribute exploration algorithm.
2. For every $r \in R$, do the following sequence of steps:
 - (a) If $L \rightarrow \{r\}$ is already refuted by some of the known individuals, go to the next r .
 - (b) If $\mathcal{O} \models \prod L \sqsubseteq r$, remember implication $L \rightarrow \{r\}$ as a valid one and go to the next r .
 - (c) Compute probabilities of acceptance p_a and rejection p_r of the implication $L \rightarrow \{r\}$ with the ML algorithm. Note that $p_a + p_r = 1$.
 - (d) If $p_a \geq \theta_a$, remember the implication $L \rightarrow \{r\}$ as a valid one and go to the next r .
 - (e) If $p_r \geq \theta_r$, go to the step 2i.
 - (f) Ask user if implication $L \rightarrow \{r\}$ is valid.
 - (g) Add considered implication with user's answer to a set of learning examples for the ML algorithm.
 - (h) If the implication is valid, remember it as a valid one and go to the next r .
 - (i) Otherwise, extend the partial context with a counterexample either provided by user or auto-generated.

The purpose of iteration through the set of conclusions R in the algorithm is twofold. We believe that this way user can more easily decide if the presented

implication is valid or not, because she does not have to consider complex relation between two conjunctions of attributes.

The other thing is that this way automated generation of counterexamples provides more concrete results. For an arbitrary implication $L \rightarrow R$ a counterexample can be generated and said to have all attributes from L and to not have at least one attribute from R . This is not in line with the method of partial context induction, as it is unclear which exactly attribute from R the counterexample does not have. Because of that partial context can not reflect knowledge base accurately anymore, and the attribute exploration algorithm can start to generate invalid implications. If the implication has a single attribute in its right-hand side, it is clear which attribute the counterexample does not have.

3.1 Application of machine learning

The task which ML algorithm is to solve can be seen as a kind of active learning with a binary classification. Every implication is classified as *valid* or *invalid* and if the algorithm is unsure, the user is asked.

One should note that not every classifier generates reasonable probabilities. For example, rule-based or tree-based systems usually are not suitable for that purpose. Problem of generating probabilities can be also seen as a regression problem.

Moreover costs of both types of mistakes are different and distribution of learning examples can be heavily imbalanced, i.e. implications with one decision may appear much often than with other decision. To reflect these fact a classifier suitable for cost-sensitive learning is required.

To apply machine learning techniques, a way to transform implications to feature vectors is required. We apply three approaches to this problem. First of all, a single purely syntactic measure is used: the number of attributes in the left-hand side divided by the number of all attributes. Secondly, there are features made of values of measures typical for association rules mining. Their computation is based on features of individuals in the ontology. Following the naming convention from [4], we use *coverage*, *prevalence*, *support*, *recall* and *lift*.

Finally, we use a mapping from the set of the attributes to Linked Data in order to obtain the number of objects in an RDF repository supporting an implication or its parts. Every attribute is mapped to a SPARQL graph pattern with a single featured variable denoting the object identifier. Following the same name convention from [4], *coverage*, *prevalence*, *support*, *recall* and *confidence* are used. All of these features can be computed using only SPARQL COUNT DISTINCT expressions and basic graph patterns and thus they maintain relatively low complexity and are suitable to use with remote SPARQL endpoints.

Such a feature vector is later labeled with the classifier mentioned above and given answer (valid/invalid/unsure) is used to either refine the ontology or ask the user. If the user is asked, her answer is then used as a correct label for the feature vector and the classifier is relearned.

4 Conclusions and future work

As we are proposing a method which is to make development and refinement of domain-specific ontologies easier, our main goal for evaluation is to validate its practical usability. We plan to apply our method to a selection of domain-specific ontologies concerning some knowledge of general type such as literature, music and movies. We plan to use a crowdsourcing service to validate our hypotheses. We hope that with ontologies with a theme being general enough and additional information available in the Internet, the crowd will be able to validate our decisions about implications and Linked Data mappings.

We believe that our approach is promising and will be able to help ontology engineers in the process of ontology refinement. We are combining three technologies very suitable for this kind of a task. First of all, the attribute exploration algorithm that has been developed especially for discovering additional relations between attributes. Moreover, Linked Data is supposed to describe parts of the world. Obviously, this description can not be assumed to be neither accurate nor complete, yet it should be sufficient to support the user in a process of ontology refinement. Finally, the whole purpose of machine learning algorithms is to adapt themselves, and thus they are suitable to replace the user in uniform, repeatable tasks.

Acknowledgement. Jędrzej Potoniec and Agnieszka Ławrynowicz acknowledge support from the PARENT-BRIDGE program of Foundation for Polish Science, cofinanced from European Union, Regional Development Fund (Grant No POMOST/2013-7/8 *LeoLOD – Learning and Evolving Ontologies from Linked Open Data*).

References

1. Baader, F., Ganter, B., et al.: Completing description logic knowledge bases using formal concept analysis. In: Proc. of IJCAI 2007. pp. 230–235. AAAI Press (2007)
2. Dastgheib, S., Mesbah, A., Kochut, K.: mOntage: Building Domain Ontologies from Linked Open Data. In: IEEE Seventh International Conference on Semantic Computing (ICSC). pp. 70–77. IEEE (2013)
3. Fleischhacker, D., Völker, J.: Inductive learning of disjointness axioms. In: Meersman, R., Dillon, T., et al. (eds.) On the Move to Meaningful Internet Systems: OTM 2011, LNCS, vol. 7045, pp. 680–697. Springer Berlin Heidelberg (2011)
4. Le Bras, Y., Lenca, P., Lallich, S.: Optimonotone measures for optimal rule discovery. *Computational Intelligence* 28(4), 475–504 (2012)
5. Rudolph, S.: Acquiring generalized domain-range restrictions. In: Medina, R., Obiedkov, S. (eds.) Formal Concept Analysis, LNCS, vol. 4933, pp. 32–45. Springer Berlin Heidelberg (2008)
6. Suárez-Figueroa, M.C., Gómez-Pérez, A., Fernández-López, M.: The NeOn Methodology for Ontology Engineering. In: Suárez-Figueroa, M.C., Gómez-Pérez, A., et al. (eds.) Ontology Engineering in a Networked World, pp. 9–34. Springer Berlin Heidelberg (2012)

ASSG: Adaptive structural summary for RDF graph data

Haiwei Zhang, Yuanyuan Duan, Xiaojie Yuan, and Ying Zhang*

Department of Computer Science and Information Security, Nankai University.
94, Weijin Road, Tianjin, China
{zhanghaiwei, duanyuanyuan, yuanxiaojie, zhangying}
@dbis.nankai.edu.cn
<http://dbis.nankai.edu.cn>

Abstract. RDF is considered to be an important data model for Semantic Web as a labeled directed graph. Querying in massive RDF graph data is known to be hard. In order to reduce the data size, we present ASSG, an **A**daptive **S**tructural **S**ummary for **R**DF **G**raph data by bisimulations between nodes. ASSG compresses only the part of the graph related to queries. Thus ASSG contains less nodes and edges than existing work. More importantly, ASSG has the adaptive ability to adjust its structure according to the updating query graphs. Experimental results show that ASSG can reduce graph data with the ratio 85% in average, higher than that of existing work.

Keywords: Adaptive structural summary, RDF graph, Equivalence class

1 Introduction

The resource description framework (RDF) data model has been designed as a flexible representation of schema-relaxable or even schema-free information for the Semantic Web [1]. RDF can be modeled by a labeled directed graph and querying in RDF data is usually thought to be a process of subgraph matching. The subgraph matching problem is defined as follows: for a data graph G and a query graph Q , retrieve all subgraphs of G that are isomorphic to Q . Existing two solutions, subgraph isomorphism and graph simulation, are expensive where subgraph isomorphism is NP-complete and graph simulation takes quadratic time. Further, indices are used to accelerate subgraph queries on large graph data, but indices incur extra cost on construction and maintenance (see [2] for a survey). Motivated by this, a new approach, using *graph compression*, has been proposed recently [3]. In [3], Fan et al. proposed query preserving graph compression G_r , which compresses massive graph into a small one by partitioning nodes into equivalence classes. For subgraph matching, G_r can reduce graph data with the ratio 57% in average. However, for a designated query graph, lots of components (nodes and edges) in G_r are redundant. Hence it is possible to construct a compressed graph for designed subgraph matching.

* Corresponding author.

In this paper, we present ASSG (**A**daptive **S**tructural **S**ummary of **G**raphs), a graph compression method that further reduces the size of the graph data. ASSG has less components than G_r and more importantly, it has adaptive ability to adjust its structure according to different subgraph matchings. In the following sections, we mainly introduce our novel technique.

2 Adaptive Structural Summary

In this section, we present our approach of adaptive structural summary for labeled directed graph data (such as RDF). ASSG is actually an compressed graph constructed by equivalence classes of nodes and it has adaptive ability to adjust its structure according to different query graphs.

Graph data is divided into different equivalence classes by bisimulation relations as [3] proposed. For computing bisimulation relation, we refer to the notion *rank* proposed in [4] for describing structural feature from leaf nodes (if exist). A.Dovier, et al.[4] proposed function of computing ranks of nodes for both directed acyclic graph (DAG) and directed cyclic graph (DCG). *Rank* is something like structural feature of nodes from leaf nodes in graph data.

An equivalence class EC_G of nodes in graph data $G = (V, E, L)$ is denoted by a triple (V_e, R_e, L_e) , where (1) V_e is a set of nodes included in the equivalence class, (2) R_e is the *rank* of the nodes, and (3) L_e denotes the labels of the nodes.

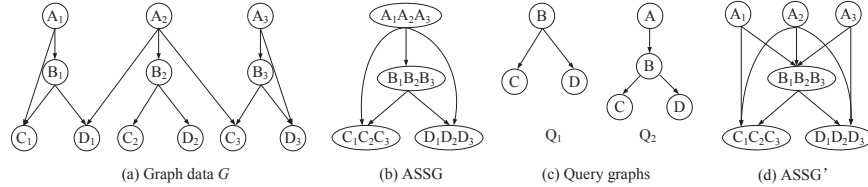


Fig. 1. Graph data and equivalence classes

Fig. 1 shows examples of a graph (Fig. 1(a)) and the equivalence classes of nodes (Fig. 1(b)). Labels and ranks of nodes in the same equivalence class are the same, such as $rank(C_1) = rank(C_2) = rank(C_3) = 0$, $rank(A_1) = rank(A_2) = rank(A_3) = 2$, and so on. Two times of DFS processing will be performed to construct equivalence classes of node. DCG will be changed into DAG by algorithm of Tarjan in the first DFS (not shown in Fig. 1). Subsequently, in the second DFS, rank of each node will be measured and then the node will be collapsed into corresponding equivalence class by its label and rank. Hence, V of G is partitioned to different equivalence classes with a cost of $O(|V| + |E|)$.

For a labeled directed graph $G = (V, E, L)$, We define ASSG as $G_{ASS} = (V_{ASS}, E_{ASS}, L_{ASS}, R_{ASS})$, where: (1) V_{ASS} denotes the set of nodes that collapsed by the nodes in the same equivalence class, (2) E_{ASS} is the set of edges, (3) L_{ASS} is the labels of nodes in V_{ASS} , (4) R_{ASS} records the *rank* of each $v \in V_{ASS}$.

Obviously, ASSG is the minimum pattern that can describe labeled directed graph data because nodes with the same label and rank will be collapsed. Unfortunately, the process of measuring ranks will lose some descendants or ancestors of nodes. And this case will not conform to the definition of bisimulation, and thus bring out wrong answers for subgraph matching. For example, in Fig. 1(b), the nodes A_1 and A_2 in the same equivalence class have different children. To solve the problem, ASSG will adaptively adjust its structure for updating query graphs.

For each subgraph matching, the procedure of adaptively updating ASSG includes two stages: matching and partitioning. Given a query graph $Q = (V_Q, E_Q, L_Q)$ and ASSG $G_{ASS} = (V_{ASS}, E_{ASS}, L_{ASS}, R_{ASS})$, assuming that $R_Q = \{rank(v_Q) | v_Q \in V_Q\}$. For the matching stage, $\forall v \in V_Q$ and $u \in V_Q$, $\exists v', u' \in V_{ASS}$, if $L_Q(v) = L_{ASS}(v')$, $L_Q(u) = L_{ASS}(u')$, and $R_Q(v) - R_Q(u) = R_{ASS}(v') - R_{ASS}(u')$, then v, u matches v', u' respectively. For the partitioning stage, nodes in ASSG matching current query graph will be partitioned into different parts according to its neighbors by the algorithm presented in [5] with the complexity of time $O(|E| \log |V_Q|)$. In Fig. 1(c), ASSG will not change while matching Q_1 , but ASSG will change to the structure shown in Fig. 1(d) while matching Q_2 . It is obvious that the size of ASSG will increase after further partition, but each partition will adjust minimum amount of nodes. While subgraph matching focuses on frequent nodes, ASSG will remain stable.

3 Experimental Evaluation

In this section, we performed experiments on both realistic and synthetic data sets to verify the performance of ASSG.

Table 1. Compress Ratio of G_r and ASSG

Data Set	$ G < V , E , L >$	G_r	ASSG(15%)
California	60K < 24K, 32K, 95 >	49.22%	33.25%
Internet	530K < 96K, 421K, 50 >	42.41%	17.08%
Citation	1.7M < 815K, 806K, 67 >	31.71%	5.83%
Synthetic	2.6M < 1.4M, 2.1M, 60 >	26.9%	3.73%

Firstly, we use compression ratio as a measurement for evaluating the effectiveness of ASSG for subgraph matchings compared with G_r . We define compression ratio of ASSG as: $C_{ASS} = |V_{ASS}|/|V|$. Similarly, the compression ratio of G_r is $C_{G_r} = |V_r|/|V|$. The ration is lower, the better. The effectiveness of ASSG compared with G_r is reported in Table 1 where $|G|$ denotes to the size of graph data. For a query graph $G_q = (V_q, E_q, L_q)$, the compression ratio of ASSG is decided by the number of labels $|L_q|$ in the query graph. Assuming that $|L_q| = 15\% \times |L|$, then we can study from table 1: By ASSG, graph data can be highly compressed according to query graphs. ASSG reduces graph data by 85% in average. The compression ratio of ASSG is lower than that of G_r .

Secondly, we evaluate the efficiency of updating ASSG. Assuming that number of labels in query graph is 15% of $|L|$. We generate two query graphs for

updating ASSG. The number of repeated labels in these two graphs are 0, 1, 2, 5 respectively as table 2 shows. We can study that the more repeated labels in different query graphs, the less time occupation for ASSG to update. As a result, for frequent subgraph matchings, ASSG can be updated and maintained with low cost of time.

Table 2. Time Occupations of Updating ASSG (s)

Data Set	0 repeated label	1 repeated label	2 repeated labels	5 repeated labels
California	8.95	2.96	2.79	2.73
Internet	28.64	25.42	21.29	9.9
Citation	55.49	53.7	47.1	6.35
Synthetic	113.47	101.32	91.24	33.73

4 Conclusion and Future work

We have proposed ASSG, adaptive structural summary for RDF graph data. ASSG is based on equivalence classes of nodes, and ASSG compresses graph data according to the query graphs. We presented main idea for constructing and updating ASSG and designed experiments on realistic and synthetic data sets to evaluate the effectiveness and efficiency of our technique. Experimental results show that the compression ratio of ASSG is lower than that of existing work G_r , and ASSG is efficiently updated for frequent queries. Further more, we will use ASSG for optimizing SPARQL queries on RDF data for semantic web.

Acknowledgments. This work is supported by National Natural Science Foundation of China under Grant No. 61170184, 61402243, the National 863 Project of China under Grant No. 2013AA013204, National Key Technology R&D Program under Grant No.2013BAH01B05, and the Tianjin Municipal Science and Technology Commission under Grant No.13ZCZDZX02200, 13ZCZDZX01098 and 13JCQNJC00100.

References

1. T.Neumann., G.Weikum.: The rdf-3x engine for scalable management of rdf data. VLDB J., 19(1), 91–113, 2010.
2. Z.Sun., H.Wang., H.Wang., B.Shao., J.Li.: Efficient Subgraph matching on billion node graphs. The VLDB Journal, 5(9), 788–799 (2012)
3. W.Fan., J.Li., X.Wang., Y.Wu.: Query preserving graph compression. In: ACM SIGMOD International Conference on Management of Data, pp. 157–168. ACM, New York (2012)
4. A.Dovier., C.Piazza., A.Policriti.: A fast bisimulation algorithm. In: Conference on Computer Aided Verification, pp. 79–90. Springer-Verlag Berlin Heidelberg (2001)
5. R.Paige., R.E.Tarjan., R.Bonic.: A linear time solution to the single function coarsest partition problem. Theoretical Computer Science, 40(1), 67–84 (1985)

Evaluation of String Normalisation Modules for String-based Biomedical Vocabularies Alignment with AnAGram

Anique van Berne,
A.vanBerne@Elsevier.com
Elsevier BV

Veronique Malaisé
V.Malaise@Elsevier.com
Elsevier BV

Abstract: Biomedical vocabularies have specific characteristics that make their lexical alignment challenging. We have built a string-based vocabulary alignment tool, AnAGram, dedicated to efficiently compare terms in the biomedical domain, and evaluate this tool's results against an algorithm based on Jaro-Winkler's edit-distance. AnAGram is modular, enabling us to evaluate the precision and recall of different normalization procedures. Globally, our normalization and replacement strategy improves the F-measure score from the edit-distance experiment by more than 100%. Most of this increase can be explained by targeted transformations of the strings with the use of a dictionary of adjective/noun correspondences yielding useful results. However, we found that the classic Porter stemming algorithm needs to be adapted to the biomedical domain to give good quality results in this area.

1. Introduction

Elsevier has a number of online tools in the biomedical domain. Improving their interoperability involves aligning the vocabularies these tools are built on. The vocabulary alignment tool needs to be generic enough to work with any of our vocabularies, but each alignment requires specific conditions to be optimal, due to vocabularies' specific lexical idiosyncrasies.

We have designed a modular, step-wise alignment tool: AnAGram. Its normalization procedures are based on previous research[1], basic Information Retrieval normalization processes, and our own observations. We chose a string-based alignment method as these perform well on the anatomical datasets of the OAEI campaign[1], and string-based alignment is an important step in most methods identified in [3][4].

We compare the precision and recall of AnAGram against an implementation of Jaro-Winkler's edit-distance method (JW)[7] and evaluate the precision of each step of the alignment process. We gain over 100% F-measure compared to the edit-distance method. We evaluate the contribution and quality of the string normalization modules independently and show that the Porter stemmer[2] does not give optimal results in the biomedical domain.

In Section 2 we present our use-case: aligning Dorland's to Elsevier's Merged Medical Taxonomy (EMMeT)¹. Section 3 describes related work in vocabulary

¹ <http://river-valley.tv/elsevier-merged-medical-taxonomy-emet-from-smart-content-to-smart-collection/>

alignment in the biomedical domain. Section 4 and 5 present AnAGram and evaluate against Jaro-Winkler's edit-distance. Section 6 presents future work and conclusions.

2. Use case: Dorland's definition alignment to EMMeT

Elsevier's Merged Medical Taxonomy (EMMeT) is used in "Smart Content" applications²; it contains more than 1 million biomedical concepts and their hierarchical, linguistic and semantic relationships. We aim at expanding EMMeT with definitions from the authoritative biomedical dictionary Dorland's³ by aligning them.

3. Related work

Cheatham and Hitzler[1] list the types of linguistic processes used by at least one alignment tool in the Ontology Alignment Evaluation Initiative (OAEI)[5]. AnAGram implements all syntactic linguistic transformations listed; instead of a generic synonym expansion system, we used a correspondence dictionary of adjective/noun pairs. This dictionary is a manually curated list based on information automatically extracted from Dorland's. It contains pairs that would not be solved by stemming such as *saturnine/lead*. Ambiguous entries, such as *gluteal/natal*, were removed.

Chua and Kim's[6] approach for string-based vocabulary alignment is the closest to AnAGram: they use WordNet⁴, a lexical knowledge base, to gather adjective/noun pairs to improve the coverage of their matches, after using string normalization steps; our set of pairs is larger than the one derived from WordNet.

4. AnAGram: biomedical vocabularies alignment tool

AnAGram was built for use on a local system⁵, and is tuned to performance by using hash-table lookup to find matches. Currently, no partial matching is possible. The matching steps are built in a modular way: one can select the set of desired steps. The source taxonomy is processed using these steps and the target taxonomy is processed sequentially: the alignment stops at the first match. Modules are ordered to increasing distance between original and transformed string, simulating a confidence value.

Exact matching: corresponds to JW edit-distance 1.

Normalization: special characters are removed or transformed (*Sjögren's syndrome* to *Sjogren's syndrome*; punctuation marks to space), string is lower cased.

Stop word removal: tokenization by splitting on spaces, removal of stop words, using a list that was fine-tuned over several rounds of indexing with EMMeT.

² http://info.clinicalkey.com/docs/Smart_Content.pdf

³ <http://www.dorlands.com/>

⁴ <http://wordnet.princeton.edu/>

⁵ Dell™ Precision™ T7500, 2x Intel® Xeon® CPU E5620 2.4 GHz processors, 64 GB RAM. Software: Windows 7 Professional 64 bit, Service Pack 1; Perl v5.16.3

Re-ordering: tokens are sorted alphabetically, enabling matches for inverted terms.
Substitution: sequences of tokens are replaced with the corresponding value from our dictionary, applying a longest string matching principle.
Stemming: using the Porter stemming algorithm[2] (Perl module Lingua::Stem::Snowball). The substitution step is then repeated, using stemmed dictionary entries.
Independent lists: stop-words list and substitution dictionary are independent files.

5. Experimentation and results

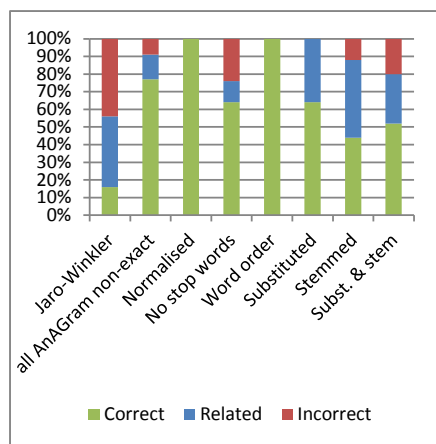
We align EMMeT version 3.2 (13/12/13) (1,027,717 preferred labels) to Dorland’s 32nd edition (115,248 entries). We evaluate AnAGram as a whole against JW, with a 0.92 threshold (established experimentally). The JW implementation can work only with preferred labels.

To evaluate the recall of AnAGram vs the JW implementation, we use a manual gold set of 115 mappings created by domain experts (Table 1). AnAGram gives better recall and better precision than the JW method.

	Correct mapping	Incorrect mapping	Recall (%)	Precision(%)	F-measure
Jaro-Winkler	46	8	43%	85%	0.57
AnAGram	80	3	71%	96%	0.82

Table 1 - Results of AnAGram vs. Jaro-Winkler on Dorland’s Gold Set pairs

We evaluate a random sample of 25 non-exact alignments from each module to get a better insight on AnAGram’s normalization process. The results are either: *Correct*, *Related* (useful but not exactly correct), or *Incorrect* (Table 2 and Figure 1). AnAGram gives more correct results but JW is useful for finding related matches.



Preferred labels	C	R	I
Jaro-Winkler	16	40	44
AnAGram non-exact	77	14	9
Normalised	25	0	0
No stop words	16	3	6
Word order	25	0	0
Substituted	16	9	0
Stemmed	11	11	3
Subst. & stem	13	7	5

Table 2 – Results for AnAGram’s modules. (C: correct; R: related; I: incorrect)

Figure 1 - Quality of matches returned by AnAGram’s modules.

We evaluate the performance of each normalization step by evaluating 25 random results for each of AnAGram’s modules separately⁶ (Table 2, Figure 1). Normal-

⁶ Some modules are based on the result of a previous transformation, so the later the module comes in the chain, the more complicated matches it faces.

ization does very well (100% correct results). Removal of stop words causes some errors and related matches: single-letter stop words can be meaningful, like *A* for *hepatitis A*. Word order rearranging ranks second: it does not often change the meaning of the term. Substitution performs reasonably well; most of the non-correct results are related matches. Stemming gives the poorest results with false positives due to nouns/verbs stemmed to the same root, such as *cilitated/ciliate*. The substituted and stemmed matches have a result similar to the stemmed results. Still, even the worst results from any AnAGram module are better than the overall results of the non-exact matches from the JW algorithm. One reason for this is that the JW does not stop the alignment at the best match, but delivers everything that satisfies the threshold of 0.92.

Not all modules account for an equal portion of the non-exact results. The normalization module delivers around 70% of matches, stemming accounts for 15 to 20% and the other modules account for 2% to 4% of the matches each.

6. Future work and conclusion

Results are good compared to OAEI large biomedical vocabularies alignment's results for string-based tools[1]. We will work on the Stemming algorithm, the improvement of our stop words list and substitution dictionary, and on adding an optimized version of the JW algorithm as a final optional module for AnAGram to improve results further. In this way we will benefit from additional related matches in cases where no previous match was found.

References

- [1] Michelle Cheatham, Pascal Hitzler. *String Similarity Metrics for Ontology Alignment*. International Semantic Web Conference (ISWC2013) (2) 2013: 294-309
- [2] Cornelis .J. van Rijsbergen, Stephen E. Robertson, MartinF. Porter. *New models in probabilistic information retrieval*. London: British Library. (British Library Research and Development Report, no. 5587), 1980
- [3] Jérôme Euzenat (Coordinator) et al. *State of the art on Ontology alignment*. Knowledge Web D 2.2.3, 2004.
- [4] Jérôme Euzenat, Pavel Shvaiko. *Ontology Matching*. Springer-Verlag, Berlin Heidelberg 2013
- [5] Jérôme Euzenat, Christian Meilicke, Heiner Stuckenschmidt, Pavel Shvaiko, Cássia Trojahn. *Ontology Alignment Evaluation Initiative: Six Years of Experience*. Journal on Data Semantics XV, Lecture Notes in Computer Science (6720) 2011: 158-192
- [6] Watson W. K. Chua and Jung-Jae Kim. *BOAT: Automatic alignment of biomedical ontologies using term informativeness and candidate selection*. Journal of Biomedical Informatics (45) 2012: 337-349
- [7] William E.Winkler. *String Comparator Metrics and Enhanced Decision Rules in the Fellegi-Sunter Model of Record Linkage*. Proceedings of the Section on Survey Research Methods (American Statistical Association) 1990: 354–359

Keyword-Based Semantic Search Engine Koios++

Björn Forcher¹, Andreas Giloj¹, and Erich Weichselgartner¹

Leibniz Institute for Psychology Information,
Germany, email: firstname.lastname@zpid.de

Abstract. In this paper, we describe the keyword-based semantic search engine *KOIOS++* which interprets the keywords and computes a set of SPARQL queries. The special feature of *KOIOS++* is that it leverages not only the class hierarchy but also the property hierarchy. The algorithm and data structures of *KOIOS++* are based on a well-established approach that we extended by minor adjustments of the data structures and a sophisticated weighting strategy.

Key words: keyword-based semantic search, RDFS semantics

1 Introduction

The RDF data model is a popular data model of the Semantic Web which can be easily transferred to a simple directed graph. RDFS extends RDF and provides representational constructs for describing ontologies, for instance the properties *rdfs:subClassOf* and *rdfs:subPropertyOf*. Both properties are transitive relations which are used to describe class or property hierarchies of ontologies.

Finding information in graph-shaped data, keyword search seems to be natural because it is the de facto standard for current search engines. The field of keyword-based search on graph-structured data in general, and in particular over RDF data, is a prevalent research topic and corresponding efforts can be referred in [1], [2], and [3] at different glances. Tran *et al.* [4] describe an interesting approach of keyword-based semantic search for computing the top-k ranked search results from RFD(S) graphs. They first compute queries from the keywords, allowing the users to choose one of them, and finally to process the query using the underlying database engine. Nevertheless, these approaches do not focus on the hierarchy of properties and thus, they cannot make use of the transitive tree of the *rdfs:subPropertyOf* relation. Recent publications of that group focused on the processing of the approximated top-k ranked results [5] to reduce computation time but they did not consider further aspects of the RDFS semantics. The primary motivation of our work is to make the *rdfs:subPropertyOf* available for keyword-based semantic search on graph shaped data. We extend the approach of Tran *et al.* by using a special graph mapping and a sophisticated weighting strategy which is implemented in the *KOIOS++* search engine. We claim that in this way we get a semantic benefit because we can favor certain graph patterns during the search. As a result, it is possible, for instance, to make use of the property hierarchy by leveraging the *rdfs:subPropertyOf* relation.

The paper is structured as follows. Sec. 2 introduces the search engine *KOIOS++* including its search algorithm and data structures. We conclude with a brief summary and a small outlook (Sec. 3).

2 Koios++

As mentioned in the previous chapter, the approach of *KOIOS++* is based on the work of Tran *et al.*[4] which is depicted in Figure 1. In the *Data Preprocessing* two main data structures are built from the RDF(S) data, namely *keyword index* and *summary graph*. The graph represents a summarization of the RDF(S) data comprising only structural information. Thus, it contains only classes and properties, but no literals and instances. Literals and instances are integrated at runtime by means of the keyword index. In general, the keyword index is used to map keywords to elements of the RDF(S) data. The basic thinking behind the establishment of both structures is to enable a high performance search along with scalability. The summary graph (kept in memory) contains only as much information as necessary whereas the keyword index (native database) represents an entry point containing additional information.

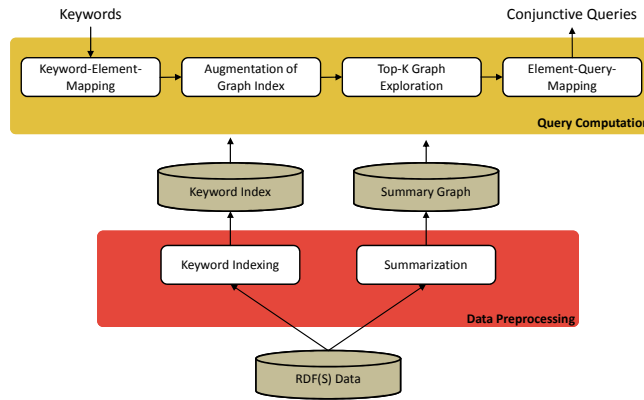


Fig. 1. Data Processing and Query Computation

One contribution of our work is the adjustment of the summary graph in order to leverage both, the class hierarchy and the property hierarchy. Figure 2 is intended to point out the main difference between the two approaches. The figure contains some example triples and the corresponding summary. In contrast to the summarization of Tran *et al.* a triple with two instances is mapped to a directed edge. Source and target correspond to the classes of the instances and the property of the triple is noted as label of the edge. In our approach every argument of a triple is mapped to a typed node (class or property node). The node of the subject and the node of the predicate are connected by a directed edge without label. The same applies for predicate and object. The only exception to that rule are triples including a 'rdfs:subClassOf' or 'rdfs:subPropertyOf'

predicate. In these cases the nodes are directly linked with an edge.

For computing SPARQL queries, the loose keywords are first disassembled into its constituent parts. For simplicity, let the keywords already be separated. Thus, there is a set of h terms $T = \{t_1, \dots, t_h\}$, whereas $t_i \in T$ is either a single word or a multi-word expression. In the following step, the terms are mapped to elements of the input RDF(S) data which are called *M-Resources*. In general, a term $t_i \in T$ is mapped to a set of j resources $R_i = \{r_{i_1}, \dots, r_{i_j}\}$. As follows, there are h sets of M-Resources R_1, \dots, R_h that are used for further processing. The resource sets R_1 to R_h were distributed to h threads and in each thread Z_i a graph exploration is performed on the prepared (augmented) summary graph for each M-Resource $r_{i_q} \in R_i$. Hence, many paths were explored starting from r_{i_q} . In case there is a resource r_c that is reached by any path in each thread a connecting subgraph G_s of the knowledge base can be constructed consisting of h paths. The resource r_c is called *C-Resource* which connects all paths with one another. The outcome of the graph search algorithm is a set of weighted subgraphs G_1, \dots, G_q , whose size can be restricted by an upper weight limit and a general time limit. The weighting strategy can be separated into two parts. The static part weights all nodes and edges in the preprocessing step (further information is presented in Tran *et al.* [4]). The dynamic part of the weighting takes place at the runtime of the system and concerns visited paths only. The weight w_{p_n} for a new path p_n is based on the path before p_o , the new edge e_a and node v_b , and the resulting path pattern m_n : $w(p_n) = w(p_o) + w(e_a) + w(v_b) + w(m_n)$. The last steps of *KOIOS++* are straightforward. For each subgraph G_1, \dots, G_q a conjunctive SPARQL query is constructed and presented as semantic network to the user. Subsequently, the user can select relevant queries to retrieve the corresponding answer from the triplestore.

Consider the example keywords 'person' and 'worked'. The first one is mapped to the class 'zpid:Person' and the second one is mapped to the property 'zpid:WorkedOn'. The exploration may track two paths starting from both nodes which may end up in the nodes 'zpid:Librarian' or 'zpid:Author'. As follows, two different SPARQL queries can be constructed.

The presented data structure has an inherent disadvantage because the queries may contain statements that are not included in the origin data, for instance, 'a librarian wrote an article'. This information is not necessarily incorrect, but if this relation is not entailed in the triplestore unnecessary graph traveling is done. As follows, the algorithm gets expensive and time-consuming. This is one reason, why we integrated the dynamic weighting as described above. If the graph traveling ends in a path p_n with an unwanted pattern the additional weight $w(m_n)$ is set to infinite (if not unwanted $w(m_n) = 0$). That means that the new path is (very likely) not considered for further exploration and query construction. Thus, the described graph mapping in combination with the dynamic weighting enables the integration of the property hierarchy without constructing unwanted SPARQL queries.

However, this kind of weighting strategy could also be used to foster ($w(m_n) < 0$) certain graph patterns which is important for explanation scenarios.

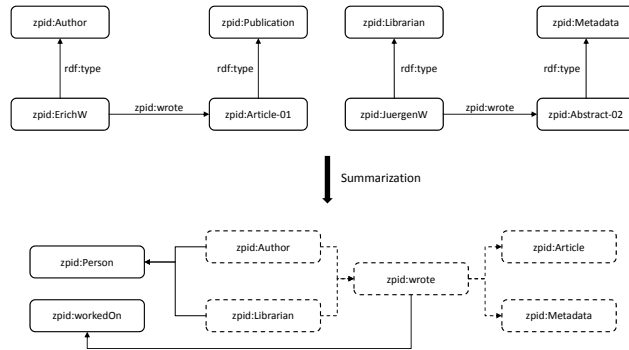


Fig. 2. Summarization of some triples

3 Conclusion

In this paper, we presented the semantic search engine *KOIOS++* that enables a keyword-based search on RDF(S) triple stores. It interprets the keywords and computes a set of SPARQL queries which can be selected by users to search the triple store. The RDF(S) data is mapped to a graph structure which is explored for the computation. In contrast to other approaches a predicate is not mapped to an edge, it is mapped to a node. To avoid unnecessary workload a sophisticated weighting strategy is applied. In particular, the strategy takes place at runtime whereas the exploration of new graph elements is based on the previous explored elements (conditional search). The presented approach enables not only heuristic reasoning on class hierarchies but also on the property hierarchies.

The next step of our work is to integrate SPARQL operators into our approach. Thus, it becomes possible to use words, such as "greater" or "smaller".

References

1. Achiezra, H., Golenberg, K., Kimelfeld, B., Sagiv, Y.: Exploratory keyword search on data graphs. In: Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data. SIGMOD '10, New York, USA, ACM (2010)
2. Cappellari, P., De Virgilio, R., Maccioni, A., Roantree, M.: A path-oriented rdf index for keyword search query processing. In: Proceedings of the 22Nd International Conference on Database and Expert Systems Applications - Volume Part II. (2011)
3. De Virgilio, R., Maccioni, A., Cappellari, P.: A linear and monotonic strategy to keyword search over rdf data. In Daniel, F., Dolog, P., Li, Q., eds.: Web Engineering. Springer Berlin Heidelberg (2013)
4. Tran, D.T., Wang, H., Rudolph, S., Cimiano, P.: Top-k exploration of query candidates for efficient keyword search on graph-shaped (rdf) data. In: Proc. of the 25th Intern. Conference on Data Engineering (ICDE'09), Shanghai, China (2009)
5. Wagner, A., Bicer, V., Tran, T.: Pay-as-you-go approximate join top-k processing for the web of data. In Presutti, V., d'Amato, C., Gandon, F., d'Aquin, M., Staab, S., Tordai, A., eds.: The Semantic Web: Trends and Challenges. Springer International Publishing (2014)

Supporting SPARQL Update Queries in RDF-XML Integration

Nikos Bikakis^{1*} Chrisa Tsinaraki² Ioannis Stavrakantonakis³
Stavros Christodoulakis⁴

¹ NTU Athens & R.C. ATHENA, Greece

² EU Joint Research Center, Italy

³ STI, University of Innsbruck, Austria

⁴ Technical University of Crete, Greece

Abstract. The *Web of Data* encourages organizations and companies to publish their data according to the Linked Data practices and offer SPARQL endpoints. On the other hand, the dominant standard for information exchange is XML. The *SPARQL2XQuery Framework* focuses on the automatic translation of SPARQL queries in XQuery expressions in order to access XML data across the Web. In this paper, we outline our ongoing work on supporting update queries in the RDF-XML integration scenario.

Keywords: SPARQL2XQuery, SPARQL to XQuery, XML Schema to OWL, SPARQL update, XQuery Update, SPARQL 1.1.

1 Introduction

The *SPARQL2XQuery Framework*, that we have previously developed [6], aims to bridge the heterogeneity issues that arise in the consumption of XML-based sources within Semantic Web. In our working scenario, mappings between RDF/S-OWL and XML sources are automatically derived or manually specified. Using these mappings, the SPARQL queries are translated on the fly into XQuery expressions, which access the XML data. Therefore, the current version of SPARQL2XQuery provides read-only access to XML data. In this paper, we outline our ongoing work on extending the SPARQL2XQuery Framework towards supporting SPARQL update queries.

Both SPARQL and XQuery have recently standardized their update operation semantics in the SPARQL 1.1 and XQuery Update Facility, respectively. We have studied the correspondences between the update operations of these query languages, and we describe the extension of our mapping model and the SPARQL-to-XQuery translation algorithm towards supporting SPARQL update queries.

Similarly to the motivation of our work, in the RDB-RDF interoperability scenario, D2R/Update [1] (a D2R extension) and OntoAccess [2] enable SPARQL update queries over relational databases. Regarding the XML-RDB-RDF interoperability scenario [5], the work presented in [3] extends the XSPARQL language [4] in order to support update queries.

* This work is partially supported by the EU/Greece funded KRIPIS: MEDA Project

2 Translating SPARQL Update Queries to XQuery

This section describes the translation of SPARQL update operations into XQuery expressions using the XQuery Update Facility. We present how similar methods and algorithms previously developed in the SPARQL2XQuery Framework can be adopted for the update operation translation. For instance, graph pattern and triple pattern translation are also used in the update operation translation. Note that, due to space limitations, some issues are presented in a simplified way in the rest of this section and several details are omitted.

Table 1 presents the SPARQL update operations and summarizes their translation in XQuery. In particular, there are three main categories of SPARQL update operations a) Delete Data; b) Insert Data; and c) Delete/Insert. For each update operation, a simplified SPARQL syntax template is presented, as well as the corresponding XQuery expressions. In SPARQL context, we assume the following sets, let tr be an *RDF triple set*, tp a *triple pattern set*, trp a set of triples and/or triple patterns, and gp a *graph pattern*. Additionally, in XQuery, we denote as xE_W , xE_I and xE_D the sets of *XQuery expressions* (i.e., FLOWR expressions) that have resulted from the translation of the graph pattern included in the Where, Insert and Delete SPARQL clauses, respectively. Let xE be a set of XQuery expressions, $xE(\$v_1, \$v_2, \dots, \$v_n)$ denote that xE are using (as input) the values assigned to XQuery variables $\$v_1, \$v_2, \dots, \$v_n$. Finally, xn denotes an *XML fragment*, i.e., a set of XML nodes, and xp denotes an *XPath expression*.

Table 1. Translation of the SPARQL Update Operations in XQuery

SPARQL		Translated XQuery Expressions
SPARQL Update Operation	Syntax Template ¹	
DELETE DATA	<pre> Delete data { tr } </pre>	<pre> delete nodes collection("http://dataset...")/xp1 ... delete nodes collection("http://dataset...")/xp_n let \$n1 := xn1 ... let \$n_n := xn_n let \$data1 := (\$n1, \$n_m, ...) // k, m, ... ∈ [1,n] ... let \$datap := (\$n1, \$n_x, ...) // j, y, ... ∈ [1,n] let \$insert_location1 := collection("http://xmldataset...")/xp1 ... let \$insert_location_p := collection("http://xmldataset...")/xp_p return(insert nodes \$data1 into \$insert_location1, ... insert nodes \$datap into \$insert_location_p) </pre>
INSERT DATA	<pre> Insert data { tr } </pre>	<pre> let \$where_gp := xE_w let \$insert_location1 := xp1 for \$iti in \$insert_location1 xE_I(\$where_gp, \$iti) return insert nodes into \$iti ... let \$where_gp := xE_w let \$insert_location_n := xp_n for \$iti in \$insert_location_n xE_I(\$where_gp, \$iti_n) return insert nodes into \$iti_n </pre>
DELETE / INSERT	<pre> (a) Delete { trp } Where { gp } (c) Delete { trp } Insert { trp } Where { gp } (b) Insert { trp } Where { gp } </pre>	<pre> (a) let \$where_gp := xE_w let \$insert_location1 := xp1 for \$iti in \$insert_location1 xE_I(\$where_gp, \$iti) return delete nodes \$delete_gp return insert nodes into \$iti ... let \$where_gp := xE_w let \$insert_location_n := xp_n for \$iti in \$insert_location_n xE_I(\$where_gp, \$iti_n) return insert nodes into \$iti_n (c) Translate Delete Where same as (a), then translate Insert Where same as (b) </pre>

¹ For simplicity, the WITH, GRAPH and USING clauses are omitted.

In the following examples, we assume that an RDF source has been mapped to an XML source. In particular, we assume the example presented in [6], where an RDF and an XML source describing persons and students have been mapped. Here, due to space limitation, we just outline the RDF and XML concepts, as well as the mappings that are involved in the following examples. In RDF, we have a class *Student* having several datatype properties, i.e., *FName*, *E-mail*, *Department*, *GivenName*, etc. In XML, we have an XML complex type *Student_type*, having an attribute *SSN* and several simple elements, i.e., *FirstName*, *Email*, *Dept*, *GivenName* etc. Based on the XML structure, the students' elements appear in the *\Persons\Student* path. We assume that the *Student* class has been mapped to the *Student_type* and the RDF datatype properties to the similar XML elements.

Delete Data. The Delete Data SPARQL operation removes a set of triples from RDF graphs. This SPARQL operation can be translated in XQuery using the Delete Nodes XQuery operation. Specifically, using the predefined mappings, the set of triples *tr* defined in the SPARQL Delete Data clause is transformed (using a similar approach such as the *BGP2XQuery* algorithm [6]) in a set of XPath expressions *XP*. For each $xp_i \in XP$ an XQuery Delete Nodes operation is defined.

Example 1. In this example, two RDF triples are deleted from an RDF graph. In addition to the mappings described above, we assume that the person "*http://rdf.gr/person1209*" in RDF data has been mapped to the person "*/Persons/Student[.@SSN=1209]*" in XML data.

<pre>SPARQL Delete Data query Delete data { <http://rdf.gr/person1209> ns:FName "John" . <http://rdf.gr/person1209> ns:E-mail "john@smith.com". }</pre>	⇒	<pre>Translated XQuery query delete nodes collection("http://xml.gr")/Persons/Student[.@SSN=1209]/FirstName[.="John"] delete nodes collection("http://xml.gr")/Persons/Student[.@SSN=1209]/Email[.="John@smith.com"]</pre>
---	---	--

Insert Data. The Insert Data SPARQL operation, adds a set of new triples in RDF graphs. This SPARQL operation can be translated in XQuery using the Insert Nodes XQuery operation. In the Insert Data translation, the set of triples *tr* defined in SPARQL are transformed into XML node sets *xn_i*, using the predefined mappings. In particular, a set of Let XQuery clauses is used to build the XML nodes and define the appropriate node nesting and grouping. Then, the location of the XML node insertion can be easily determined considering the triples and the mappings. Finally, the constructed nodes are inserted in their location of insertion using the XQuery Insert nodes clause.

Example 2. In this example, the RDF triples deleted in the previous example are re-inserted in the RDF graph.

<pre>SPARQL Insert Data query Insert data { <http://rdf.gr/person1209> ns:FName "John" . <http://rdf.gr/person1209> ns:E-mail "john@smith.com". }</pre>	⇒	<pre>Translated XQuery query let \$n1 := <FirstName>John</FirstName> let \$n2 := <Email>john@smith.com</Email> let \$data1 := (\$n1, \$n2) let \$insert_location1 := collection("http://xml.gr")/Persons/Student[.@SSN=1209] return insert nodes \$data1 into \$insert_location1</pre>
---	---	--

Insert / Delete. The Delete/Insert SPARQL operations are used to remove and/or add a set of triples from/to RDF graphs, using the bindings that resulted from the evaluation

of the graph pattern defined in the `Where` clause. According to the SPARQL 1.1 semantics, the `Where` clause is the first one that is evaluated. Then, the `Delete/Insert` clause is applied over the produced results. Especially, in case, that both `Delete` and `Insert` operations exist, the deletion is performed before the insertion, and the `Where` clause is evaluated once. The `Delete` and the `Insert` SPARQL operations can be translated to XQuery using the `Delete Nodes` and `Insert Nodes` operations, respectively. In brief, initially the graph pattern used in the `Where` clause is translated to XQuery expressions xE_W (similarly as in the *GP2XQuery* algorithm [6]). Then, the graph pattern used in the `Delete/Insert` clause is translated to XQuery expressions xE_D/xE_I (as it is also in the *BGP2XQuery* algorithm [6]) using also the bindings that resulted from the evaluation of xE_W .

Example 3. In this example, the `Where` clause selects all the students studying in a computer science (CS) department. Then, the `Delete` clause deletes all the triples that match with its triple patterns, using the `?student` bindings determined from the `Where` clause. In particular, from all the retrieved students (i.e., CS students), the students which have as first name the name "John" should be deleted.

<pre>SPARQL Delete query ⇒ Translated XQuery query Delete{ ?student ns: FName "John" . }Where{ ?student ns: Department "CS" . }</pre>	<pre>let \$where_gp := collection("http://xml.gr")/Persons/Student[./Dept="CS"] let \$delete_gp := \$where_gp[./FirstName="John"] return delete nodes \$delete_gp</pre>
---	---

Example 4. In this example, the `Where` clause selects all the students studying in a CS department, as well as their first names. Then, the `Insert` clause creates new triples according to its triple patterns, using the `?student` and `?name` bindings determined from the `Where` clause. In particular, a new triple having as predicate "`ns: GivenName`" and as object the first name of the `?student`, is inserted for each `?student`.

<pre>SPARQL Insert query ⇒ Translated XQuery query Insert{ ?student ns: GivenName ?name . }Where{ ?student ns: FName ?name . ?student ns: Department "CS" . }</pre>	<pre>let \$where_gp := collection("http://xml.gr")/Persons/Student[./Dept="CS"] let \$insert_location1 := \$where_gp for \$it1 in \$insert_location1 let \$insert_gp1 := <GivenName>{fn:string(\$it1/FirstName)}</GivenName> return insert nodes \$insert_gp1 into \$it1</pre>
---	--

References

1. Eisenberg V., Kanza Y.: "D2RQ/update: updating relational data via virtual RDF". In WWW 2012
2. Hert M., Reif G., Gall H. C.: "Updating relational data via SPARQL/update". In EDBT/ICDT Workshops 2010.
3. Ali M.I., Lopes N., Friel O., Mileo A.: "Update Semantics for Interoperability among XML, RDF and RDB". In APWeb 2013
4. Bischof S., Decker S., Krennwallner T., Lopes N., Polleres A.: "Mapping between RDF and XML with XSPARQL". J. Data Semantics 1(3), (2012)
5. Bikakis N., Tsinaraki C., Gioldasis N., Stavrakantonakis I., Christodoulakis S.: "The XML and Semantic Web Worlds: Technologies, Interoperability and Integration. A survey of the State of the Art". In Semantic Hyper/Multi-media Adaptation: Schemes and Applications, Springer 2013
6. Bikakis N., Tsinaraki C., Stavrakantonakis I., Gioldasis N., Christodoulakis S.: "The SPARQL2XQuery Interoperability Framework". World Wide Web Journal (WWWJ), 2014

CURIOS: Web-based Presentation and Management of Linked Datasets

Hai H. Nguyen¹, Stuart Taylor¹, Gemma Webster¹, Nophadol Jekjantuk¹,
Chris Mellish¹, Jeff Z. Pan¹, and Tristan ap Rheinallt²

¹ dot.rural Digital Economy Hub, University of Aberdeen, Aberdeen AB24 5UA, UK
² Hebridean Connections, Ravenspoint, Kershader, Isle of Lewis HS2 9QA, UK

1 Introduction

A number of systems extend the traditional web and Web 2.0 technologies by providing some form of integration with semantic web data [1,2,3]. These approaches build on tested content management systems (CMSs) for facilitating users in the semantic web. However, instead of directly managing existing linked data, these systems provide a mapping between their own data model to linked datasets using an RDF or OWL vocabulary. This sort of integration can be seen as a read or write only approach, where linked data is either imported into or exported from the system. The next step in this evolution of CMSs is a full integration with linked data: allowing ontology instances, already published as linked data, to be directly managed using widely used web content management platforms. The motivation is to keep data (i.e., linked data repositories) loosely-coupled to the tool used to maintain them (i.e., the CMS).

In this poster we extend [3], a query builder for SPARQL, with an update mechanism to allow users to directly manage their linked data from within the CMS. To make the system sustainable and extensible in future, we choose to use Drupal as the default CMS and develop a module to handle query/update against a triple store. Our system, which we call a *Linked Data Content Management System* (Linked Data CMS) [4], performs similar operations to those of a traditional CMS but whereas a traditional CMS uses a data model of content types stored in some relational database back end, a Linked Data CMS performs CRUD (create, read, update and delete) operations on linked data held in a triple store. Moreover, we show how the system can assist users in producing and consuming linked data in the cultural heritage domain and introduce 2 case studies used for system evaluation.

2 Using CURIOS

We introduce CURIOS, an implementation of a Linked Data CMS.³ A dataset managed by CURIOS needs to have a structure described by an OWL ontology that imports a small CURIOS “upper ontology”. It must relate some of its classes and properties to constructs in that ontology. This has the benefit that they can

³ Available open-source at <https://github.com/curiosproject/curios>.

be recognised and treated specially by the generated website. For instance, an image can be presented in a special way (see Fig. 1) if it is an instance of the `hc:ImageFile` class and its URL is provided by the `hc:URL` property.

Once the ontology is defined, it is necessary to provide a separate description of which parts of the data (and the level of detail) are to be managed by the website. This description takes the form of an application-dependent *configuration file* which is loaded as part of the Drupal module. This file describes the classes, fields, and relationships to be shown in the website and how these relate to the constructs of the ontology [4]. Although the configuration file could be generated automatically, it is a declarative description and can easily be edited by hand. The configuration file centralises the maintenance of the structure of the CMS with respect to the ontology, e.g., if a new type of page is required, the user can update the configuration and then run the Linked Data CMS mapping to create the required Drupal entities. Additionally our approach can handle some changes to the schema of the ontology. For example if a change in the ontology occurs, such as a domain/range, additional classes or a change of URIs, then the configuration can be reloaded to synchronise Drupal with the ontology schema.

When the CURIOS Drupal module is initialised, it automatically creates a set of Drupal resources and Views based on the configuration file, along with an additional set of pages allowing the linked data to be maintained via CRUD operations. Drupal site administrators can then maintain the website generated by the configuration in the same way as a regular Drupal site.

2.1 Browsing and Update Functionalities


4 Caverstay
 Croft 4 was first occupied by Donald Mackinnon and then by his son Roderick.

The croft was particularly congested in the early years of the 20th century, supporting four large families of Mackinnons. The situation forced many to leave the village for Stornoway, the mainland or abroad.

[Back to listing](#)

Title: 4 Caverstay
Record Type: Crofts and Residences
Gaelic Name: 4 Cabhairstaith
Type: Croft
Record Owned By: CEP
Record Maintained By: CEP
Subject Id: 7560

[Louis Mackinnon & family, Caverstay](#)



Lived Here
 Angus Mackinnon
 Johanna Mackinnon
 Donald Mackinnon
 Mary Bell Macleod
 Ann Mackinnon
 Catherine MacLennan
 Roderick Mackinnon
 Donald Mackinnon
 Louis Mackinnon
 John Mackinnon
 Mary Ann Mackinnon
 Louis Mackinnon
 Mary Macdonald
 Roderick Mackinnon
 Christina Mackinnon

Associated With
 Euphemia Maciver
 Ruairidh Rob Mackinnon I: Memories...
 Ruairidh Rob Mackinnon II: Off to...
 John Murdo Macdonald
 Catherine Macdonald

Located At
 Caverstay

Figure 1: Details of a croft

CURIOS allows users to browse and update their linked data in a triple store directly without transforming RDF triples to Drupal content and vice versa. A CURIOS record consists of a set of RDF triples where the subject is a unique URI representing the record identifier. For browsing, depending on

different conditions, CURIOS presents data in different ways. For instance, a list of records or details of a record (see Fig. 1) will be displayed depending on whether the record URI is provided. To navigate between linked individuals, object properties of an RDF individual are presented as hyperlinks to other records instead of the normal text used for datatype properties.

Users are also able to create, update, or delete a record via a user-friendly GUI. Firstly, CURIOS assists users entering data by providing different widgets depending on the datatype the user wants to edit (Fig. 2a). For instance, with geographical coordinates, a map is displayed to allow users to choose a location rather than to type in the coordinates as text. Secondly, to prevent users from entering incorrect values for some special properties such as an occupation or a type of place, an auto-complete widget is provided. Thirdly, it is typical that in the cultural heritage domain, temporal data such as dates are rather vague and not recorded in a consistent format. To facilitate users during data entry process, CURIOS provides a simple treatment to vague dates by introducing the `hc:DateRange` class which consists of two datetime datatype properties: `hc:dateFrom` and `hc:dateTo`. A user can enter an exact date or a vague date such as a year, a season in a year, a decade, a century, etc, and CURIOS can convert the vague date into an appropriate instance of `hc:DateRange`. Finally, to manage object properties (i.e., links) between individuals, CURIOS allows property add and remove operations as presented in Fig. 2b, which are then mapped onto corresponding SPARQL update queries, e.g., `INSERT` and `DELETE`, to insert and remove appropriate triples.

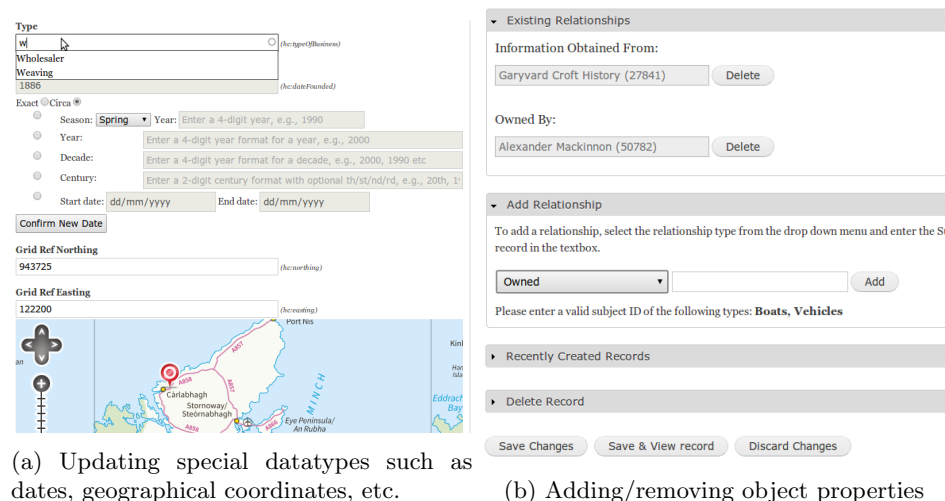


Figure 2: Updating Linked Data in CURIOS

2.2 Use of the Triple Store

Although in principle we could use any SPARQL 1.1 compliant triple store/SPARQL server, in practice we are using Jena Fuseki [5]. The reasoner in Fuseki

creates a complete graph of all the consequences of the stated information when a SPARQL query is presented, and this is kept in a memory cache. Unfortunately, this reasoning has to be repeated after an update has been performed, and especially with complex updates, this can take an unreasonable amount of time that can affect the website's responsiveness. Also, although one ideally wants to show a user all the inferred information in order that they have an accurate model of the system's knowledge, if they are allowed to specify arbitrary updates on this then they may remove a piece of inferred information which is then re-inferred whenever the reasoner is next invoked. For these two reasons, we perform all updates via a Fuseki endpoint where no reasoning takes place. A second endpoint, where reasoning is enabled, is used for normal browsing. With this method, the information shown for browsing gradually becomes out of date as nothing prompts the recomputation of the inference graph. This is overcome by allowing the user to explicitly invoke the recomputation or by having a process outside of the website causing the recomputation at regular time intervals.

3 Case Studies and Future Work

To test the generality of our system, we conduct 2 case studies, one involving historical societies based in the Western Isles of Scotland (Hebridean Connections) and another one with the local historical group at Portsoy, a fishing village located in the North East of Scotland. The dataset used in the Hebridean Connections case study consists of over 45,000 records with about 850,000 RDF triples (before inference), incorporated within a relatively simple OWL ontology. The Portsoy case study uses a similar ontology with 1370 records and 23258 RDF triples (before inference). The Drupal website which we have built with the software is already being used by Hebridean Connections at <http://www.hebrideanconnections.com>.

In future we plan to make the system easier to set up for naïve users as well as to evaluate our system with different SPARQL servers/RDF stores.

Acknowledgements The research described here is supported by the award made by the RCUK Digital Economy programme to the dot.rural Digital Economy Hub; award reference: EP/G066051/1.

References

1. Krötzsch, M., Vrandečić, D., Völkel, M.: Semantic MediaWiki. In: ISWC. (2006)
2. Corlosquet, S., Delbru, R., Clark, T., Polleres, A., Decker, S.: Produce and consume Linked Data with Drupal! In: ISWC. (2009)
3. Clark, L.: SPARQL Views: A Visual SPARQL Query Builder for Drupal. In Polleres, A., Chen, H., eds.: ISWC Posters&Demos. Volume 658 of CEUR Workshop Proceedings., CEUR-WS.org (2010)
4. Taylor, S., Jekjantuk, N., Mellish, C., Pan, J.Z.: Reasoning driven configuration of linked data content management systems. In: JIST 2013. (2013)
5. Seaborne, A.: Fuseki: serving RDF data over HTTP. http://jena.apache.org/documentation/serving_data/ (2011) Accessed: 2012-10-27.

The uComp Protégé Plugin for Crowdsourcing Ontology Validation

Florian Hanika¹, Gerhard Wohlgenannt¹, and Marta Sabou²

¹ WU Vienna

{florian.hanika,gerhard.wohlgenannt}@wu.ac.at

² MODUL University Vienna

marta.sabou@modul.ac.at

Abstract. The validation of ontologies using domain experts is expensive. Crowdsourcing has been shown a viable alternative for many knowledge acquisition tasks. We present a Protégé plugin and a workflow for outsourcing a number of ontology validation tasks to Games with a Purpose and paid micro-task crowdsourcing.

Keywords: Protégé plugin, ontology engineering, crowdsourcing, human computation

1 Introduction

Protégé³ is a well-known free and open-source platform for ontology engineering. Protégé can be extended with *plugins* using the Protégé Development Kit. We present a plugin for crowdsourcing ontology engineering tasks, as well as the underlying technologies and workflows. More specifically, the plugin supports outsourcing of some typical ontology validation tasks (see Section 2.2) to Games with a Purpose (GWAP) and paid-for crowdsourcing.

The research question our work focuses on is how to integrate ontology engineering processes with human computation (HC), to study which tasks can be outsourced, how this affects the quality of the ontological elements, and to provide tool support for HC. This paper concentrates on the integration process and tool support. As manual ontology construction by domain experts is expensive and cumbersome, HC helps to decrease cost and increase scalability by distributing jobs to multiple workers.

2 The uComp Protégé Plugin

The uComp Protégé Plugin allows the validation of certain parts of an ontology, which makes it useful in any setting where the quality of an ontology is questionable, for example if an ontology was generated automatically with ontology learning methods, or if a third-party ontology needs to be evaluated before use. This section covers the uComp API, and the uComp Protégé plugin (functionality and installation).

³ protege.stanford.edu

2.1 The uComp API

The Protégé plugin sends all validation tasks to the uComp HC API. Depending on the settings, the API further delegates the tasks to a GWAP or to CrowdFlower⁴. CrowdFlower is a platform for paid micro-task crowdsourcing. The uComp API⁵ currently supports classification tasks (other task types are under development). The API user can create new HC jobs, cancel jobs, and collect results from the service. All communication is done via HTTP and JSON.

2.2 The plugin

The plugin supports the validation of various parts of an ontology: *relevance of classes*, *subClassOf relations*, *domain and range axioms*, *instanceOf relations*, etc. The general usage pattern is as follows: the user selects the respective part of the ontology, provides some information for the crowdworkers, and submits the job. As soon as available, the results are presented to the user.

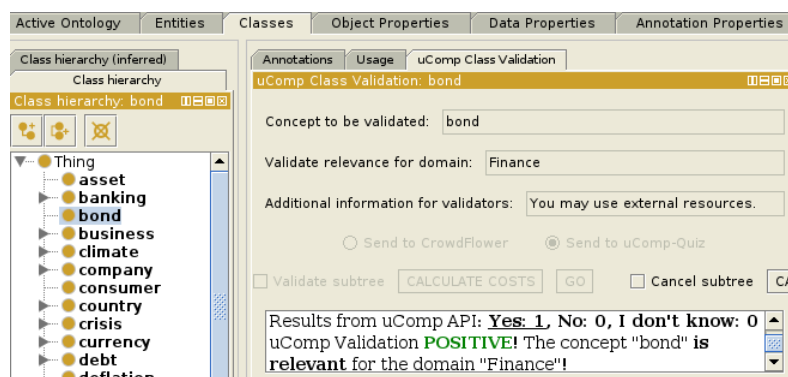


Fig. 1. Class relevance check for class *bond* including results.

Class relevance check For the sake of brevity, we only describe the *Class Relevance Check* and *SubClass Relation Validation* in some detail. The other task types follow a very similar pattern. Class Relation Check helps to decide if a given class (or a set of classes) – based on the class label – is relevant for the given domain. Figure 1 shows an example class relevance check for the class *bond*. After selecting a class, the user can enter an ontology *domain* (here: *Finance*) to validate against, and give additional advice to the crowdworkers. Furthermore, (s)he can choose between the GWAP and CrowdFlower for validation. If CrowdFlower is

⁴ www.crowdflower.com

⁵ tinyurl.com/mkarmk9

selected, the expected cost of the job can be calculated. The *validate subtree* option allows to validate not only the current class, but also all its subclasses (recursively). To validate the whole ontology in one go, the user selects the root class (Thing) and marks the validate subtree option. When available, the results of the HC task are presented in a textbox. In Figure 1 only one judgment was collected – the crowdworker stated that class *bond* is relevant for the domain.

Validation of SubClass Relations With this component, a user can ask the crowd if there exists a subClass relation between a given class and its superclasses.

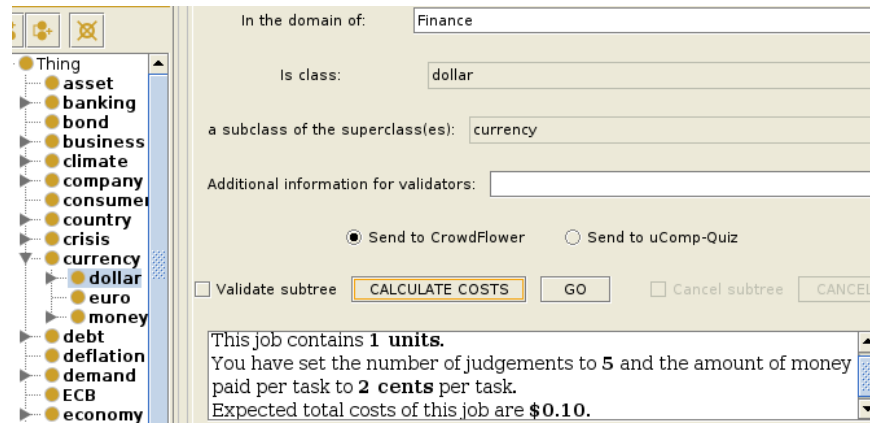


Fig. 2. Validation of class *dollar* and its superclass *currency*.

Similar to the class relevance check, users can set the ontology domain, and choose CrowdFlower or GWAP (“uComp-Quiz”). In Figure 2 the subClass relation between *dollar* and *currency* is evaluated. Before sending to CrowdFlower, expected costs can be calculated as number of units (elements to evaluate) multiplied by number of judgments per unit and payment per judgment.

2.3 Installation and Configuration

As the uComp plugin is part of the official Protégé repository, it can easily be installed from within Protégé with *File* → *Check for plugins* → *Downloads*. To configure and use the plugin, the user needs to create a file name `ucomp_api_settings.txt` in folder `.Protege`. The file contains the uComp API key⁶, the number of judgments per unit which we be collected, and the payment per judgment (if using CrowdFlower), for example: `abcdefghijklmnopqrst,5,2`

⁶ For API requests see tinyurl.com/mkarmk9

Detailed information about the functionality, usage and installation of the plugin is provided with the plugin documentation.

3 Related Work

Human computation outsources computing steps to humans, typically for problems computers can not solve (yet). Together with altruism, fun (as in GWAPs) and monetary incentives are central ways to motivate humans to participate. Early work in the field of GWAPs was done by von Ahn [1]. Games have successfully been used for example in ontology alignment [6] or to verify class definitions [3]. Micro-task crowdsourcing is very popular recently in knowledge acquisition and natural language processing, and has also been integrated into the popular NLP framework GATE [2]. A number of studies show that crowdworkers provide results of similar quality as domain experts [4, 5].

4 Conclusions

In this paper we introduce a Protégé plugin for validating ontological elements, and its integration into a human computation workflow. The plugin delegates validation tasks to a GWAP or to CrowdFlower and displays the results to the user. Future work includes an extensive evaluation of various aspects: HC workflows in ontology engineering, quality of crowdsourcing results, and the usability of the plugin itself.

Acknowledgments. The work presented was developed within project uComp, which receives the funding support of EPSRC EP/K017896/1, FWF 1097-N23, and ANR-12-CHRI-0003-03, in the framework of the CHIST-ERA ERA-NET.

References

1. von Ahn, L.: Games With a Purpose. *Computer* 39(6), 92–94 (2006)
2. Bontcheva, K., Roberts, I., Derczynski, L., Rout, D.: The GATE Crowdsourcing Plugin: Crowdsourcing Annotated Corpora Made Easy. In: *Proc. of the 14th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*. ACL (2014)
3. Markotschi, T., Voelker, J.: Guess What?! Human Intelligence for Mining Linked Data. In: *Proceedings of the Workshop on Knowledge Injection into and Extraction from Linked Data (KIELD) at the International Conference on Knowledge Engineering and Knowledge Management (EKAW)* (2010)
4. Noy, N.F., Mortensen, J., Musen, M.A., Alexander, P.R.: Mechanical Turk As an Ontology Engineer?: Using Microtasks As a Component of an Ontology-engineering Workflow. In: *Proc. 5th ACM WebSci Conf.* pp. 262–271. WebSci '13, ACM (2013)
5. Sabou, M., Bontcheva, K., Scharl, A., Föls, M.: Games with a Purpose or Mechanised Labour?: A Comparative Study. In: *Proc. of the 13th Int. Conf. on Knowledge Management and Knowledge Technologies.* pp. 1–8. i-Know '13, ACM (2013)
6. Siorpaes, K., Hepp, M.: Games with a Purpose for the Semantic Web. *IEEE Intelligent Systems* 23(3), 50–60 (2008)

Frame-Semantic Web: a Case Study for Korean*

Jungyeul Park^{†‡}, Sejin Nam[‡], Youngsik Kim[‡]
Younggyun Hahm[‡], Dosam Hwang^{‡§}, and Key-Sun Choi[‡]

[†]UMR 6074 IRISA, Université de Rennes 1, France

[‡]Semantic Web Research Center, KAIST, Republic of Korea

[§]Department of Computer Science, Yeungnam University, Republic of Korea
<http://semanticweb.kaist.ac.kr>

Abstract. FrameNet itself can become a resource for the Semantic Web. It can be represented in RDF. However, mapping FrameNet to other resources such as Wikipedia for building a knowledge base becomes more common practice. By such mapping, FrameNet can be considered to provide capability to describe the semantic relations between RDF data. Since the FrameNet resource has been proven very useful, multiple global projects for other languages have arisen over the years, parallel to the original English FrameNet. Accordingly, significant steps were made to further develop FrameNet for Korean. This paper presents how frame semantics becomes a frame-semantic web. We also provide the Wikipedia coverage by Korean FrameNet lexicons in the context of constructing a knowledge base from sentences in Wikipedia to show the usefulness of our work on frame semantics in the Semantic Web environment.

Keywords: Semantic Web, Frame Semantics, FrameNet, Korean FrameNet.

1 Introduction

FrameNet [1]¹ is a both human- and machine-readable large-scale on-line lexical database, not only consists of thousands and thousands of words and sentences, but, moreover, an extensive and complex range of semantic information as well. Based on a theory of meaning called frame semantics, FrameNet strongly supports an idea that the meanings of words and sentences can be best understood on the basis of a semantic frame, a coherent conceptual structure of a word describing a type of event, relation, or entity and the participants in it. It is believed that semantic frames of related concepts are inseparable from each other, so that, one cannot have complete understanding of a word, without knowledge of all the semantic frames related to that word. FrameNet itself serves as a great example of such a principle, wherein 1,180 semantic frames closely link together by a system of semantic relations and provide a solid basis for reasoning about the meaning of the entire text.

* This work was supported by the IT R&D program of MSIP/KEIT. [10044494, WiseKB: Big data based self-evolving knowledge base and reasoning platform]

¹ <https://framenet.icsi.berkeley.edu>

FrameNet itself can become a resource for the Semantic Web as represented in RDF/OWL [2, 3]. Mapping FrameNet to other resources such as Wikipedia for building a knowledge base can also be considered to provide capability to describe the semantic relations between RDF data. Since the FrameNet resource has been proven useful in the development of a number of other NLP applications, even in the Semantic Web environment such as in [4], multiple global projects have arisen over the years, parallel to the original English FrameNet, for a wide variety of languages around the world. In addition to Brazilian Portuguese², French³, German (the SALSA Project)⁴, Japanese⁵, Spanish⁶, and Swedish⁷, significant steps were made to further develop FrameNet for Korean, and the following sections of this paper present the process and mechanisms. By using FrameNet, it can become a frame-semantic web where frame semantics is enabled for the Semantic Web. We also provide the Wikipedia coverage by Korean FrameNet lexicons in the context of constructing a knowledge base from sentences in Wikipedia. It can show how the frame-semantic web would be useful in the Semantic Web environment.

2 Building a Database of Frame Semantic Information for Korean

We describe the manual construction of a FrameNet-style annotated corpus for Korean translated from the FrameNet corpus and its FE transfer based on English-Korean alignment using cross-linguistic projection proposed in [5, ?]. We also explain this process by using the translated Korean FrameNet corpus and its counterpart English corpus as our bilingual parallel corpus. We propose a method for mapping a Korean LU to an existing FrameNet-defined frame to acquire a Korean frame semantic lexicon. Finally, we illustrate a self-training technique that can build a database of large-scale frame semantic information for Korean.

Manual Construction: The development of FrameNet for Korean has been the central goal of our project, and we have chosen to perform this task by starting off with “manually translating” the already-existing FrameNet from English to Korean language. Such decision was made on the grounds that, even though obtaining a large set of data through means of manual translation can be a difficult, costly and time-consuming process, its expected advantages indeed far outweigh the charge in the long run. The fact that only humans can really develop a true understanding and appreciation of the complexities of languages, subject knowledge and expertise, creativity and cultural sensitivity also makes manual translation the best option to adopt for our project. Expert translators

² <http://www.ufjf.br/framenetbr>

³ <https://sites.google.com/site/anrasfalda>

⁴ <http://www.coli.uni-saarland.de/projects/salsa>

⁵ <http://jfn.st.hc.keio.ac.jp>

⁶ <http://sfn.uab.es/SFN>

⁷ <http://spraakbanken.gu.se/eng/swefn>

performed manual translation for all FrameNet full text annotated corpus with a word alignment recommendation system. A guideline manual for translating the FrameNet-style annotated corpus to Korean sentences was prepared for the clean transferring of English FrameNet annotated sentences to Korean.

Automatic Construction: We also extend previous approaches described in [5] using a bilingual English-Korean parallel corpus. Assuming that the same kinds of frame elements (FEs) exist for each frame for the English and Korean sentences, we achieve the cross-linguistic projection of English FE annotation to Korean via alignment of tokenized English and Korean sentences. English FE realization can be projected to its corresponding Korean sentences by transforming consecutive series of Korean tokens in the Korean translation of any given sentence. Since the alignment of English tokens to Korean tokens defines the transformation, the success of token alignment is crucial for the cross-linguistic projection process. For frame population to Korean lexical units (LUs), we present our method for the automatic creation of the Korean frame semantic lexicon for verbs in this section. We start by finding an appropriate translation for each verb to create a mapping between a Korean LU and an existing FrameNet-defined frame. In contrast to mapping from one sense to one frame, mapping to more than one frame requires using a further disambiguation process to select the most probable frame for a given verb. We use maximum likelihood estimation (MLE) for possible frames from the existing annotated corpora to select the correct frame. For the current work, we only used FrameNets lexicographic annotation to estimate MLE. We use the *Sejong* predicate dictionary⁸ for frame semantic lexicon acquisition. We place 16,807 Korean verbs in FrameNet-defined frames, which constitute 12,764 distinctive orthographic units in Korean. We assume that FEs with respect to the assigned frame for Korean LUs are directly equivalent to the FEs in the corresponding English frames. Thus, we do not consider redefining FEs specifically for Korean.

Bootstrapping Frame-Semantic Information: Self-training for frame semantic role projection consists of annotating FrameNet-style semantic information, inducing word alignments between two languages, and projecting semantic information of the source language onto the target language. We used the bilingual parallel corpus for self-training, and a probabilistic frame-semantic parser [6] to annotate semantic information of the source language (English). Then, we induced an HMM word alignment model between English and Korean with a statistical machine translation toolkit. Finally, we projected semantic roles information from the English onto the Korean sentences. For the experiment, we employed a large bilingual English-Korean parallel corpus, which contains almost 100,000 bilingual parallel sentences to bootstrap the semantic information. During self-training, errors in the original model would be amplified in the new model; thus, we calibrate the results of the frame-semantic parser by using the confidence score of the frame-semantic parser as a threshold. As a result, 120,621 pairs of frames with their FEs are obtained and among them 30,149 are unique; 715 frames are used for 10,898 different lexica.

⁸ <http://www.sejong.or.kr>

3 Linking FrameNet to Wikipedia

DBpedia⁹ is a knowledge base constructed from Wikipedia based on DBpedia ontology (DBO). DBO can be viewed as a vocabulary to represent knowledge in Wikipedia. However, DBO is a Wikipedia-Infobox-driven ontology. That is, although DBO is suitable to represent essential information of Wikipedia, it does not guarantee enough to represent knowledge in Wikipedia written in a natural language. In overcoming such a problem, FrameNet has been considered useful in linguistic level as a language resource representing semantics. We calculate the Wikipedia coverage rate by DBO and FrameNets LUs to match the relation instantiation from DBpedia and FrameNet to Wikipedia. Before we calculate the Wikipedia coverage rate, we need to know which sentences within Wikipedia actually contain knowledge. We define that a typical sentence with *extractable knowledge* can be linked to DBpedia entities as a triple. From almost three millions sentences in Korean Wikipedia, we find over four millions predicates for cases where only a subject appears, only an object appears, or both of a subject and an object appear (2.11 predicates per sentence). We obtain 6.92% and 95.19% for DBO and FrameNets LUs, respectively. The shortage of DBO can be explained that DBO is too small to cover actual predicates in Wikipedia only by pre-defined predicates in DBO. However, FrameNet gives almost full coverage for sentences with extractable knowledge, which is very promising for extracting and representing knowledge in Wikipedia using FrameNet.

4 Discussion and Conclusion

Throughout this paper, by building a database of frame semantic information, we explained that FrameNet can become a resource for the Semantic Web and it can gather lexical linked data and knowledge patterns with almost full coverage for Wikipedia.

References

1. Ruppenhofer, J., Ellsworth, M., Petruck, M.R.L., Johnson, C.R., Scheffczyk, J.: FrameNet II: Extended Theory and Practice. (2010)
2. Narayanan, S., Fillmore, C.J., Baker, C.F., Petruck, M.R.L.: FrameNet Meets the Semantic Web: A DAML+OIL Frame Representation. In: Proc. of AAAI-02.
3. Narayanan, S., Baker, C.F., Fillmore, C.J., Petruck, M.R.L.: FrameNet Meets the Semantic Web: Lexical Semantics for the Web. In: ISWC 2003.
4. Fossati, M., Tonelli, S., Giuliano, C.: Frame Semantics Annotation Made Easy with DBpedia. In: Proc. of CrowdSem2013. 69–78
5. Padó, S., Lapata, M.: Cross-lingual Bootstrapping of Semantic Lexicons: The Case of FrameNet. In: Proc. of AAAI-05.
6. Das, D., Schneider, N., Chen, D., Smith, N.A.: Probabilistic Frame-Semantic Parsing. In: Proc. of NAACL 2010.

⁹ <http://dbpedia.org/About>

SparkRDF: Elastic Discreted RDF Graph Processing Engine With Distributed Memory

Xi Chen, Huajun Chen, Ningyu Zhang, and Songyang Zhang

College of Computer Science, Zhejiang University,
Hangzhou 310027, China
{xichen, huajunsir, zxlzr, syzhang1991}@zju.edu.cn

Abstract. With the explosive growth of semantic data on the Web over the past years, many large-scale RDF knowledge bases with billions of facts are generating. This poses significant challenges for the storage and retrieval of big RDF graphs. In this paper, we introduce the SparkRDF, an elastic discreted semantic graph processing engine with distributed memory. To reduce the high I/O and communication costs for distributed platforms, SparkRDF implements SPARQL query based on Spark, a novel in-memory distributed computing framework. All the intermediate results are cached in the distributed memory to accelerate the process of iterative join. To reduce the search space and memory overhead, SparkRDF splits the RDF graph into the multi-layer subgraphs based on the relations and classes. For SPARQL query optimization, SparkRDF generates an optimal execution plan for join queries, leading to effective reduction on the size of intermediate results, the number of joins and the cost of communication. Our extensive evaluation demonstrates the efficiency of our system.

Keywords: Big RDF Graph, SPARQL, SPARK, Distributed memory.

1 Introduction

With the development of Semantic technologies and Web 3.0, the amount of Semantic Web data represented by the Resource Description Framework (RDF) is increasing rapidly. Traditional RDF systems are mainly facing two challenges. i) scalability: the ability to process the big RDF data. Most existing RDF systems are based on single node[4][1], which are easily vulnerable to the growth of the data size because they usually need to load large indexes into the limited memory. ii) real-time: the capacity to implement SPARQL query over big RDF graph in near real time. For highly iterative SPARQL query, existing MapReduce-based RDF systems suffer from high I/O cost because of iteratively reading and writing large intermediate results in disk[3].

In this paper, we introduce SparkRDF, an elastic discreted RDF graph processing system with distributed memory. It is based on Spark, a in-memory cluster computing system which is quite suitable for large-scale real-time iterative computing jobs[5]. SparkRDF splits the big RDF graph into MESGs(Multi-layer Elastic SubGraph) based on relations and classes by creating 5 kinds of

indexes(C,R,CR,RC,CRC) with different grains to cater for diverse triple patterns(TP). These index files on demand are modeled as RDSG(Resilient Discreted SubGraph), a collection of in-memory semantic subgraph objects partitioned across machines, which can implement SPARQL query by a series of basic operators. All intermediate results(IR), which are also regarded as the RDSG, remain in the distributed memory to support further fast joins. Based on the query model, several corresponding optimization tactics are then presented.

The remaining of this paper is organized as follows. Section 2 introduces the index data model and iterative query model of *SparkRDF*. In Section 3, we present the results of our experiments. Finally, we conclude and discuss the future work in Section 4.

2 SparkRDF

2.1 Index Data Model: MESH

We create the index model called MESH based on relations and classes, which extends traditional vertical partitioning solution by connecting class indexes with predicate indexes, whose goal is to construct a smaller index file for every TP in the SPARQL query. At the same time, as it is uncertain that the class information about the entities can be given in the SPARQL query, the SparkRDF needs a multi-layer elastic index scheme to meet the query need for different kinds of TP. Specifically, we first construct the class indexes(C) and relation indexes(R). Then a set of finer-grained index files(CR,RC,CRC) are created by joining the two kinds of index files. All the index files are stored in the HDFS.

2.2 RDSG-based Iterative Query Model

For SparkRDF, all the index files and IRs can be modeled as an unified concept called RDSG(Resilient Discreted SubGraph). It is a distributed memory abstraction that lets us perform in-memory query computations on large clusters by providing the following basic operators: RDSG_Gen, RDSG_Filter, RDSG_G_Partition, RDSG_Join. Figure 1 illustrates the RDSG-based query process. Every job corresponds to one query variable.

2.3 Optimization techniques

Based on the data model and query model, several optimization strategies are made to improve query efficiency. First, TR-SPARQL refers to a Type-Restrictive SPARQL by passing variable's implicit class message to corresponding TPs that contains the variable. It cuts down the number of task (remove the TPs whose predicate is rdf:type) and the cost of parsing every TP(form a more restrictive index file). Then we use a selectivity-based greedy algorithm to design a optimal execution order of TPs, greatly reducing the size of IR. At last, the location-free repartitioning is implemented to avoid the shuffling cost in the distributed join. It ignores the partitioning information of index files, while repartitioning the data with the same join key to the same node.

3 Evaluation

We implement the experiment on a cluster with three machines. Each node has 16 GB DDR3 RAM, 8-core Intel Xeon(R) E5606 CPUs at 2.13GHz. We compare SparkRDF with the state-of-the-art centralized RDF-3X and distributed HadoopRDF. We run the RDF-3X on one of the nodes. HadoopRDF and SparkRDF were executed in the cluster. We use the widely-used LUBM dataset with the scale of 10000, 20000 and 30000 universities, consisting of 1.3 , 2.7 and 4.1 billion triples. For the LUBM queries, we chose 7 representative queries which are roughly classified into 2 categories: highly selective queries (Q4,Q5,Q6) and unselective queries(Q1,Q2,Q3,Q7). A short description on the chosen queries is provided in the Appendix.

Table 1 summarizes our comparison with HadoopRDF and RDF-3X(best times are boldfaced). The first observation is that SparkRDF performs much better than HadoopRDF for all queries. This can be mainly attributed to the following three characteristics of SparkRDF: finer granularity of index scheme, optimal query order and effective memory-based joining. Another observation is that SparkRDF outperformed RDF-3X in Q1,Q2,Q3,Q7, while RDF-3X did better in Q4,Q5,Q6. The result conforms to our initial conjecture: RDF-3X can achieve high performance for queries with high selectivity and bound objects or subjects, while SparkRDF did well for queries with unbound objects or subjects, low selectivity or large intermediate results joins. Another result is that RDF-3X fails to answer Q1 and Q3 when the data set size is 4.1 billion triples. On the contrary, SparkRDF scales linearly and smoothly when the scale of the datasets increases from 1.3 to 4.1 billion triples. It proves that SparkRDF has a good scalability.

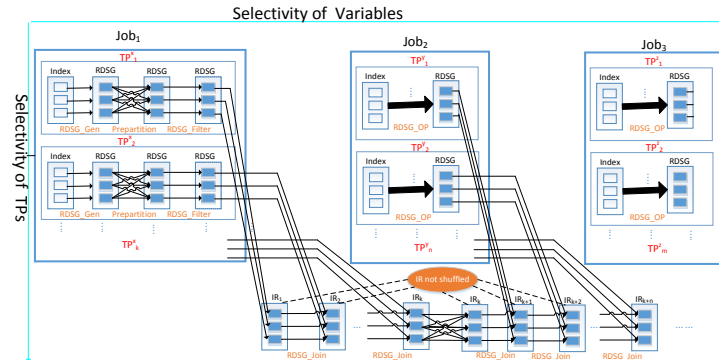


Fig. 1. The Iterative Query Model of SparkRDF

Table 1. Performance Comparison in seconds for SparkRDF(SRDF), HadoopRDF(HRDF) and RDF3X.

	LUBM-10000			LUBM-20000			LUBM-30000		
	cluster systems		centralized system	cluster systems		centralized system	cluster systems		centralized system
	SRDF	HRDF	RDF3X	SRDF	HRDF	RDF3X	SRDF	HRDF	RDF3X
Q1	478.5	8475.4	2131.4	1123.2	>3h	4380.3	1435.4	>3.5h	failed
Q2	11.9	3425.2	13.8	25.8	>2h	28.9	40.3	>2.5h	43.5
Q3	1.4	6869.7	24.6	1.4	>2.5h	90.7	1.4	>3h	failed
Q4	14.4	11940.3	0.7	23.8	>4h	0.8	32.5	>8h	0.8
Q5	6.8	2587.5	0.7	10.9	>1h	0.7	13.0	>3h	0.7
Q6	10.3	7210.5	0.6	16.4	>2.5h	0.7	20.0	>3h	0.7
Q7	54.6	1911.2	101.5	112.5	>0.7h	198.5	201.3	>1h	853.0

4 Conclusion and Future Work

In the paper, we introduce the SparkRDF, a real-time scalable big RDF graph processing engine. Also We give some the experimental results to show effectiveness of the SparkRDF. In the future, we would like to extend the work in few directions. First, we will handle more complex SPARQL patterns(such as OPTIONAL). Finally, we will make a more complete and comprehensive experiment to validate the efficiency of SparkRDF.

References

1. Atre, M., Chaoji, V., Zaki, M.J., Hendler, J.A.: Matrix Bit loaded: a scalable lightweight join query processor for rdf data. In: Proceedings of the 19th international conference on World wide web. pp. 41–50. ACM (2010)
2. Guo, Y., Pan, Z.: LUBM: A benchmark for owl knowledge base systems. Web Semantics: Science, Services and Agents on the World Wide Web 3(2), 158–182 (2005)
3. Husain, M., McGlothlin, J., Masud, M.M., Khan, L., Thuraisingham, B.: Heuristics-based query processing for large rdf graphs using cloud computing. Knowledge and Data Engineering, IEEE Transactions on 23(9), 1312–1327 (2011)
4. Neumann, T., Weikum, G.: The rdf-3x engine for scalable management of rdf data. The VLDB Journal 19(1), 91–113 (2010)
5. Zaharia, M., Chowdhury, M., Das, T., Dave, A., Ma, J., McCauley, M., Franklin, M.J., Shenker, S., Stoica, I.: Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing. In: Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation. pp. 2–2 (2012)

APPENDIX

We provide the SPARQL queries used in the experimental section:

Q1-Q6 are the same as [1]. Q7 corresponds to the Q14 of [2].

LEAPS: A Semantic Web and Linked data framework for the Algal Biomass Domain

Monika Solanki¹ and Johannes Skarka²

¹ Aston University, UK
m.solanki@aston.ac.uk

² Karlsruhe Institute of Technology, ITAS, Germany
johannes.skarka@kit.edu

Abstract. In this paper we present, *LEAPS*, a Semantic Web and Linked data framework for searching and visualising datasets from the domain of Algal biomass. *LEAPS* provides tailored interfaces to explore algal biomass datasets via REST services and a SPARQL endpoint for stakeholders in the domain of algal biomass. The rich suite of datasets include data about potential algal biomass cultivation sites, sources of CO₂, the pipelines connecting the cultivation sites to the CO₂ sources and a subset of the biological taxonomy of algae derived from the world's largest online information source on algae.

1 Motivation

Recently the idea that algae biomass based biofuels could serve as an alternative to fossil fuels has been embraced by councils across the globe. Major companies, government bodies and dedicated non-profit organisations such as ABO (Algal Biomass Organisation)³ and EABA (European Algal Biomass Association)⁴ have been pushing the case for research into clean energy sources including algae biomass based biofuels.

It is quickly evident that because of extensive research being carried out, the domain itself is a very rich source of information. Most of the knowledge is however largely buried in various formats of images, spreadsheets, proprietary data sources and grey literature that are not readily machine accessible/interpretable. A critical limitation that has been identified is the lack of a knowledge level infrastructure that is equipped with the capabilities to provide semantic grounding to the datasets for algal biomass so that they can be interlinked, shared and reused within the biomass community.

Integrating algal biomass datasets to enable knowledge representation and reasoning requires a technology infrastructure based on formalised and shared vocabularies. In this paper, we present *LEAPS*⁵, a Semantic Web/Linked data framework for the representation and visualisation of knowledge in the domain

³ <http://www.algalbiomass.org/>

⁴ <http://www.eaba-association.eu/>

⁵ <http://www.semanticwebservices.org/enalgae>

of algal biomass. One of the main goals of *LEAPS* is to enable the stakeholders of the algal biomass domain to interactively explore, via linked data, potential algal sites and sources of their consumables across NUTS (Nomenclature of Units for Territorial Statistics)⁶ regions in North-Western Europe.

Some of the objectives of *LEAPS* are,

- motivate the use of Semantic Web technologies and LOD for the algal biomass domain.
- laying out a set of ontological requirements for knowledge representation that support the publication of algal biomass data.
- elaborating on how algal biomass datasets are transformed to their corresponding RDF model representation.
- interlinking the generated RDF datasets along spatial dimensions with other datasets on the Web of data.
- visualising the linked datasets via an end user LOD REST Web service.
- visualising the scientific classification of the algae species as large network graphs.

2 *LEAPS* Datasets

The transformation of the raw datasets to linked data takes place in two steps. The first part of the data processing and the potential calculation are performed in a GIS-based model which was developed for this purpose using ArcGIS⁷ 9.3.1. The second step of lifting the data from XML to RDF is carried out using a bespoke parser that exploits XPath⁸ to selectively query the XML datasets and generate linked data using the ontologies.

The transformation process yielded four datasets which were stored in distributed triple store repositories: Biomass production sites, CO₂ sources, pipelines and region potential. We stored the datasets in separate repositories to simulate the realistic scenario of these datasets being made available by distinct and dedicated dataset providers in the future. While a linked data representation of the NUTS regions data⁹, was already available there was no SPARQL endpoint or service to query the dataset for region names. We retrieved the dataset dump and curated it in our local triple store as a separate repository. The NUTS dataset was required to link the biomass production sites and the CO₂ sources to regions where they would be located and to the dataset about the region potential of biomass yields. The transformed datasets interlinked resources defining sites, CO₂ sources, pipelines, regions and NUTS data using link predicates defined in the ontology network.

Datasets about algae cultivation can become more meaningful and useful to the biomass community, if they are integrated with datasets about algal strains.

⁶ <http://bit.ly/I7y5st>

⁷ <http://www.esri.com/software/arcgis/index.html>

⁸ <http://www.w3.org/TR/xpath/>

⁹ <http://nuts.geovocab.org/>

This can help the plant operators in taking judicious decisions about which strain to cultivate at a specific geospatial location. Algaebase¹⁰ provides the largest online database of algae information. While Algaebase does not make RDF versions of the datasets directly available through its website, they can be programmatically retrieved via their LSIDs (Life Science Identifiers) from the LSID Web resolver¹¹ made available by Biodiversity Information Standards (TDWG)¹² working group.

We retrieved RDF metadata for 113061 species of algae¹³ and curated in our triple store. We then used the Semantic import plugin with Gephi to visualise the biological taxonomy of the algae species.

3 System Description

LEAPS provides an integrated view over multiple heterogeneous datasets of potential algal sites and sources of their consumables across NUTS regions in North-Western Europe. Figure 1 illustrates the conceptual architecture of *LEAPS*. The

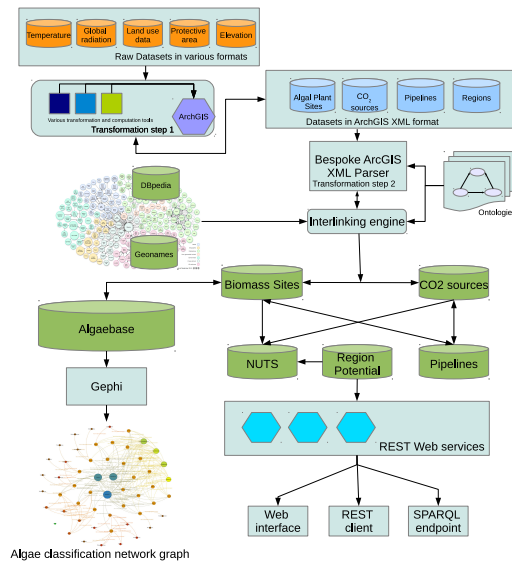


Fig. 1. Architecture of *LEAPS*

main components of the application are

¹⁰ <http://www.algaebase.org/about/>

¹¹ <http://lsid.tdwg.org/>

¹² <http://www.tdwg.org/>

¹³ The retrieval algorithm ran on an Ubuntu server for three days

- **Parsing modules:** As shown in Figure 1, the parsing modules are responsible for lifting the data from their original formats to RDF. The lifting process takes place in two stages to ensure uniformity in transformation.
- **Linking engine:** The linking engine along with the bespoke XML parser is responsible for producing the linked data representation of the datasets. The linking engine uses ontologies, dataset specific rules and heuristics to generate interlinking between the five datasets. From the LOD cloud, we currently provide outgoing links to DBpedia¹⁴ and Geonames¹⁵.
- **Triple store:** The linked datasets are stored in a triple store. We use OWLIM SE 5.0¹⁶.
- **Web services:** Several REST Web services have been implemented to provide access to the linked datasets.
- **Ontologies:** A suite of OWL ontologies for the algal biomass domain have been designed and made available.
- **Interfaces:** The Web interface provides an interactive way to explore various facets of sites, sources, pipelines, regions, ontologies and SPARQL endpoints. The map visualisation has been rendered using Google maps. Besides the SPARQL endpoint and the interactive Web interface, a REST client has been implemented for access to the datasets. Query results are available in RDF/XML, JSON, Turtle and XML formats.

4 Application access

*LEAPS*¹⁷ is available on the Web. The interface currently provides visualisation and navigation of the algae cultivation datasets in a way most intuitive for the phycologists. The application has been demonstrated to several stakeholders of the community at various algae-related workshops and congresses. They have found the navigation very useful and made suggestions for future dataset aggregation. At the time of this writing, data retrieval is relatively slow for some queries because of their federated nature, however optimisation work on the retrieval mechanism is in progress to enable faster retrieval of information.

Acknowledgments

The research described in this paper was partly supported by the Energetic Algae project (EnAlgae), a 4 year Strategic Initiative of the INTERREG IVB North West Europe Programme. It was carried out while the first author was a researcher at BCU, UK.

References

¹⁴ <http://dbpedia.org/About>

¹⁵ <http://sws.geonames.org/>

¹⁶ <http://www.ontotext.com/owlim/editions>

¹⁷ <http://www.semanticwebservices.org/enalgae>

Bridging the Semantic Gap between RDF and SPARQL using Completeness Statements

Fariz Darari, Simon Razniewski, and Werner Nutt

Faculty of Computer Science, Free University of Bozen-Bolzano, Italy
`fariz.darari@stud-inf.unibz.it, {razniewski, nutt}@inf.unibz.it`

Abstract. RDF data is often treated as incomplete, following the Open-World Assumption. On the other hand, SPARQL, the standard query language over RDF, usually follows the Closed-World Assumption, assuming RDF data to be complete. This gives rise to a semantic gap between RDF and SPARQL. In this paper, we address how to close the semantic gap between RDF and SPARQL in terms of certain answers and possible answers using completeness statements.

Keywords: SPARQL, RDF, data completeness, OWA, CWA

1 Introduction

Due to its open and incremental nature, data on the Semantic Web is generally incomplete. This can be formalized using the Open-World Assumption (OWA)[1]. SPARQL, on the other hand, interprets RDF data under closed-world semantics. While for positive queries, this does not create problems [2], SPARQL queries with negation make only sense under closed-world semantics. This ambiguous interpretation of RDF data poses the question how to determine which semantics is appropriate in a given situation.

As a particular example, suppose we want to retrieve all Oscar winners that have tattoos. For obvious reasons, no sources on the Semantic Web contain complete information about this topic and hence the Open-World Assumption applies. On the other hand, suppose we want to retrieve all Oscar winners. Here, an RDF version of IMDb¹ would contain complete data and hence one may intuitively apply closed-world reasoning.

In [3], completeness statements were introduced, which are metadata that allow one to specify that the CWA applies to parts of a data source. We argue that these statements can be used to understand the meaning of query results in terms of certain and possible answers [4], which is especially interesting for queries with negation: Suppose an RDF data source has the completeness statement “complete for all Oscar winners”. The result of a SPARQL query for Oscar winners contains not only all certain but also all possible answers. The result of a SPARQL query for Oscar winners with tattoos contains only all certain

¹ <http://www.imdb.com/oscars/>

answers. Moreover, the result of a SPARQL query for people with tattoos that did not win an Oscar contains all certain answers, but not all possible answers. The result of a SPARQL query for Oscar winners not having tattoos contains all possible answers but none of the answers is certain.

In this paper, we discuss how to assess the relationship between certain answers, possible answers and the results retrieved by SPARQL queries in the presence of completeness statements.

2 Formalization

SPARQL Queries. A basic graph pattern (BGP) is a set of triple patterns [5]. In this work, we do not consider blank nodes. We define a *graph pattern* inductively as follows: (1) a BGP P is a graph pattern; (2) for a BGP P , a NOT-EXISTS pattern $\neg P$ is a graph pattern; (3) for graph patterns P_1 and P_2 , P_1 AND P_2 is a graph pattern. Any graph pattern P can be equivalently written as a conjunction of a BGP (called the positive part and denoted as P^+) and several NOT-EXISTS patterns $\{\neg P_1, \dots, \neg P_n\}$ (referred to as the negative part and denoted as P^-). A query with negation has the form $Q = (W, P)$ where P is a graph pattern and $W \subseteq \text{var}(P^+)$ is the set of distinguished variables. The evaluation of a graph pattern P over a graph G is defined in [5]:

$$\llbracket P^+ \text{ AND } \neg P_1 \text{ AND } \dots \text{ AND } \neg P_n \rrbracket_G = \{ \mu \mid \mu \in \llbracket P^+ \rrbracket_G \wedge \forall i. \llbracket \mu(P_i) \rrbracket_G = \emptyset \}$$

The result of evaluating (W, P) over a graph G is the restriction of $\llbracket P \rrbracket_G$ to W . This fragment of SPARQL queries is safe, that is, for a query with negation Q and a graph G , the evaluation $\llbracket Q \rrbracket_G$ returns finite answers. We assume all queries to be consistent, that is, there is a graph G where $\llbracket Q \rrbracket_G \neq \emptyset$.

Completeness Statements. Formally, a *completeness statement* C is of the form $\text{Compl}(P_1|P_2)$ where P_1 , called the pattern, and P_2 , called the condition, are BGPs. Intuitively, such a statement expresses that the source contains all instantiations of P_1 that satisfy condition P_2 (e.g., all Golden Globe winners that won an Oscar).

In line with the Open-World Assumption of RDF, for a graph G , we call any graph G' such that $G' \supseteq G$ an *interpretation* of G [2]. We associate to a completeness statement C the CONSTRUCT query $Q_C = (\text{CONSTRUCT } \{ P_1 \} \{ P_1 \text{ AND } P_2 \})$. A pair (G, G') of a graph and one of its interpretations *satisfies* a completeness statement C , written $(G, G') \models C$ if $\llbracket Q_C \rrbracket_{G'} \subseteq G$ holds. It satisfies a set \mathcal{C} of completeness statements, written $(G, G') \models \mathcal{C}$ if it satisfies every element in \mathcal{C} . A set of completeness statements \mathcal{C} *entails* a statement C , written $\mathcal{C} \models C$, if for all pairs (G, G') of a graph G and an interpretation G' of G such that $(G, G') \models \mathcal{C}$, it is the case that $(G, G') \models C$.

Completeness statements restrict the set of interpretations:

Definition 1 (Valid Interpretation with Completeness Statements). *Let G be a graph and \mathcal{C} be a set of completeness statements. An interpretation $G' \supseteq G$ is valid w.r.t. \mathcal{C} iff $(G, G') \models \mathcal{C}$. We write the set of all valid interpretations of G w.r.t. \mathcal{C} as $\text{int}(G, \mathcal{C})$.*

Definition 2 (Certain and Possible Answers with Completeness Statements). Let Q be a query, G be a graph and \mathcal{C} be a set of completeness statements. Then the certain and possible answers of Q over G w.r.t. \mathcal{C} are $CA_{\mathcal{C}}^Q(G) = \bigcap_{G' \in \text{int}(G, \mathcal{C})} \llbracket Q \rrbracket_{G'}$ and $PA_{\mathcal{C}}^Q(G) = \bigcup_{G' \in \text{int}(G, \mathcal{C})} \llbracket Q \rrbracket_{G'}$, respectively.

If \mathcal{C} is empty, then $PA_{\mathcal{C}}^Q(G)$ and $CA_{\mathcal{C}}^Q(G)$ correspond to the classical certain and possible answers [4]. With completeness statements, some query results may correspond to the certain and possible answers while others may not.

Example 1 (Possible Answers). Consider again the query asking for people that have tattoos. Since this is private information we have no knowledge how complete our data source is, and hence the set of possible answers is (nearly) infinite: Anyone not explicitly stated to have tattoos might still have tattoos.

On the other hand, consider the query for people who won an Oscar. It is relatively easy for a data source to be complete on this topic, by comparing: e.g., to the Internet Movie Database (IMDb). If a data source is complete for all Oscar winners, then there are no further possible answers.

The reasoning for queries with negation is more complex. By [2], under the OWA, for a monotonic query Q and a graph G the certain answers are $\llbracket Q \rrbracket_G$, while for queries with negation, the certain answers are empty. With completeness statements, the certain answers of a query with negation can be non-empty.

Example 2 (Certain Answers). Consider first a query for people that won an Oscar but no Golden Globe. If a data source is complete both for Oscar winners and Golden Globe winners, then the query result contains all possible and all certain answers.

On the other hand, a query for people with tattoos that did not win an Oscar would only return certain answers, but not all possible answers, because there are probably many more people with tattoos that did not win an Oscar.

The result of a query for people that won an Oscar and do not have tattoos contains all possible answers but no certain answers, because we do not know for certain which Oscar winners have tattoos and which do not.

We next define for each query some completeness statements that allow one to capture the crucial information for getting certain or possible answers. Knowing about the crucial statements helps in data acquisition identify which data is needed in order to achieve desired answer semantics.

Definition 3 (Crucial Statements). For a query $Q = (W, P)$, the positive crucial statement of Q , denoted as C_Q^+ , is the statement $\text{Compl}(P^+ | \text{true})$. The set of negative crucial statements of Q , denoted as \mathcal{C}_Q^- , is the set $\{ \text{Compl}(P_1 | P^+), \dots, \text{Compl}(P_n | P^+) \}$, where P_1, \dots, P_n are from the negative part P^- of Q .

The next theorems show that the crucial statements can be used to infer relationships between certain answers, possible answers and SPARQL query results.

Theorem 1 (Bounded Possible Answers). *Let \mathcal{C} be a set of completeness statements and Q be a positive query. Then*

$$\mathcal{C} \models C_Q^+ \quad \text{implies} \quad \text{for all graphs } G, \quad PA_{\mathcal{C}}^Q(G) = CA_{\mathcal{C}}^Q(G) (= \llbracket Q \rrbracket_G).$$

While the equality $\llbracket Q \rrbracket_G = CA_{\mathcal{C}}^Q(G)$ always holds, the new insight is that $\llbracket Q \rrbracket_G = PA_{\mathcal{C}}^Q(G)$. This means the query results cannot miss any information w.r.t. reality.

Theorem 2 (Queries with Negation). *Let \mathcal{C} be a set of completeness statements and Q be a query. Then*

1. $\mathcal{C} \models C_Q^-$ implies for all graphs G , $CA_{\mathcal{C}}^Q(G) = \llbracket Q \rrbracket_G$;
2. $\mathcal{C} \models C_Q^+ \wedge C_Q^-$ implies for all graphs G , $PA_{\mathcal{C}}^Q(G) = CA_{\mathcal{C}}^Q(G) = \llbracket Q \rrbracket_G$.

The first item means that if $\mathcal{C} \models C_Q^-$, then every answer returned by the query is a certain answer. The second item means that if additionally $\mathcal{C} \models C_Q^+$, then there also cannot be any other possible answers than those returned by $\llbracket Q \rrbracket_G$. The completeness statement entailment problems can be solved using standard query containment techniques [6].

3 Discussion

We have shown that in the presence of completeness statements, the semantics of SPARQL may correspond to the certain answer or possible answer semantics. Our work is based on the observation that parts of data on the Semantic Web are actually complete. In future research, we would like to consider explicit negative RDF knowledge and completeness statements over it as an alternative for getting certain and possible answers. In this work, we assume all information contained in a graph is correct. An interesting case is when this assumption does not hold in general. We would like to investigate correctness statements as the dual of completeness statements. A further study is also needed for an effective way of maintenance of completeness statements to cope with information changes. One possible way is to add timestamps to completeness statements. An extended version with proofs of this paper is available at <http://arxiv.org/abs/1408.6395>.

References

1. Raymond Reiter. *On Closed World Data Bases*. 1977.
2. Marcelo Arenas and Jorge Pérez. Querying Semantic Web Data with SPARQL. In *PODS*, 2011.
3. Fariz Darari, Werner Nutt, Giuseppe Pirrò, and Simon Razniewski. Completeness Statements about RDF Data Sources and Their Use for Query Answering. In *ISWC 2013*, pages 66–83. Springer Berlin Heidelberg, 2013.
4. Serge Abiteboul, Paris C. Kanellakis, and Gösta Grahne. On the Representation and Querying of Sets of Possible Worlds. *Theor. Comput. Sci.*, 78(1):158–187, 1991.
5. Steve Harris and Andy Seaborne. SPARQL 1.1 Query Language. Technical report, W3C, 2013.
6. Simon Razniewski and Werner Nutt. Completeness of Queries over Incomplete Databases. *PVLDB*, 4(11):749–760, 2011.

COLINA: A Method for Ranking SPARQL Query Results through Content and Link Analysis

Azam Feyznia, Mohsen Kahani, Fattane Zarrinkalam

Computer Engineering Department, Ferdowsi University of Mashhad, Mashhad, Iran

azam.feyznia@stu-mail.um.ac.ir
kahani@um.ac.ir
fattane.zarrinkalam@stu-mail.um.ac.ir

Abstract. The growing amount of Linked Data increases the importance of semantic search engines for retrieving information. Users often examine the first few results among all returned results. Therefore, using an appropriate ranking algorithm has a great effect on user satisfaction. To the best of our knowledge, all previous methods for ranking SPARQL query results are based on popularity calculation and currently there isn't any method for calculating the relevance of results with SPARQL query. However, the proposed ranking method of this paper calculates both relevancy and popularity ranks for SPARQL query results through content and link analysis respectively. It calculates the popularity rank by generalizing PageRank method on a graph with two layers, data sources and semantic documents. It also assigns weights automatically to different semantic links. Further, the relevancy rank is based on the relevance of semantic documents with SPARQL query.

Keywords: Ranking, SPARQL, Semantic Search Engine, Link Analysis, Content Analysis, Semantic Web

1 Introduction

Structured data has enabled users to search semantic web, based on SPARQL queries. The increasing amount of structured data on the web has led to many results to be returned by a SPARQL query [1]. Further, since in most cases, all returned results equally satisfy query conditions, checking all of them and finding the best answers takes too much time. Therefore, the semantic web search engines whose have provided a SPARQL endpoint for processing and running SPARQL queries on their indexed data, require some mechanisms for ranking SPARQL query results besides the ranking methods applied to keyword queries, to help users find their desired answers in less time.

In search engines, ranking are usually done by content and link analysis and the final rank for each result is calculated by combining scores obtained from each analysis algorithm [2-3]. The content analysis ranking algorithms calculate the relevancy between each result with the user query in online mode. In the link analysis ranking algorithms,

popularity calculation is done in offline mode, before the user query is received, by constructing data graph and analyzing the existing links in it.

To the best of our knowledge, all previous methods for ranking SPARQL query results are based on popularity calculation and currently there is no method for calculating the relevance of sub-graph results with SPARQL query. The ranking methods which are based on link analysis, compute rank for entities of result graphs by utilizing entity-centric data models. It is worth noting that, the results of a *SPARQL* query in addition to the entities, may be made up of predicates and constant values. As a result, the proposed algorithms by [4] and [5] which are only based on entity ranking, cannot rank all results of *SPARQL* queries. One of the cornerstones in ranking SPARQL query results are language model based ranking methods [6]. Providing an approach for analyzing content of structured queries such as SPARQL queries, is a significant advance which is obtained by these methods.

Therefore, by studying the limitations presented in existing researches and considering specific features of SPARQL queries and results, this paper proposed a ranking method which calculates relevancy and popularity scores through content and link analysis respectively.

2 Proposed Method: COLINA

We are interested in measuring how valuable the result graph is, for a given query. Our method ranks SPARQL query results by combining content and link analysis scores of semantic documents which results are retrieved from. In the next subsections, we briefly describe two key components of our method.

2.1 Offline Ranker

The offline ranker calculates data popularity by applying weighted PageRank algorithm on data graph. We first explain our data model and then reveal our scheme for weighting semantic links.

Data Model. In order to consider the provenance of data in our link analysis ranking, we choose a two-layer graph including data source and semantic document layers. Data source layer is made up of a collection of inter-connected data sources. A data source is the source which has authority to assign URI identifier and is defined as a pay-level domain similar to [3]. The semantic document layer is composed of independent graphs of semantic documents. Each graph contains a set of internal nodes and edges.

Our explanation for using document-centric data model instead of entity-centric data model is that in response to a SPARQL query, the sub-graphs that meet query conditions are returned as results. Depending on the number of triple patterns in query, each sub-graph constitutes several triples. Hence, we can estimate the rank score of triples by the rank score of documents which are appeared in them. The document graph was constructed by extracting explicit and implicit links between semantic documents according to [7].

Weighting mechanism. We categorized links in two classes based on their labels, but not their frequency: specific and general links. In semantic web, links are semantically different and so they have different importance. Our method for measuring the importance of link labels goes beyond just measuring the frequency of labels by also taking these categories into account. We first determine which category the link label belongs to, then we use different frequency based measurements. The intuition behind this idea is that general and common link labels such as owl:sameAs, which convey high importance, get high weight. On the other hand, specific link labels, that hold much information based on information-theory, get high weight too. This way we can consider the importance of common link labels and also maintain the importance of specific link labels. In this paper, we exploit a hierarchical approach to separate the link labels that are between data sources. From this point of view, the link label that is defined for a particular class is considered general for all of its subclasses. Hence each data source is a subclass of owl:thing, we can derive general labels through extracting link labels which rdfs:domain of them is defined owl:thing by Virtuoso¹.

2.2 Online Ranker

Unlike keyword-based queries which are collection of words that are specified by users, each triple pattern in SPARQL queries has two arguments: the bound arguments which are labeled by users and the unbound arguments which are variable. We can measure the relevancy of document, based on bound and unbound query arguments as follows:

$$S_q(doc) = \beta r_q(doc) + (1 - \beta) r_r(doc) \quad (1)$$

where $r_q(doc)$ and $r_r(doc)$ denotes the relevancy score of a document with respect to unlabeled arguments in query and produced answer, respectively. Parameter β set empirically to a calibrated global value.

For example, assuming that “*Bob a Physicist*” is an answer for “*?x a Physicist*”. If this triple appears in a document which is exclusively about physicist or Bob, it is more relevant than when it is included in a document which is about anything else. This example highlights our justification for using both bound and unbound arguments in the relevance calculation for documents.

Since the computing value for $r_q(doc)$ and $r_r(doc)$ depends on query formulation, we need to deal with possible forms of triple patterns. For this, we define ACDT and QCDT functions for estimating $r_q(doc)$ and $r_r(doc)$ respectively.

The ACDT is Answer Container Document’s Triples. In short, it computes frequency of a result in semantic documents with respect to the position of unbound arguments in intended triple pattern. The QCDT is Query Container Document’s Triples. Similarly, it computes frequency of a query in semantic documents with respect to the position of bound arguments in intended triple pattern. The basic idea for ACDT and QCDT is derived from TF Scheme in information retrieval.

¹ <http://lod.openlinksw.com/sparql>

3 Combine Content and Link Analysis Ranks

We combine relevancy score S_r and popularity score S_p in order to compute final score S_f for document. Since the foundation of our ranking algorithms is similar to algorithms presented in [2], we use his method for combining scores of our algorithms.

4 Conclusion

In this paper we presented a method for ranking SPARQL query results based on content and link analysis, which can be used as ranking component in semantic web search engines. In our method, the rank of triples that constitute the result graphs are approximated by the rank score of semantic documents which expressed them. We introduced a two-layer data model and proposed a novel link weighting mechanism based on separation of link labels incorporating the notion frequency of labels in a convenient manner. Our content analysis ranking algorithm provides an approach to compute the relevancy of results with respect to the bound and unbound arguments in intended SPARQL query. We believe that using content analysis ranking in combination with link analysis ranking which is powered by our data model and weighting mechanism, can improve accuracy of ranking algorithm for SPARQL query results.

References

1. J. Hees, M. Khamis, R. Biedert, S. Abdennadher, and A. Dengel. Collecting links between entities ranked by human association strengths. *In Proceedings of ESWC-13*, pages 517-531, 2013.
2. R. Delbru. Searching Web Data: an Entity Retrieval Model. *Ph.D. Thesis, National University of Ireland*, Ireland, 2010.
3. A. Hogan, A. Harth, J. Umbrich, S. Kinsella, A. Polleres, S. Decker. Searching and Browsing Linked Data with SWSE: the Semantic Web Search Engine. *Journal of web semantics*, pages 365-401, 2011.
4. K. Mulay, P.S. Kumar. SPRING: Ranking the results of SPARQL queries on Linked Data. *17th International Conference on Management of Data COMAD*, Bangalore, India, 2011.
5. A. Buikstra, H. Neth, L. Schooler, A. ten Teije, F. van. Harmelen. Ranking query results from linked open data using a simple cognitive heuristic. *In Workshop on discovering meaning on the go in large heterogeneous data 2011 (LHD-11), Twenty-second International Joint Conference on Artificial Intelligence (IJCAI-11)*, Barcelona, Spain, 2011.
6. G. Kasneci, F. M. Suchanek, G. Ifrim, M. Ramanath, G. Weikum. NAGA: Searching and Ranking Knowledge. *In 24th International Conference on Data Engineering (ICDE 2008)*. IEEE, 2008.
7. A. Feyznia, M. Kahani, R. Ramezani. A Link Analysis Based Ranking Algorithm for Semantic Web Documents. *In 6th Conference on Information and Knowledge (IKT 2014)*, Shahrood, Iran, 2014.

Licentia: a Tool for Supporting Users in Data Licensing on the Web of Data

Cristian Cardellino¹, Serena Villata¹, Fabien Gandon¹,
Guido Governatori^{2*}, Ho-Pun Lam², and Antonino Rotolo³

¹ INRIA Sophia Antipolis, France - `firstname.lastname@inria.fr`

² NICTA Queensland Research Laboratory

`firstname.lastname@nicta.com.au`

³ University of Bologna

`antonino.rotolo@unibo.it`

Abstract. Associating a license to data is a fundamental task when publishing data on the Web. However, in many cases data producers and publishers are not legal experts, and they usually have only a basic knowledge about the possible constraints they want to ensure concerning the use and reuse of their data. In this paper, we propose a framework called Licentia that offers to the data producers and publishers a suite of services to deal with licensing information. In particular, Licentia supports, through a user-friendly interface, the users in selecting the license that better suits their needs, starting from the set of constraints proposed to regulate the terms of use and reuse of the data.

1 Introduction

In order to ensure the high quality of the data published on the Web of Data, part of the self-description of the data should consist in the licensing terms which specify the admitted use and re-use of the data by third parties. This issue is relevant both for data publication as underlined in the “Linked Data Cookbook”¹ where it is required to specify an appropriate license for the data, and for the open data publication as expressing the constraints on the reuse of the data would encourage the publication of more open data. The main problem is that data producers and publishers often do not have extensive knowledge about the existing licenses, and the legal terminology used to express the terms of data use and reuse. To address this open issue, we present Licentia, a suite of services to support data producers and publishers in data licensing by means of a user-friendly interface that masks to the user the complexity of the legal reasoning process. In particular, Licentia offers two services: *i*) the user selects among a pre-defined list those terms of use and reuse (i.e., permissions, prohibitions, and obligations) she would assign to the data and the system returns

* NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Centre of Excellence program.

¹ http://www.w3.org/2011/gld/wiki/Linked_Data_Cookbook

the set of licenses meeting (some of) the selected requirements together with the machine readable licenses' specifications, and *ii*) the user selects a license and she can verify whether a certain action² is allowed on the data released under such license. Licentia relies on the dataset of machine-readable licenses (RDF, Turtle syntax, ODRL vocabulary³ and Creative Commons vocabulary⁴) available at <http://datahub.io/dataset/rdflicense>. We rely on the deontic logic presented by Governatori et al. [2] to address the problem of verifying the compatibility of the licensing terms in order to find the license compatible with the constraints selected by the user. The need for licensing compatibility checking is high, as shown by other similar services (e.g., Licensius⁵ or Creative Commons Choose service⁶). However, the advantage of Licentia with respect to these services is twofold: first, in these services compatibility is pre-calculated among a pre-defined and small set of licenses, while in Licentia compatibility is computed at runtime and we consider more than 50 heterogeneous licenses; second, Licentia provides a further service that is not considered by the others, i.e., it allows to select a license from our dataset and verify whether some selected actions are compatible with such license.

2 Licentia: services for supporting data licensing

Licentia is implemented as a Web service⁷. It is written as a Play Framework application in Scala and Java, using the Model-View-Controller architecture, and powered with SPINdle [3] Java library as background reasoner. The architecture of the Web service is shown in Fig. 1. The workflow is defined by three steps: selection and specification of licensing conditions, reasoning and incompatibility checking, and process and return of results.

Selection and specification of conditions. Using the web interface form, the data producer and publisher specifies a set of licensing conditions she wants to associate to the data. These conditions are divided into three categories: permissions (e.g., Distribution), obligations (e.g., Attribution) and prohibitions (e.g., Commercial Use). The chosen set of conditions is taken by a server side controller, which also gets, through a SPARQL endpoint from a RDF triplestore server containing our licenses repository, a list of the stored licenses and their corresponding conditions. This data is delivered to a module that handles the information and formalizes it into defeasible logic rules for process in SPINdle⁸ – a modular and efficient reasoning engine for defeasible logic and modal defeasible

² In the interface, we adopt the terminology and the rights of the ODRL vocabulary.

³ <http://www.w3.org/ns/odrl/2/>

⁴ <http://creativecommons.org/ns>

⁵ <http://oeg-dev.dia.fi.upm.es/licensius/>

⁶ <https://creativecommons.org/choose/>

⁷ A demo video of Licentia showing the finding licenses service is available at <http://wimmics.inria.fr/projects/licentia/>.

⁸ <http://spin.nicta.org.au/spindle/index.html>

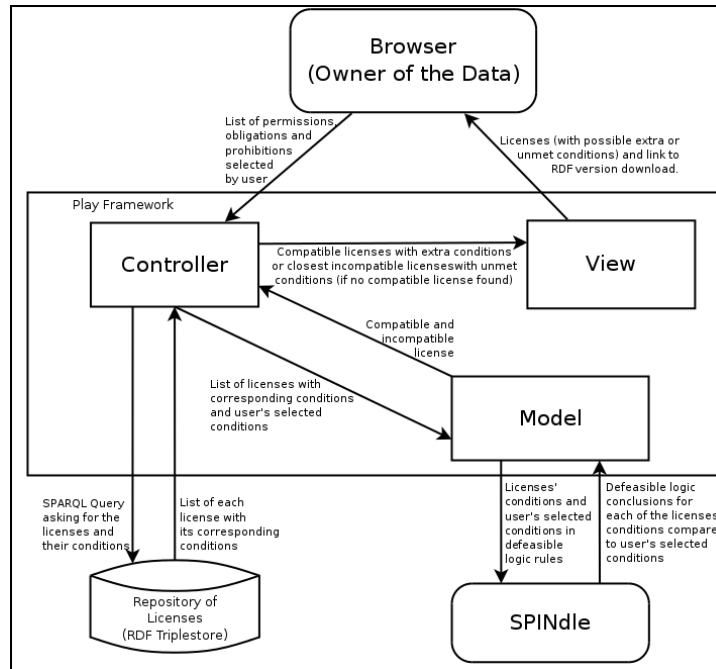


Fig. 1: Licentia Web service architecture.

logic [3]. Licentia is based on the logic and the licenses compatibility verification process proposed by Governatori et al. [2]. The module translates the licenses and the set of conditions from the RDF specification to defeasible logic rules so that SPINdle can reason over them. The module considers every single license in the repository and compares them to the list of conditions selected by the user.

Reasoning and incompatibility checking. SPINdle returns a set of conclusions for each license's conditions compared to the set of conditions selected by the user. From these conclusions, the module gets the set of incompatible conditions chosen by the user with respect to each license: all defeasible non provable rules are incompatible conditions. If this list is empty, then the license is compatible to the set of conditions the user selected. After the module gets all the conclusions for each license, it has two partial results: one containing compatible licenses, and one containing incompatible ones. If the set of compatible licenses is non empty, the module divides this set in two parts: one with those licenses containing the complete set of user's conditions, and the other with those licenses that do not contain all the user's conditions (but still are compatible), highlighting those user's conditions that are not explicitly mentioned in such license. If the set of compatible licenses is empty, the module returns the set of incompatible licenses along with a list highlighting for each license what are the user's conditions that are incompatible with the license.

Process and return of results. If a set of compatible licenses is returned, the controller provides a view listing all the licenses that are compatible and contain the user’s selected conditions on the top of the page. Secondly, it returns a list of all other compatible licenses that do not share all of the user’s conditions, in ascending order by the number of not contained conditions, highlighting each of the user’s conditions that are not explicitly defined in the license. If the set of incompatible licenses is returned, the controller filters all licenses not matching any of the conditions the user selected, keeping those licenses containing at least one of the conditions chosen by the user. If the filtered set is non empty, the system shows a message stating there is no license in the repository compatible with the selected conditions, but there exist some licenses that meet *some of* the conditions. Such licenses are thus listed in ascending order by number of incompatible conditions, highlighting every unmet condition. In any case, the list provides a link to the legal specification of the license as well as a link to a downloadable RDF version of the license. If no compatible license is found then a disclaimer is shown. Note that the evaluation of the performances of the SPINdle⁹ reasoning module to verify the compatibility of licensing terms has been presented in [2].

3 Future Perspectives

In this demo, we present the Licentia tool that proposes a set of services for supporting users in data licensing. We are currently finalizing the second service (verification of compatible actions with respect to a specific license), and we are increasing the number of considered machine readable licenses. We plan to extend Licentia by integrating an improved version of the generator of RDF licenses specifications from natural language texts introduced in [1]. Finally, a user evaluation should not be underestimated in order to improve the usability of the user interface.

References

1. Cabrio, E., Aprosio, A.P., Villata, S.: These are your rights - a natural language processing approach to automated rdf licenses generation. In: ESWC. Lecture Notes in Computer Science, vol. 8465, pp. 255–269. Springer (2014)
2. Governatori, G., Rotolo, A., Villata, S., Gandon, F.: One license to compose them all - a deontic logic approach to data licensing on the web of data. In: ISWC. Lecture Notes in Computer Science, vol. 8218, pp. 151–166. Springer (2013)
3. Lam, H.P., Governatori, G.: The making of SPINdle. In: Proceedings of RuleML, LNCS 5858. pp. 315–322. Springer (2009)
4. Maher, M.J., Rock, A., Antoniou, G., Billington, D., Miller, T.: Efficient defeasible reasoning systems. International Journal of Artificial Intelligence Tools 10, 483–501 (2001)

⁹ SPINdle has been experimentally tested against the benchmark of [4] showing that it is able to handle very large theories, indeed the largest theory it has been tested with has 1 million rules.

Automatic Stopword Generation using Contextual Semantics for Sentiment Analysis of Twitter

Hassan Saif, Miriam Fernandez and Harith Alani

Knowledge Media Institute, The Open University, United Kingdom
{h.saif, m.fernandez, h.alani}@open.ac.uk

Abstract. In this paper we propose a semantic approach to automatically identify and remove stopwords from Twitter data. Unlike most existing approaches, which rely on outdated and context-insensitive stopword lists, our proposed approach considers the contextual semantics and sentiment of words in order to measure their discrimination power. Evaluation results on 6 Twitter datasets show that, removing our semantically identified stopwords from tweets, increases the binary sentiment classification performance over the classic pre-compiled stopword list by 0.42% and 0.94% in accuracy and F-measure respectively. Also, our approach reduces the sentiment classifier’s feature space by 48.34% and the dataset sparsity by 1.17%, on average, compared to the classic method.

Keywords: Sentiment Analysis, Contextual Semantics, Stopwords, Twitter

1 Introduction

The excessive presence of abbreviations and irregular words in tweets make them very noisy, sparse and hard to extract sentiment from [7, 8]. Aiming to address this problem, existing works on Twitter sentiment analysis remove stopwords from tweets as a pre-processing procedure [5]. To this end, these works usually use pre-compiled lists of stopwords, such as the *Van stoplist* [3]. These stoplists, although widely used, have previously been criticised for: (i) being outdated [2] and, (ii) for not accounting for the specificities of the context under analysis [1]. Words with low informative values in some context or corpus, may have discrimination power in a different context. For example, the word “like”, generally considered as a stopword, has an important sentiment discrimination power in the sentence “I like you”.

In this paper, we propose an unsupervised approach for automatically generating context-aware stoplists for the sentiment analysis task on Twitter. Our approach captures the contextual semantics and sentiment of words in tweets in order to calculate their informative value. Words with low informative value are then selected as stopwords. Contextual semantics (aka statistical semantics) are based on the proposition that meaning can be extracted from words co-occurrences [9].

We evaluate our approach against the *Van stoplist* (so-called classic method) using six Twitter datasets. In particular, we study how removing stopwords generated by our approach affects: (i) the level of data sparsity of the used datasets and (ii) the performance of the Maximum Entropy (MaxEnt) classifier in terms of: (a) the size of the classifier’s feature space and, (b) the classifier’s performance. Our preliminary results show that our approach outperforms the classic stopword removal method in both accuracy and F1-measure by 0.42% and 0.94% respectively. Moreover, removing our semantically-identified stopwords reduces the feature space by 48.34% and the dataset sparsity by 1.17%, compared to the classic method, on average.

2 Stopwords Generation using Contextual Semantics

The main principle behind our approach is that the informativeness of words in sentiment analysis relies on their semantics and sentiment within the contexts they occur. Stopwords correspond to those words of weak contextual semantics and sentiment. Therefore, our approach functions by first capturing the contextual semantics and sentiment of words and then calculating their informative values accordingly.

2.1 Capturing Contextual Semantics and Sentiment

To capture the contextual semantics and sentiment of words, we use our previously proposed semantic representation model SentiCircles [6].

In summary, the SentiCircle model extracts the contextual semantics of a word from its co-occurrences with other words in a given tweet corpus. These co-occurrences are then represented as a geometric circle which is subsequently used to compute the contextual sentiment of the word by applying simple trigonometric identities on it. In particular, for each unique term m in a tweet collection, we build a two-dimensional geometric circle, where the term m is situated in the centre of the circle, and each point around it represents a context term c_i (i.e., a term that occurs with m in the same context). The position of c_i , as illustrated in Figure 1, is defined jointly by its Cartesian coordinates x_i, y_i as:

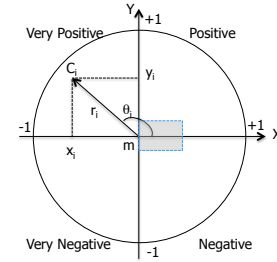


Fig. 1: SentiCircle of a term m . Stopwords region is shaded in gray.

$$x_i = r_i \cos(\theta_i * \pi) \quad y_i = r_i \sin(\theta_i * \pi)$$

Where θ_i is the polar angle of the context term c_i and its value equals to the prior sentiment of c_i in a sentiment lexicon before adaptation, r_i is the radius of c_i and its value represents the degree of correlation (tdoc) between c_i and m , and can be computed as:

$$r_i = tdoc(m, c_i) = f(c_i, m) \times \log(N/N_{c_i})$$

where $f(c_i, m)$ is the number of times c_i occurs with m in tweets, N is the total number of terms, and N_{c_i} is the total number of terms that occur with c_i . Note that all terms' radii in the SentiCircle are normalised. Also, all angles' values are in radian.

The trigonometric properties of the SentiCircle allow us to encode the contextual semantics of a term as *sentiment orientation* and *sentiment strength*. Y-axis defines the sentiment of the term, i.e., a positive y value denotes a positive sentiment and vice versa. The X-axis defines the sentiment strength of the term. The smaller the x value, the stronger the sentiment.¹ This, in turn, divides the circle into four sentiment quadrants. Terms in the two upper quadrants have a positive sentiment ($\sin \theta > 0$), with upper left quadrant representing stronger positive sentiment since it has larger angle values than those in the top right quadrant. Similarly, terms in the two lower quadrants have negative sentiment values ($\sin \theta < 0$). Moreover, a small region called the “Neutral Region” can be defined. This region is located very close to X-axis in the “Positive” and the “Negative” quadrants only, where terms lie in this region have very weak sentiment (i.e., $|\theta| \approx 0$).

¹ This is because $\cos \theta < 0$ for large angles.

The overall Contextual Semantics and Sentiment An effective way to compute the overall sentiment of m is by calculating the geometric median of all the points in its SentiCircle. Formally, for a given set of n points (p_1, p_2, \dots, p_n) in a SentiCircle Ω , the 2D geometric median g is defined as: $g = \arg \min_{g \in \mathbb{R}^2} \sum_{i=1}^n \|p_i - g\|_2$. We call the geometric median g the **SentiMedian** as its position in the SentiCircle determines the total contextual-sentiment orientation and strength of m .

2.2 Detecting Stopwords with SentiCircles

Stopwords in sentiment analysis are those who have weak semantics and sentiment within the context they occur. Hence, stopwords in our approach are those whose SentiMedians are located in the SentiCircle within a very small region close to the origin, as shown in Figure 1. This is because points in this region have: (i) very weak sentiment (i.e., $|\theta| \approx 0$) and (ii) low importance or low degree of correlation (i.e., $r \approx 0$). We call this region the *stopword region*. Therefore, to detect stopwords in our approach, we first build a SentiCircle for each word in the tweet corpus, calculate its overall Contextual semantics and sentiment by means of its SentiMedian, and check whether the word’s SentiMedian lies within the stopword region or not.

We assume the same stopword region boundary for all SentiCircles emerging from the same Twitter corpus, or context. To compute these boundaries we first build the SentiCircle of the complete corpus by merging all SentiCircles of each individual term and then we plot the density distribution of the terms within the constructed SentiCircle. The boundaries of the stopword region are delimited by an increase/decrease in the density of terms along the X- and Y-axis. Table 1 shows the X and Y boundaries of the stopword region for all Twitter datasets that we use in this work.

Dataset	OMD	HCR	STS-Gold	SemEval	WAB	GASP
X-boundary	0.0001	0.0015	0.0015	0.002	0.0006	0.0005
Y-boundary (Y)	0.0001	0.00001	0.001	0.00001	0.0001	0.001

Table 1: Stopword region boundary for all datasets

3 Evaluation and Results

To evaluate our approach, we perform binary sentiment classification (positive / negative classification of tweets) using a MaxEnt classifier and observe fluctuations (increases and decreases) after removing stopwords on: the classification performance, measured in terms of accuracy and F-measure, the size of the classifier’s feature space and the level of data sparsity. To this end, we use 6 Twitter datasets: *OMD*, *HCR*, *STS-Gold*, *SemEval*, *WAP* and *GASP* [4]. Our baseline for comparison is the *classic method*, which is based on removing stopwords obtained from the pre-compiled *Van stoplist* [3].

Figure 2 depicts the classification performance in accuracy and F1-measure as well as the reduction in the classifier’s features space obtained by applying our SentiCircle stopword removal methods on all datasets. As noted, our method outperforms the classic stopword list by 0.42% and 0.94% in accuracy and F1-measure on average respectively. Moreover, we observe that our method shrinks the feature space substantially by 48.34%, while the classic method has a reduction rate of 5.5% only.

Figure 3 shows the average impact of the SentiCircle and the classic methods on the sparsity degree of our datasets. We notice that our SentiCircle method always lowers the sparsity degree of all datasets by 1.17% on average compared to the classic method.

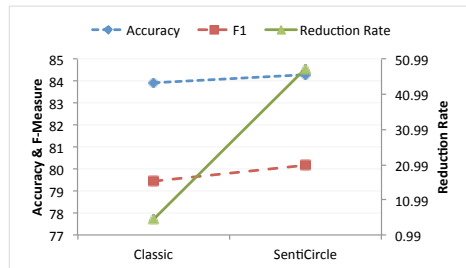


Fig. 2: Average accuracy, F-measure and reduction rate of MaxEnt using different stoplists

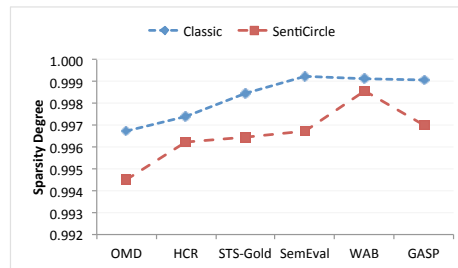


Fig. 3: Impact of the classic and SentiCircles methods on the sparsity degree of all datasets.

4 Conclusions

In this paper we proposed a novel approach for generating context-aware stopword lists for sentiment analysis on Twitter. Our approach exploits the contextual semantics of words in order to capture their context and calculates their discrimination power accordingly. We have evaluated our approach for binary sentiment classification using 6 Twitter datasets. Results show that our stopword removal approach outperforms the classic method in terms of the sentiment classification performance and the reduction in both the classifier’s feature space and the dataset sparsity.

Acknowledgment

This work was supported by the EU-FP7 project SENSE4US (grant no. 611242).

References

1. Ayral, H., Yavuz, S.: An automated domain specific stop word generation method for natural language text classification. In: International Symposium on Innovations in Intelligent Systems and Applications (INISTA) (2011)
2. Lo, R.T.W., He, B., Ounis, I.: Automatically building a stopword list for an information retrieval system. In: Journal on Digital Information Management: Special Issue on the 5th Dutch-Belgian Information Retrieval Workshop (DIR) (2005)
3. Rijsbergen, C.J.V.: Information Retrieval. Butterworth-Heinemann, Newton, MA, USA, 2nd edn. (1979)
4. Saif, H., Fernandez, M., He, Y., Alani, H.: Evaluation datasets for twitter sentiment analysis a survey and a new dataset, the sts-gold. In: Proceedings, 1st ESSEM Workshop. Turin, Italy (2013)
5. Saif, H., Fernandez, M., He, Y., Alani, H.: On Stopwords, Filtering and Data Sparsity for Sentiment Analysis of Twitter. In: Proc. 9th Language Resources and Evaluation Conference (LREC). Reykjavik, Iceland (2014)
6. Saif, H., Fernandez, M., He, Y., Alani, H.: Senticircles for contextual and conceptual semantic sentiment analysis of twitter. In: Proc. 11th Extended Semantic Web Conf. (ESWC). Crete, Greece (2014)
7. Saif, H., He, Y., Alani, H.: Alleviating data sparsity for twitter sentiment analysis. In: Proc. 2nd Workshop on Making Sense of Microposts (#MSM2012). Layon, France (2012)
8. Saif, H., He, Y., Alani, H.: Semantic sentiment analysis of twitter. In: Proceedings of the 11th international conference on The Semantic Web. Boston, MA (2012)
9. Turney, P.D., Pantel, P., et al.: From frequency to meaning: Vector space models of semantics. Journal of artificial intelligence research 37(1), 141–188 (2010)

The Manchester OWL Repository: System Description

Nicolas Matentzoglou, Daniel Tang, Bijan Parsia, and Uli Sattler

The University of Manchester
Oxford Road, Manchester, M13 9PL, UK
{matentzn,bparsia,sattler}@cs.manchester.ac.uk

Abstract. Tool development for and empirical experimentation in OWL ontology research require a wide variety of suitable ontologies as input for testing and evaluation purposes and detailed characterisations of real ontologies. Findings of surveys and results of benchmarking activities may be biased, even heavily, towards manually assembled sets of “somehow suitable” ontologies. We are building the Manchester OWL Repository, a resource for creating and sharing ontology datasets, to push the quality frontier of empirical ontology research and provide access to a great variety of well curated ontologies.

Keywords: Repository, Ontologies, Empirical

1 Introduction

Empirical work with ontologies comes in a wide variety of forms, for example surveys of the modular structure of ontologies[1], surveys of modelling patterns to inform design decisions of engineering environments [4] and benchmarking activities for reasoning services such as Description Logic (DL) classification [2]. Since it is generally difficult to obtain *representative* datasets, both due to technical reasons (lack of suitable collections) and conceptual reasons (lack of agreement on what they should be representative of), it is common practice to manually select a somewhat arbitrary set of ontologies that usually supports the given case. On top of that, few authors ever publish the datasets they used, often for practical reasons (e.g. size, effort), which makes reproducing experiment results often impossible. The currently best option for ontology related research is the BioPortal repository [5], which provides a web based interface for browsing ontologies in the biomedical domain and a REST web service to programmatically obtain copies of all (public) versions of a wide range of biomedical ontologies. There are, however, certain problems with this option. First, the repository is limited to biomedical ontologies, which makes BioPortal unsuitable for surveys that require access to ontologies of different domains. The second problem is the technical barrier of accessing the web service: It requires a good amount of work to download all interesting ontologies, for example due to a range of ontologies published in a compressed form or the logistical hurdle of recreating new snapshots over and over again. The third problem is due to the fact that there is

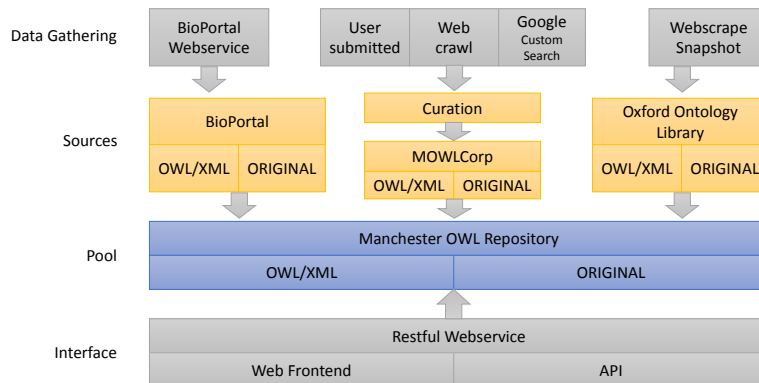


Fig. 1. The repository architecture.

no shared understanding of what it means to “use BioPortal”. Different authors have different inclusion and exclusion criteria, for example they only take the ones that are easily parseable after download, or the ones that were accessible at a particular point in time. The Manchester OWL Repository aims to bridge that gap by providing a framework for conveniently retrieving some standard datasets and allowing users to create, and share, their own.

2 Overall architecture

The Manchester OWL repository can be divided into four layers (see Figure 1). The first layer represents the data gathering. Through web crawls, web scrapes, API calls, and user contributions ontologies are collected and stored in their respective collections. The second layer represents the three main data sources of the repository, each providing ontologies in their original and curated (OWL/XML) form. The third layer, the pool, represents a virtual layer in which access to the ontologies is unified, providing some means for de-duplication because of the possibility of corpora intersection. Lastly, the interface layer provides access to the repository through a REST service and a web-based front end.

3 Data Gathering

The main component of the data gathering layer is a web crawl based on crawler4j, a java-based framework for custom web crawling and daily calls to the Google Custom Search API that fills the MOWLCorp, which makes up the bulk of the repository’s data. An ongoing BioPortal downloader creates a snapshot of BioPortal once per month using the BioPortal web services, whilst retaining copies of all available versions so far. The third (minor) component of the repository is a web scrape of the Oxford Ontology Library (OOL), a hand curated set of ontologies which features some particularly difficult, and thus in-

interesting to reasoner developers, ontologies. Ontologies are downloaded in their raw form and thrown in the curation pipeline.

4 Data curation

Ontology candidates from all three sources undergo a mild form of repair (undeclared entity injection, rewrite of non absolute IRIs) and are exported into OWL/XML, with their imports closure merged into a single ontology, while retaining information about the axiom source ontology through respective annotations. Metrics and files for both the original and the curated versions of the ontologies are retained and form part of the repository. The data curation looks slightly different for all three data sources, especially with respect to filtering. Apart from the criterion of OWL API [3] parse-ability, BioPortal and the OOL are left unfiltered because they are already deemed curated. This means that some ontologies in the corpus may not contain any logical axioms at all. In MOWLCorp, on the other hand, we filter out ontologies that 1) have an empty TBox (root ontology) and 2) have byte-identical duplicates after serialisation into OWL/XML. The reason for the first step is our focus on ontologies (which excludes pure collections of RDF instance data) and the fact that the imports closure is part of the repository, i.e., they are downloaded and evaluated independently of the root ontology.

5 Accessing the repository

There are currently three different means to access the repository: 1) A web frontend¹ provides access to preconstructed datasets and their descriptions, 2) an experimental data set creator allows users to create custom datasets based on a wide range of metrics and 3) an experimental REST-based web service that allows users to create a dataset using the REST API. Since 2) is based on 3), we now describe the query language that allows users to create their own datasets and access the web service.

The query language allows the user to construct statements that represent filter criteria for ontologies based on some essential metrics such as axiom and entity counts, or profile membership. It roughly conforms to the following grammar:

```
q = comp {("&&"|"|" ) comp}
comp = metric (">=" | "<=" | "=") n
metric = "axiom_count" | "class_count" | ...
```

where "metric" should be a valid metadata element. The query language parser was built with open-source parser generator Yacc and Lex.

The repository web services are built using the PHP framework Laravel. Laravel is an advanced framework which implements the REST protocol, so that users can get access to the services using a REST client, or simply using a web

¹ <http://mowlrepo.cs.manchester.ac.uk/>

Table 1. The REST service parameters.

service	url	method	param	return
query	/api/	POST	query	JSON array with fields: status, count, size, message, progress
check status	/api/checkStatus/	GET	id	JSON array with fields: status, progress
download	/api/resource	GET	id	file stream

browser and web-based tools such as Curl. For now, we have implemented three services: query, checkStatus and download. The query service accepts a query string that complies to the query language and returns an id string. Afterwards, users can use the id string to check the status of their query, and to download the final dataset using checkStatus and download services.

The usage of the services are listed in the Table 1; note that urls should be appended to mowlrepo.cs.manchester.ac.uk which has been omitted.

6 Next steps

We have presented the Manchester OWL Repository and a range of prototype interfaces to access pre-constructed datasets and create custom ones. We believe that the repository will help pushing the quality frontier of empirical ontology-related research by providing access to shareable, well curated datasets. We are currently working on the REST services, the dataset creator and improved dataset descriptions. In the near future, we are aiming to 1) integrate the repository with Zenodo, a service that allows hosting large datasets that are citable via DOIs, 2) extend our metadata to capture even more ontology properties (in particular consistency and coherence) and 3) improving the curation pipeline by implementing extended yet save fixes for OWL DL profile violations.

References

1. C. Del Vescovo, P. Klinov, B. Parsia, U. Sattler, T. Schneider, and D. Tsarkov. Empirical study of logic-based modules: Cheap is cheerful. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 8218 LNCS, pages 84–100, 2013.
2. R. S. Gonçalves, S. Bail, E. Jiménez-Ruiz, N. Matentzoglou, B. Parsia, B. Glimm, and Y. Kazakov. OWL Reasoner Evaluation (ORE) Workshop 2013 Results: Short Report. In *ORE*, pages 1–18, 2013.
3. M. Horridge and S. Bechhofer. The OWL API: A Java API for OWL ontologies. *Semantic Web*, 2:11–21, 2011.
4. M. Horridge, T. Tudorache, J. Vendetti, C. Nyulas, M. A. Musen, and N. F. Noy. Simplified OWL Ontology Editing for the Web: Is WebProt{g}{é} Enough? In *International Semantic Web Conference (1)*, pages 200–215, 2013.
5. N. F. Noy, N. H. Shah, P. L. Whetzel, B. Dai, M. Dorf, N. Griffith, C. Jonquet, D. L. Rubin, M. A. Storey, C. G. Chute, and M. A. Musen. BioPortal: Ontologies and integrated data resources at the click of a mouse. *Nucleic Acids Research*, 37, 2009.

A Fully Parallel Framework for Analyzing RDF Data

Long Cheng¹, Spyros Kotoulas², Tomas E Ward³, Georgios Theodoropoulos⁴

¹ Technische Universität Dresden, Germany ² IBM Research, Ireland
³ National University of Ireland Maynooth, Ireland ⁴ Durham University, UK
long.cheng@tu-dresden.de, spyros.kotoulas@ie.ibm.com
tomas.ward@nuim.ie, theogeorgios@gmail.com

Abstract. We introduce the design of a fully parallel framework for quickly analyzing large-scale RDF data over distributed architectures. We present three core operations of this framework: dictionary encoding, parallel joins and indexing processing. Preliminary experimental results on a commodity cluster show that we can load large RDF data very fast while remaining within an interactive range for query processing.

1 Introduction

Fast loading speed and query interactivity are important for exploration and analysis of RDF data at Web scale. In such scenarios, large computational resources would be tapped in a short time, which requires very fast data loading of the target dataset(s). In turn, to shorten the data processing life-cycle for each query, exploration and analysis should be done in an interactive manner. In the context of these conditions, we follow the following design paradigm.

Model. We employ the commonly used relational model. Namely, RDF data is stored in the form of triples and SPARQL queries are implemented by a sequence of lookups and joins. We do not use the graph-based approaches, because they focus on subgraph matching, which is not suitable for handling large-scale RDF data, as described in [1]. Moreover, for a row-oriented output format, graph exploration is not sufficient to generate the final join results, as presented in [2] and graph-partitioning approaches are too costly, in terms of loading speed.

Parallelism. We parallelize all operations such as dictionary encoding, indexing and query processing. Although asynchronous parallel operations (such as joins) have been seen to improve load balancing in state-of-art systems [2], we still adopt the conventional synchronous manner, since asynchronous operations always rely on specified communication protocols (*e.g.* MPI). To remedy the consequent load-imbalance problem, we focus on techniques to improve the implementations of each operation. For example, for a series of parallel joins, we keep each join operation load-balanced.

Performance. We are interested in the performance in both data loading and querying. In fact, current distributed RDF systems generally operate on a trade-off between loading complexity and query efficiency. For example, the similar-size partitioning method [3, 4] offers superior loading performance at the cost of more complex/slower querying, and the graph-based partitioning approach [2, 5] requires significant computational effort for data loading and/or partitioning. Given the trade-offs between the two

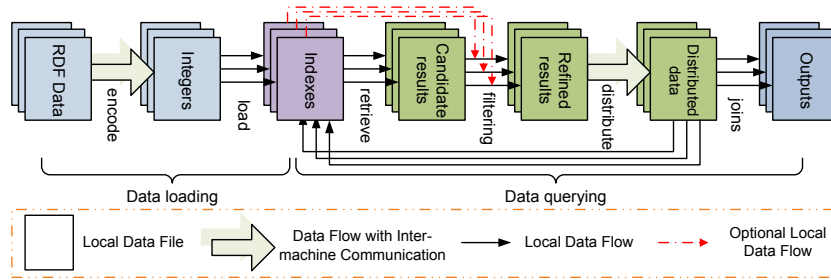


Fig. 1. General design of our parallel framework.

approaches, we combine elements of both to achieve fast loading while still keeping query time in an interactive range.

Our parallel framework is shown in Figure 1. The entire data process is divided into two parts: data loading and data querying. (1) The equal-size partitioned raw RDF data at each computation node (core) is encoded in parallel in the form of integers and then loaded in memory in local indexes (without redistributing data). (2) Based on the query execution plan, the candidate results are retrieved from the built indexes, and parallel joins are applied to formulate the final outputs. In the latter process, local filters¹ at each node can be used to reduce/remove the retrieved results that have no contribution for the final outputs, and the redistributed data during *parallel joins* can be used to create additional sharded indexes.

2 Core Operations

Triple Encoding. We utilise a distributed dictionary encoding method, as described in [6, 7], to transform RDF terms into 64-bit integers and to represent statements (aligned in memory) using this encoding. Using a straightforward technique and an efficient skew-handling strategy, our implementation [6] is shown to be notable faster than [8] and additionally supports *small incremental updates*.

Parallel Joins. Based on existing indexes, we can lookup the candidate results for each graph pattern and then use joins to compute SPARQL queries. For the most critical join operation, *parallel hash joins* [3] are commonly used in current RDF systems. However, they always bring in load-imbalance problems. The reason is that the terms in real-world Linked Data are highly skewed [9]. In comparison to that, our implementation adopts the *query-based distributed joins* we proposed in [10–13] so as to achieve more efficient and robust performance on each join operation in the presence of different query workloads.

Two-tier Indexing. We adopt an efficient two-tier index architecture we presented in [14]. We build the primary index l_1 for the encoded triples at each node using a modified *vertical partitioning* approach [15]. Different from [15], to speedup the load process, we do not do any sort operation, but just insert each tuple in a corresponding *vertical table*. For join operations, we could have to redistribute a large number of

¹ Though our system supports filtering operations, we do not give the details in this paper.

(intermediate) results around all computation nodes, which is normally very costly. To remedy this, we employ a bottom-up dynamic programming-like parallel algorithm to build a multi-level secondary index ($l_2 \dots l_n$), based on each query execution plan. With that, we will simply *copy* the redistributed data of each *join* to the local secondary indexes, and these parts of data will be re-used by other queries that contain patterns in common, so as to reduce (or remove) the corresponding network communication during the execution. In fact, according to the terminology regarding *graph partitioning* used in [5], the k -level index l_k on each node in our approach will dynamically construct a k -hop subgraph. This means that our method essentially does dynamic graph-based partitioning based on the query load, starting from an initial equal-size partitioning. Therefore, our approach can combine the loading speed of similar-size partitioning with the execution speed of graph-based partitioning in an analytical environment.

3 Preliminary Results

Experiments were conducted on 16 IBM iDataPlex[®] nodes with two 6-core Intel Xeon[®] X5679 processors, 128GB of RAM and a single 1TB SATA hard-drive, connected using Gigabit Ethernet. We use Linux kernel version 2.6.32-220 and implement our method using X10 version 2.3, compiled to C++ with gcc version 4.4.6.

Data Loading. We test the performance of *triple encoding* and *primary index building* through loading 1.1 billion triples (LUBM8000 with indexes on P, PO and PS) in memory. The entire process takes 340 secs, for an average throughput of 540MB or 3.24M triples per second (254 secs to encode triples and 86 secs to build the primary index).

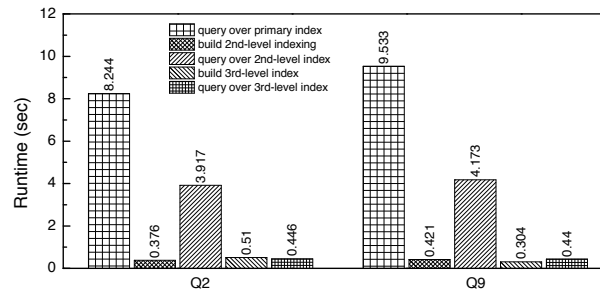


Fig. 2. Runtime over different indexes using 192 cores.

Data Querying². We implement queries over the indexes l_1 , l_2 and l_3 to examine the efficiency of our *secondary indexes*. We run the two most complex queries Q2 and Q9 of LUBM. As we do not support RDF inference, the query Q9 is modified as below so as to guarantee that we can get results for each basic graph pattern.

Q9: select ?x ?y ?z where { ?x rdf:type ub:GraduateStudent. ?y rdf:type ub:FullProfessor. ?z rdf:type ub:Course. ?x ub:advisor ?y. ?y ub:teacherOf ?z. ?x ub:takesCourse ?z. }

² The results presented here is a mirror of our previous work [14].

To focus on analyzing the core performance only, we report times for the operations of *results retrieval* and the *joins* (namely we are excluding the time to decode the output) in the execution phase.

The results in Figure 2 show that the secondary indexes can obviously improve the query performance. Moreover, the higher the level of index is, the lower the execution time. Additionally, it can be seen that building a high-level index is very fast, taking only hundreds of *ms*, which is extremely small compared to the query execution time.

We did not employ the *query-based* joins as mentioned in our query execution presented here, as we found the data skew in our tests was not obvious (due to the nature structure of the LUBM benchmark). We plan to integrate the joins with the development of our system, and then present more detailed results using much more complex workloads (*e.g.*, similar to the one used in [4]).

Acknowledgments. This work was supported by the DFG in grant KR 4381/1-1. We thank Markus Krötzsch for comments that greatly improved the manuscript.

References

1. Sun, Z., Wang, H., Wang, H., Shao, B., Li, J.: Efficient subgraph matching on billion node graphs. *Proc. VLDB Endow.* **5**(9) (May 2012) 788–799
2. Gurajada, S., Seufert, S., Miliaraki, I., Theobald, M.: TriAD: A distributed shared-nothing RDF engine based on asynchronous message passing. In: SIGMOD. (2014) 289–300
3. Weaver, J., Williams, G.T.: Scalable RDF query processing on clusters and supercomputers. In: SSWS. (2009)
4. Kotoulas, S., Urbani, J., Boncz, P., Mika, P.: Robust runtime optimization and skew-resistant execution of analytical SPARQL queries on PIG. In: ISWC. (2012) 247–262
5. Huang, J., Abadi, D.J., Ren, K.: Scalable SPARQL querying of large RDF graphs. *Proc. VLDB Endow.* **4**(11) (2011) 1123–1134
6. Cheng, L., Malik, A., Kotoulas, S., Ward, T.E., Theodoropoulos, G.: Efficient parallel dictionary encoding for RDF data. In: WebDB. (2014)
7. Cheng, L., Malik, A., Kotoulas, S., Ward, T.E., Theodoropoulos, G.: Scalable RDF data compression using X10. *arXiv preprint arXiv:1403.2404* (2014)
8. Urbani, J., Maassen, J., Drost, N., Seinstra, F., Bal, H.: Scalable RDF data compression with MapReduce. *Concurrency and Computation: Practice and Experience* **25**(1) (2013) 24–39
9. Kotoulas, S., Oren, E., Van Harmelen, F.: Mind the data skew: Distributed inferencing by speeddating in elastic regions. In: WWW. (2010) 531–540
10. Cheng, L., Kotoulas, S., Ward, T.E., Theodoropoulos, G.: QbDJ: A novel framework for handling skew in parallel join processing on distributed memory. In: HPCC. (2013) 1519–1527
11. Cheng, L., Kotoulas, S., Ward, T.E., Theodoropoulos, G.: Efficiently handling skew in outer joins on distributed systems. In: CCGrid. (2014) 295–304
12. Cheng, L., Kotoulas, S., Ward, T.E., Theodoropoulos, G.: Robust and efficient large-large table outer joins on distributed infrastructures. In: Euro-Par. (2014) 258–269
13. Cheng, L., Kotoulas, S., Ward, T.E., Theodoropoulos, G.: Robust and skew-resistant parallel joins in shared-nothing systems. In: CIKM. (2014)
14. Cheng, L., Kotoulas, S., Ward, T.E., Theodoropoulos, G.: A two-tier index architecture for fast processing large RDF data over distributed memory. In: HT. (2014) 300–302
15. Abadi, D.J., Marcus, A., Madden, S.R., Hollenbach, K.: Scalable semantic web data management using vertical partitioning. In: VLDB. (2007) 411–422

Objects as results from graph queries using an ORM and generated semantic-relational binding

Marc-Antoine Parent
maparent@acm.org

Imagination for People

Abstract. This poster describes a method to expose data in a object-relational model as linked data. It uses Virtuoso's Linked Data views on relational data to expose and query relational data. As our object-relational model is evolving, we generate the linked data view definition from augmented ORM declarations.

This project has received funding from the European Union's Seventh Framework Programme for research, technological development and demonstration under grant agreement n°6611188.

1 Interoperability in the Catalyst Consortium

The Catalyst consortium¹ has been funded by the European Commission to develop existing tools for argument mapping and argument analytics into an ecosystem of collective intelligence tools [2] using Linked Data, and test those tools on a large scale.

The existing tools (Cohere² by Open University's KMI³, Deliberatorium⁴ by MIT's Centre for Collective Intelligence⁵ and University of Zürich⁶, EdgeSense⁷ by Wikitalia⁸, and Assembl⁹ by Imagination for People¹⁰) have used a disparate array of languages (PHP, Lisp, Python) and relational databases (PostgreSQL, MySQL). More important, the data models differ in many ways: Idea-centric or message-centric, importance of data history, of social analytics, idea classification at idea creation or *post-hoc*, etc. Finally, we were all dealing with algorithmic problems that we knew could benefit from graph queries in semantic databases.

¹ <http://catalyst-fp7.eu/>

² <http://cohere.open.ac.uk/>

³ <http://kmi.open.ac.uk/>

⁴ <http://cci.mit.edu/klein/deliberatorium.html>

⁵ <http://cci.mit.edu/>

⁶ <http://www.ifi.uzh.ch/index.html>

⁷ <https://github.com/Wikitalia/edgesense>

⁸ <http://www.wikitalia.it/>

⁹ <http://assembl.org/>

¹⁰ <http://imaginationforpeople.org/fr/>

The technical partners agreed to use Linked Data technologies for interoperability, both for its inherent flexibility and because there were relevant standard ontologies that could be leveraged. We co-designed a format [6] that could accommodate the model diversity, leveraging a few common ontologies, such as SIOC, OpenAnnotation, and FOAF; and with equivalences to other relevant ontologies such as AIF. New ontologies were developed when necessary, for example for IBIS data. To lower the barrier to entry for partners with limited expertise with Semantic Web technologies, we agreed on RESTful exchange of JSON-LD data as the main interchange protocol. Partners with legacy relational models could enter the ecosystem with simple JSON parsers and generators.

The Assembl platform could not follow that simple model: our legacy model was object-oriented, using SQLAlchemy [1], a Python declarative ORM, with PostgreSQL. We wanted to use a semantic database rather than a semantic wrapper on a traditional relational database, so we could display the results of complex graph queries efficiently. On the other hand, we had a fair amount of business logic coded at the object layer, which we wanted to leverage; and the object model was under continuous development, and we did not want to maintain a semantic-relational wrapper independently.

2 Existing solutions for proxies to data storage

The first alternatives we rejected were: Pure relational (weakness of graph-oriented queries), pure semantic (relative obscurity of object-semantic tooling in Python), and partitioning our data between two databases, with the relationships in a semantic database and the content in a RDBMS (overhead of joining across database systems).

When dealing with relational data, Object-Relational Mappings (ORMs) allow developers to write an object model annotated with mappings to the relational model. This annotated object model can act as a OO wrapper, or proxy to the relational model. Many ORMs also allow developers to generate the relational model from the object model, or more rarely the object model (with relational annotations) from the relational model through relational introspection. In either case, we have a single authoritative model, which software engineers also call the “don’t repeat yourself” (DRY) principle.

With semantic data, we also have object proxies over semantic data¹¹. As with an ORM, OO code can be annotated with semantic mapping annotations, as with RDFAlchemy¹² or Elmo¹³) Also similarly, the OO code of those proxys can be generated from the RDFS or OWL models, as with RdfReactor¹⁴ or Jastor¹⁵ respectively, and others [5]. In the case of dynamic languages like Python, it is

¹¹ <http://semanticweb.org/wiki/Tripesso>

¹² <http://www.openvest.com/trac/wiki/RDFAlchemy>

¹³ <http://www.openrdf.org/elmo.jsp>

¹⁴ <http://rdfreactor.semweb4j.org/>

¹⁵ <http://jastor.sourceforge.net/>

also possible to dynamically translate accessor queries to unchecked RDF access, as with, for example, SuRF¹⁶ or OldMan¹⁷.

Another bridging technique involves semantic mapping over relational data. An adapter will use this mapping to act as a semantic proxy. The most well-known mapping language is the R2RML standard [3], but Virtuoso offers its own Linked Data Views syntax [4]. Those technologies allow great flexibility, but require to maintain the semantic-relational mapping in synchrony with the pre-existing semantic and relational models.

3 Generated semantic-relational binding for Assembl

We opted to use the Virtuoso database and enrich the relational annotation layer of SQLAlchemy with a semantic layer, with enough information to generate not only the relational model, but also the semantic mapping. This gives us both OO and semantic proxies over relational data. Simple traversals are converted by the ORM into relational queries, while more complex traversals are written as embedded SPARQL. We can expose the data as a SPARQL endpoint, or export it as JSON-LD for the benefit of the Catalyst ecosystem.

We generate Virtuoso linked data view definitions rather than R2RML mappings (which our annotations would also allow.) This allows us to also exploit Virtuoso's capability to embed a SPARQL subquery in a SQL query. Thus, our code receives ORM objects directly from a SPARQL query, and a single code-base serves as both an OO and semantic proxy to our data. We still have to keep this annotation layer up-to-date with both our relational and semantic models, as in the case of a hand-crafted semantic-relational mapping; but we avoid the maintainability cost of updating a distinct OO layer.

The poster¹⁸ contains example of data definitions in the object model, and their translation to a Virtuoso linked data binding.

Implementation Our semantic annotation layer¹⁹ is based on work by William Waites²⁰ to extend SQLAlchemy with specificities of the Virtuoso SQL dialect. An extensible introspector visits the classes known to the ORM and obtains the following information from class and columns annotations: the IRI pattern for the class; a global condition for the class to be included in the mapping; and for each database column, a specification needed to define a specific quad in the Linked data view.

SQLAlchemy has advanced capability to translate Object-Oriented (OO) inheritance into complex relational patterns²¹, so the introspector has to cater to class definitions spanning many tables, or multiple classes sharing a single table,

¹⁶ <https://pythonhosted.org/SuRF/>

¹⁷ <https://github.com/oldm/OldMan>

¹⁸ <http://maparent.ca/iswc2014poster.pdf>

¹⁹ <https://github.com/maparent/virtuoso-python>

²⁰ <http://river.styx.org/ww/2010/10/pyodbc-spasql/index>

²¹ http://docs.sqlalchemy.org/en/rel_0_9/orm/inheritance.html

and generate appropriate bindings. In `Assembl`, a subclass of the introspector also allows more quad specifications tied to more than one column, multiple graphs, global conditions that apply to class patterns, etc.

Much of the quad specification besides the predicate can be left blank, as the introspector can be initialized with a default graph, the subject is the class' subject IRI pattern, and the object is the column to which the quad specification is attached, which is interpreted to be either a literal or the application of an IRI pattern which can be inferred from foreign key information.

The quad specification may also specify a condition of applicability using the ORM constructs. The condition's structure is visited to define a coherent set of table aliases for this condition, which will be used in the linked data binding. A reference to a column defined in a superclass (which may appear in the object or condition of the quad specification) will enrich the condition with the appropriate table join; similarly, a reference to a subclass which does not define its own table will re-use the appropriate ORM condition.

4 Open issues

Having exposed our relational data as linked data, we will next work on importation of semantic data, and translating it into our relational model. We have to contend with the fact that open-world semantic data may not conform to referential integrity constraints defined at the relational layer. Also, because it is based on SQLAlchemy models, our solution follows its OO model with single inheritance.

References

1. Bayer, M.: SQLAlchemy. In: Brown, A., Wilson, G. (eds.) *The Architecture of Open Source Applications: Elegance, Evolution, and a Few More Fearless Hacks.*, vol. 2. Lulu.com (2012), <http://aosabook.org/en/sqlalchemy.html> 2
2. Buckingham Shum, S., De Liddo, A., Klein, M.: *Dcla meet cida: Collective intelligence deliberation analytics. The Second International Workshop on Discourse-Centric Learning Analytics* (Mar 2014), http://dcla14.files.wordpress.com/2014/03/dcla14_buckinghamshumdeliddoklein1.pdf 1
3. Das, S., Sundara, S., Cyganiak, R.: *R2rml: Rdb to rdf mapping language. W3c recommendation, World Wide Web Consortium* (September 2012), <http://www.w3.org/TR/2012/REC-r2rml-20120927/> 3
4. Haynes, T.: *Mapping relational data to rdf with virtuoso's rdf views. Tech. rep., OpenLink Software* (2010), <http://virtuoso.openlinksw.com/whitepapers/Mapping%20Relational%20Data%20to%20RDF.pdf> 3
5. Kalyanpur, A., Pastor, D.J., Battle, S., Padget, J.A.: *Automatic mapping of owl ontologies into java.* In: Maurer, F., Ruhe, G. (eds.) *SEKE*. pp. 98–103 (2004) 2
6. Parent, M.A., Grégoire, B.: *Software architecture and cross-platform interoperability specification. D 3.1, Catalyst-FP7* (Mar 2014), <http://catalyst-fp7.eu/wp-content/uploads/2014/03/D3.1-Software-Architecture-and-Cross-Platform-Interoperability-Specification.pdf> 2

Hedera: Scalable Indexing and Exploring Entities in Wikipedia Revision History

Tuan Tran and Tu Ngoc Nguyen

L3S Research Center / Leibniz Universität Hannover, Germany
{ttran, tunguyen}@L3S.de

Abstract. Much of work in semantic web relying on Wikipedia as the main source of knowledge often work on static snapshots of the dataset. The full history of Wikipedia revisions, while contains much more useful information, is still difficult to access due to its exceptional volume. To enable further research on this collection, we developed a tool, named *Hedera*, that efficiently extracts semantic information from Wikipedia revision history datasets. Hedera exploits Map-Reduce paradigm to achieve rapid extraction, it is able to handle one entire Wikipedia articles' revision history within a day in a medium-scale cluster, and supports flexible data structures for various kinds of semantic web study.

1 Introduction

For over decades, Wikipedia has become a backbone of semantic Web research, with the proliferation of high-quality big knowledge bases (KBs) such as DBpedia [1], where information is derived from various Wikipedia public collections. Existing approaches often rely on one offline snapshot of datasets, they treat knowledge as static and ignore the temporal evolution of information in Wikipedia. When for instance a fact changes (e.g. death of a celebrity) or entities themselves evolve, they can only be reflected in the next version of the knowledge bases (typically extracted fresh from a newer Wikipedia dump). This undesirable quality of KBs make them unable to capture temporally dynamical relationship latent among revisions of the encyclopedia (e.g., *participate* together in complex events), which are difficult to detect in one single Wikipedia snapshot. Furthermore, applications relying on obsolete facts might fail to reason under new contexts (e.g. question answering systems for recent real-world incidents), because they were not captured in the KBs. In order to complement these temporal aspects, the whole Wikipedia revision history should be well-exploited. However, such longitudinal analytics over enormous size of Wikipedia require huge computation. In this work, we develop *Hedera*, a large-scale framework that supports processing, indexing and visualising Wikipedia revision history. Hedera is an end-to-end system that works directly with the raw dataset, processes them to streaming data, and incrementally indexes and visualizes the information of entities registered in the KBs in a dynamic fashion. In contrast to existing work that handle the dataset in centralized settings [2], Hedera employs the Map-Reduce paradigm to achieve the scalable performance, which is able to transfer raw data of 2.5 year revision history of 1 million entities into full-text index

within a few hours in an 8-node cluster. We open-sourced Hedera to facilitate further research ¹.

2 Extracting and Indexing Entities

2.1 Preprocessing Dataset

Here we describe the Hedera architecture and workflow. As shown in Figure 1, the core data input of Hedera is a Wikipedia Revision history dump ². Hedera currently works with the raw XML dumps, it supports accessing and extracting information directly from compressed files. Hedera makes use heavily the Hadoop framework. The preprocessor is responsible for re-partitioning the raw files into independent units (a.k.a *InputSplit* in Hadoop) depending on users' need. There are two levels of partitioning: Entity-wise and Document-wise. Entity-wise partitioning guarantees that revisions belonging to the same entity are sent to one computing node, while document-wise sends content of revisions arbitrarily to any node, and keeps track in each revision the reference to its preceding ones for future usage in the Map-Reduce level. The preprocessor accepts user-defined low-level filters (for instance, only partition articles, or revisions within 2011 and 2012), as well as list of entity identifiers from a knowledge base to limit to. If filtered by the knowledge base, users must provide methods to verify one revision against the map of entities (for instance, using Wikipedia-derived URL of entities). The results are Hadoop file splits, in the XML or JSON formats.

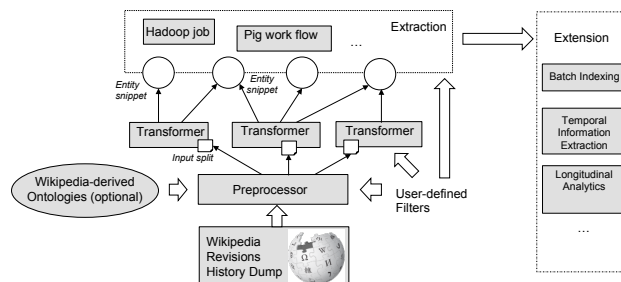


Fig. 1. Hedera Framework Architecture

2.2 Extracting Information

Before extracted in the Map-Reduce phase (Extraction component in Figure 1), file splits outputted from the preprocessor are streamed into a *Transformer*. The main goal of the transformer is to consume the files and emits $(key, value)$ pairs suitable for inputting into one Map function. Hedera provides several classes of transformer, each of which implements one operator specified in the extraction

¹ Project documentation and code can be found at: <https://github.com/antoine-tran/Hedera>

² <http://dumps.wikimedia.org>

layer. Pushing down these operators into transformers reduces significantly the volume of text sent around the network. The extraction layer enables users to write extraction logic in high-level programming languages such as Java or Pig³, which can be used in other applications. The extraction layer also accepts user-defined filters, allowing user to extract and index different portions of the same partitions at different time. For instance, the user can choose to first filter and partition Wikipedia articles published in 2012; and later she can sample, from one partition, the revisions about people published in May 2012. This flexibility facilitates rapid development of research-style prototypes in Wikipedia revision dataset, which is one of our major contributions.

3 Indexing and Exploring Entity-based Evolutions in Wikipedia

In this section, we illustrate the use of Hedera in one application - incremental indexing and visualizing Wikipedia revision history. Indexing large-scale longitudinal data collections i.e., the Wikipedia history is not a straightforward problem. Challenges in finding a scalable data structure and distributed storage that can most exploit data along the time dimension are still not fully addressed. In Hedera, we present a distributed approach in which the collection is processed and thereafter the indexing is parallelized using the Map-Reduce paradigm. This approach (that is based on the document-based data structure of ElasticSearch) can be considered as a baseline for further optimizations. The index's schema is loosely structured, which allows flexible update and incremental indexing of new revisions (that is of necessity for the evolving Wikipedia history collection). Our preliminary evaluation showed that this approach outperformed the well-known centralized indexing method provided by [2]. The time processing (indexing) gap is exponentially magnified along with the increase of data volume. In addition, we also evaluated the querying time (and experienced the similar result) of the system. We describe how the temporal index facilitate large-scale analytics on the semantic-ness of Wikipedia with some case studies. The detail of the experiment is described below.

We extract 933,837 entities registered in DBpedia, each of which correspond to one Wikipedia article. The time interval spans from 1 Jan 2011 to 13 July 2013, containing 26,067,419 revisions, amounting for 601 GBytes of text in uncompressed format. The data is processed and re-partitioned using Hedera before being passed out and indexed into ElasticSearch⁴ (a distributed real-time indexing framework that supports data at large scale) using Map-Reduce. Figure 2 illustrates one toy example of analysing the temporal dynamics of entities in Wikipedia. Here we aggregate the results for three distinct entity queries, i.e., *obama*, *euro* and *olympic* on the temporal *anchor-text* (a visible text on a hyperlink between two Wikipedia revision) index. The left-most table shows the top terms appear in the returned results, whereas the two timeline graphs illustrate the dynamic evolvement of the entities over the studied time period (with

³ <http://pig.apache.org>

⁴ <http://www.elasticsearch.org>

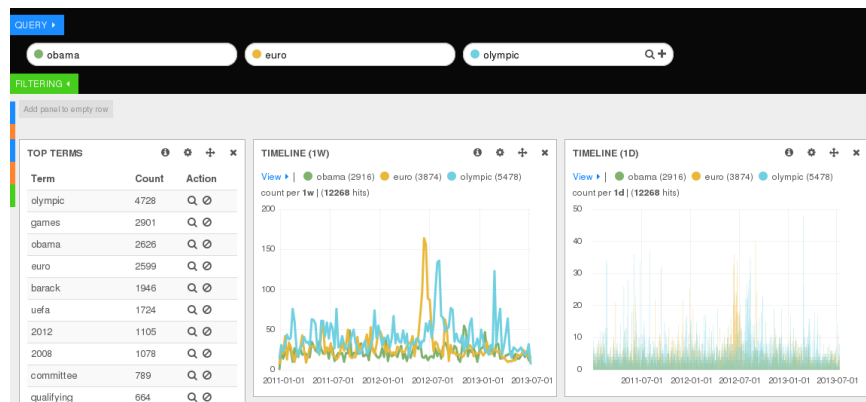


Fig. 2. Exploring Entity Structure Dynamics Over Time

1-week and 1-day granularity, from left to right respectively). As easily observed, the three entities peak at the time where a related event happens (Euro 2012 for *euro*, US Presidential Election for *obama* and the Summer and Winter Olympics for *olympic*). This further shows the value of temporal anchor text in mining the Wikipedia entity dynamics. We analogously experimented on the Wikipedia *full-text* index. Here we brought up a case study of the entity co-occurrence (or temporal relationship) (i.e., between Usain Bolt and Mo Farah), where the two co-peak in the time of Summer Olympics 2012, one big tournament where the two athletes together participated. These examples demonstrate the value of our temporal Wikipedia indexes for temporal semantic research challenges.

4 Conclusions and Future Work

In this paper, we introduced Hedera, our ongoing work in supporting flexible and efficient access to Wikipedia revision history dataset. Hedera can work directly with raw data in the low-level, it uses Map-Reduce to achieve the high-performance computation. We open-source Hedera for future use in research communities, and believe our system is the first in public of this kind. Future work includes deeper integration with knowledge bases, with more API and services to access the extraction layer more flexibly.

Acknowledgements

This work is partially funded by the FP7 project ForgetIT (under grant No. 600826) and the ERC Advanced Grant ALEXANDRIA (under grant No. 339233).

References

1. S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives. *DBpedia: A nucleus for a web of open data*. Springer, 2007.
2. O. Ferschke, T. Zesch, and I. Gurevych. Wikipedia revision toolkit: efficiently accessing wikipedia’s edit history. In *HLT*, pages 97–102, 2011.

Evaluating Ontology Alignment Systems in Query Answering Tasks

Alessandro Solimando¹, Ernesto Jiménez-Ruiz², and Christoph Pinkel³

¹ Dipartimento di Informatica, Bioingegneria, Robotica e Ingegneria dei Sistemi,
Università di Genova, Italy

² Department of Computer Science, University of Oxford, UK

³ fluid Operations AG, Walldorf, Germany

Abstract. Ontology matching receives increasing attention and gained importance in more recent applications such as ontology-based data access (OBDA). However, query answering over aligned ontologies has not been addressed by any evaluation initiative so far. A novel Ontology Alignment Evaluation Initiative (OAEI) track, Ontology Alignment for Query Answering (OA4QA), introduced in the 2014 evaluation campaign, aims at bridging this gap in the practical evaluation of matching systems w.r.t. this key usage.

1 Introduction

Ontologies play a key role in the development of the Semantic Web and are being used in many application domains such as biomedicine and energy industry. An application domain may have been modeled with different points of view and purposes. This situation usually leads to the development of different ontologies that intuitively overlap, but they use different naming and modeling conventions.

The problem of (semi-)automatically computing mappings between independently developed ontologies is usually referred to as the *ontology matching problem*. A number of sophisticated ontology matching systems have been developed in the last years [5]. Ontology matching systems, however, rely on lexical and structural heuristics and the integration of the input ontologies and the mappings may lead to many undesired logical consequences. In [1] three principles were proposed to minimize the number of potentially unintended consequences, namely: *(i) consistency principle*, the mappings should not lead to unsatisfiable classes in the integrated ontology; *(ii) locality principle*, the mappings should link entities that have similar *neighbourhoods*; *(iii) conservativity principle*, the mappings should not introduce alterations in the classification of the input ontologies. The occurrence of these violations is frequent, even in the reference mapping sets of the Ontology Alignment Evaluation Initiative⁴ (OAEI) [6].

Violations to these principles may hinder the usefulness of ontology mappings. The practical effect of these violations, however, is clearly evident when ontology alignments are involved in complex tasks such as query answering [4].

⁴ <http://oaei.ontologymatching.org/>

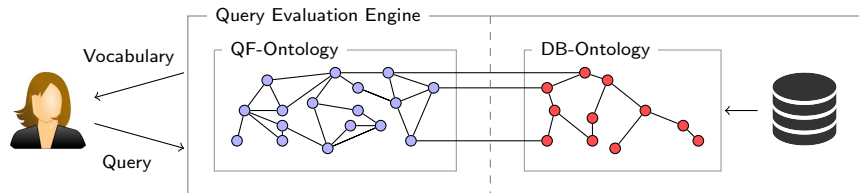


Fig. 1. Ontology Alignment in an OBDA Scenario

The traditional tracks of *OAEI* evaluate ontology matching systems w.r.t. scalability, multi-lingual support, instance matching, reuse of background knowledge, etc. Systems' effectiveness is, however, only assessed by means of classical information retrieval metrics (*i.e.*, precision, recall and f-measure) w.r.t. a manually-curated reference alignment, provided by the organisers. The new *OA4QA* track⁵ evaluates those same metrics, but w.r.t. the ability of the generated alignments to enable the answer of a set of queries in an OBDA scenario, where several ontologies exist. Figure 1 shows an OBDA scenario where the first ontology provides the vocabulary to formulate the queries (QF-Ontology) and the second is linked to the data and it is not visible to the users (DB-Ontology). Such OBDA scenario is presented in real-world use cases (*e.g.*, Optique project⁶ [2, 6]). The integration via ontology alignment is required since only the vocabulary of the DB-Ontology is connected to the data. The *OA4QA* will also be key for investigating the effects of logical violations affecting the computed alignments, and evaluating the effectiveness of the repair strategies employed by the matchers.

2 Ontology Alignment for Query Answering

This section describes the considered dataset and its extensions (Section 2.1), the query processing engine (Section 2.2), and the evaluation metrics (Section 2.3).

2.1 Dataset

The set of ontologies coincides with that of the *conference* track,⁷ in order to facilitate the understanding of the queries and query results. The dataset is however extended with synthetic ABoxes, extracted from the *DBLP* dataset.⁸

Given a query q expressed using the vocabulary of ontology \mathcal{O}_1 , another ontology \mathcal{O}_2 enriched with synthetic data is chosen. Finally, the query is executed over the aligned ontology $\mathcal{O}_1 \cup \mathcal{M} \cup \mathcal{O}_2$, where \mathcal{M} is an alignment between \mathcal{O}_1 and \mathcal{O}_2 . Referring to Figure 1, \mathcal{O}_1 plays the role of QF-Ontology, while \mathcal{O}_2 that of DB-Ontology.

⁵ <http://www.cs.ox.ac.uk/isg/projects/Optique/oa4qa/>

⁶ <http://www.optique-project.eu/>

⁷ <http://oa4qa.ontologymatching.org/2014/conference/index.html>

⁸ <http://dblp.uni-trier.de/xml/>

2.2 Query Evaluation Engine

The evaluation engine considered is an extension of the OWL 2 reasoner *HermiT*, known as *OWL-BGP*⁹ [3]. OWL-BGP is able to process SPARQL queries in the SPARQL-OWL fragment, under the *OWL 2 Direct Semantics entailment regime*.¹⁰ The queries employed in the *OA4QA* track are standard conjunctive queries, that are fully supported by the more expressive SPARQL-OWL fragment. SPARQL-OWL, for instance, also support queries where variables occur within complex class expressions or bind to class or property names.

2.3 Evaluation Metrics and Gold Standard

As already discussed in Section 1, the evaluation metrics used for the *OA4QA* track are the classic information retrieval ones (*i.e.*, precision, recall and f-measure), but on the result set of the query evaluation. In order to compute the gold standard for query results, the publicly available reference alignments *ra1* has been manually revised. The aforementioned metrics are then evaluated, for each alignment computed by the different matching tools, against the *ra1*, and manually repaired version of *ra1* from conservativity and consistency violations.

Three categories of queries will be considered in *OA4QA*: (*i*) basic, (*ii*) queries involving violations, (*iii*) advanced queries involving nontrivial mappings.

2.4 Impact of the Mappings in the Query Results

As an illustrative example, consider the aligned ontology \mathcal{O}_U computed using *confof* and *ekaw* as input ontologies (\mathcal{O}_{confof} and \mathcal{O}_{ekaw} , respectively), and the *ra1* reference alignment between them. \mathcal{O}_U entails $ekaw:Student \sqsubseteq ekaw:Conf_Participant$, while \mathcal{O}_{ekaw} does not, and therefore this represents a conservativity principle violation. Clearly, the result set for the query $q(x) \leftarrow ekaw:Conf_Participant(x)$ will erroneously contain any student not actually participating at the conference. The explanation for this entailment in \mathcal{O}_U is given below, where Axioms 1 and 3 are mappings from the reference alignment.

$$confof:Scholar \equiv ekaw:Student \quad (1)$$

$$confof:Scholar \sqsubseteq confof:Participant \quad (2)$$

$$confof:Participant \equiv ekaw:Conf_Participant \quad (3)$$

The softening of Axiom 3 into $confof:Participant \sqsupseteq ekaw:Conf_Participant$ represents a possible repair for the aforementioned violation.

3 Preliminary Evaluation

In Table 1¹¹ a preliminary evaluation using the alignments of the *OAEI 2013* participants and the following queries is shown: (*i*) $q_1(x) \leftarrow ekaw:Author(x)$,

⁹ <https://code.google.com/p/owl-bgp/>

¹⁰ <http://www.w3.org/TR/2010/WD-sparql11-entailment-20100126/#id45013>

¹¹ $\#q(x)$ refers to the cardinality of the result set.

Category	Query	# \mathcal{M}	Reference Alignment				Repaired Alignment			
			#q(x)	Prec.	Rec.	F-meas.	#q(x)	Prec.	Rec.	F-meas.
Basic	q_1	5	98	1	1	1	98	1	1	1
Violations	q_2	4	53	0.8	1	0.83	38	0.57	1	0.68
Advanced	q_3	7	-	-	-	-	182	1	0.5	0.67

Table 1. Preliminary query answering results for the OAEI 2013 alignments

over the ontology pair $\langle cmt, ekaw \rangle$; (ii) $q_2(x) \leftarrow ekaw:Conf_Participant(x)$, over $\langle confof, ekaw \rangle$, involving the violation described in Section 2.4; (iii) and $q_3(x) \leftarrow confof:Reception(x) \cup confof:Banquet(x) \cup confof:Trip(x)$, over $\langle confof, edas \rangle$. The evaluation¹² shows the negative effect on precision of logical flaws affecting the computed alignments (q_2) and a lowering in recall due to missing mapping (q_3). For q_3 the results w.r.t. the reference alignment ($ra1$) are missing due to the unsatisfiability of the aligned ontology $\mathcal{O}_{confof} \cup \mathcal{O}_{edas} \cup ra1$.

4 Conclusions and Future Work

We have presented the novel OAEI track addressing query answering over pairs of ontologies aligned by a set of ontology-to-ontology mappings. From the preliminary evaluation the main limits of the traditional evaluation, for what concerns logical violations of the alignments, clearly emerged. As a future work we plan to cover increasingly complex queries and ontologies, including the ones in the Optique use case [6]. We also plan to consider more complex scenarios involving a single QF-Ontology aligned with several DB-Ontologies.

Acknowledgements. This work was supported by the EU FP7 IP project Optique (no. 318338), the MIUR project CINA (Compositionality, Interaction, Negotiation, Autonomicity for the future ICT society) and the EPSRC project Score!.

References

1. Jiménez-Ruiz, E., Cuenca Grau, B., Horrocks, I., Berlanga, R.: Logic-based Assessment of the Compatibility of UMLS Ontology Sources. *J. Biomed. Semant.* (2011)
2. Kharlamov, E., et al.: Optique 1.0: Semantic Access to Big Data: The Case of Norwegian Petroleum Directorate’s FactPages. *ISWC (Posters & Demos)* (2013)
3. Kollia, I., Glimm, B., Horrocks, I.: SPARQL query answering over OWL ontologies. In: *The Semantic Web: Research and Applications*, pp. 382–396. Springer (2011)
4. Meilicke, C.: Alignments Incoherency in Ontology Matching. Ph.D. thesis, University of Mannheim (2011)
5. Shvaiko, P., Euzenat, J.: Ontology Matching: State of the Art and Future Challenges. *IEEE Transactions on Knowl. and Data Eng. (TKDE)* (2012)
6. Solimando, A., Jiménez-Ruiz, E., Guerrini, G.: Detecting and Correcting Conservativity Principle Violations in Ontology-to-Ontology Mappings. In: *International Semantic Web Conference* (2014)

¹² Out of the 26 alignments of OAEI 2013, only the ones shown in column # \mathcal{M} were able to produce a result (either for logical problems or for an empty result set due to missing mappings). Reported precision/recall values are averaged values.

Using Fuzzy Logic For Multi-Domain Sentiment Analysis

Mauro Dragoni¹, Andrea G.B. Tettamanzi², and Célia da Costa Pereira²

¹ FBK-IRST, Trento, Italy

² Université Nice Sophia Antipolis, I3S, UMR 7271, Sophia Antipolis, France
dragoni@fbk.eu andrea.tettamanzi|celia.pereira@unice.fr

Abstract. Recent advances in the Sentiment Analysis field focus on the investigation about the polarities that concepts describing the same sentiment have when they are used in different domains. In this paper, we investigated on the use of fuzzy logic representation for modeling knowledge concerning the relationships between sentiment concepts and different domains. The developed system is built on top of a knowledge base defined by integrating WordNet and SenticNet, and it implements an algorithm used for learning the use of sentiment concepts from multi-domain datasets and for propagating such information to each concept of the knowledge base. The system has been validated on the Blitzer dataset, a multi-domain sentiment dataset built by using reviews of Amazon products, by demonstrating the effectiveness of the proposed approach.

1 Introduction

Sentiment Analysis is a kind of text categorization task that aims to classify documents according to their opinion (polarity) on a given subject [1]. This task has created a considerable interest due to its wide applications. However, in the classic Sentiment Analysis the polarity of each term of the document is computed independently with respect to domain which the document belongs to. Recently, the idea of adapting terms polarity to different domains emerged [2]. The rationale behind the idea of such investigation is simple. Let's consider the following example concerning the adjective "small":

1. The sidebar is **small** and it is not able to contain a lot of stuff.
2. The **small** dimensions of this decoder allow to move it easily.

In the first text, we considered the Furnishings domain and, within it, the polarity of the adjective "small" is, for sure, "negative" because it highlights an issue of the described item. On the other side, in the second text, where we considered the Electronics domain, the polarity of such adjective can be considered "positive".

In literature, different approaches related to the Multi-Domain Sentiment Analysis have been proposed. Briefly, two main categories may be identified: (i) the transfer of learned classifiers across different domains [3] [4], and (ii) the use of propagation of labels through graphs structures [5] [6]. Independently from the kind of approach, works using concepts rather than terms for representing different sentiments have been proposed.

Differently from the approaches already discussed in the literature, we address the multi-domain sentiment analysis problem by applying the fuzzy logic theory for modeling membership functions representing the relationships between concepts and domains. Moreover, the proposed system exploits the use of semantic background knowledge for propagating information represented by the learned fuzzy membership functions to each element of the network.

2 System

The main aim of the implemented system is the learning of fuzzy membership functions representing the belonging of a concept with respect to a domain in terms of both sentiment polarity as well as aboutness. The two pillars on which the system has been thought are: (i) the use of fuzzy logic for modeling the polarity of a concept with respect to a domain as well as its aboutness, and (ii) the creation of a two-levels graph where the top level represents the semantic relationships between concepts, while the bottom level contains the links between all concept membership functions and the domains.

Figure 1 shows the conceptualization of the two-levels graph. Relationships between the concepts of the Level 1 (the Semantic Level) are described by the background knowledge exploited by the system. The type of relationships are the same generally used in linguistic resource: for example, concepts C_1 and C_3 may be connected through an Is-A relationship rather than the Antonym one. Instead, each connection of the Level 2 (the Sentiment Level) describes the belonging of each concept with respect to the different domains taken into account.

The system has been trained by using the Blitzer dataset³ in two steps: first, the fuzzy membership functions have been initially estimated by analyzing only the explicit information present within the dataset (Section 2.1); then, (ii) the explicit information has been propagated through the Sentiment Level graph by exploiting the connections defined in the Semantic Level.

2.1 Preliminary Learning Phase

The Preliminary Learning (PL) phase aims to estimate the starting polarity of each concept with respect to a domain. The estimation of this value is done by analyzing only the explicit information provided by the training set. This phase allows to define the preliminary fuzzy membership functions between the concepts defined in the Semantic Level of the graph and the domains that are defined in the Sentiment one. Such a value is computed by the Equation 1

$$\text{polarity}_i^*(C) = \frac{k_C^i}{T_C^i} \in [-1, 1] \quad \forall i = 1, \dots, n, \quad (1)$$

where C is the concept taken into account, index i refers to domain D_i which the concept belongs to, n is the number of domains available in the training set, k_C^i is the arithmetic sum of the polarities observed for concept C in the training set restricted to

³ <http://www.cs.jhu.edu/~mdredze/datasets/sentiment/>

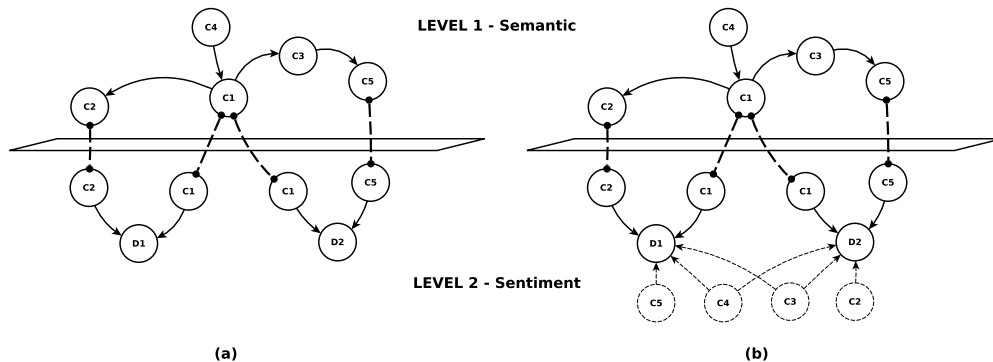


Fig. 1: The two-layer graph initialized during the Preliminary Learning Phase (a) and its evolution after the execution of the Information Propagation Phase (b).

domain D_i , and T_C^i is the number of instances of the training set, restricted to domain D_i , in which concept C occurs. The shape of the fuzzy membership function generated during this phase is a triangle with the top vertex in the coordinates $(x, 1)$, where $x = \text{polarity}_i^*(C)$ and with the two bottom vertices in the coordinates $(-1, 0)$ and $(1, 0)$ respectively. The rationale is that while we have one point (x) in which we have full confidence, our uncertainty covers the entire space because we do not have any information concerning the remaining polarity values.

2.2 Information Propagation Phase

The Information Propagation (IP) phase aims to exploit the explicit information learned in the PL phase in order to both (i) refine the fuzzy membership function of the known concepts, as well as, (ii) to model such functions for concepts that are not specified in the training set, but that are semantically related to the specified ones. Figure 1 presents how the two-levels graph evolves before and after the execution of the IP phase. After the PL phase only four membership functions are modeled: C_1 and C_2 for the domain D_1 , and C_1 and C_5 for the domain D_2 (Figure 1a). However, as we may observe, in the Semantic Level there are concepts that are semantically related to the ones that were explicitly defined in the training set, namely C_3 and C_4 ; while, there are also concepts for which a fuzzy membership function has not been modeled for some domains (i.e. C_2 for the domain D_2 and C_5 for the domain D_1).

Such fuzzy membership functions may be inferred by propagating the information modeled in the PL phase. Similarly, existing fuzzy membership functions are refined by the influence of the other ones. Let's consider the polarity between the concept C_3 and the domain D_2 . The fuzzy membership function representing this polarity is strongly influenced by the ones representing the polarities of concepts C_1 and C_5 with respect to the domain D_2 .

The propagation of the learned information through the graph is done iteratively where, in each iteration, the estimated polarity value of the concept x learned during the PL phase is updated based on the learned values of the adjoining concepts. At each

iteration, the updated values is saved in order to exploit it for the re-shaping of the fuzzy membership function associating the concept x to the domain i .

The resulting shapes of the inferred fuzzy membership functions will be trapezoids where the extension of the upper base is proportional to the difference between the value learned during the PL phase (V_{pi}) and the value obtained at the end of the IP phase (V_{ip}); while, the support is proportional to both the number of iterations needed by the concept x to converge to the V_{ip} and the variance with respect to the average of the values computed after each iteration of the IP phase.

3 Concluding Remarks

The system have been validated on the full version of the Blitzer dataset⁴ and the results, compared with the precision obtained by three baselines, are shown in Table 1.

SVM [7] Precision (Rec. 1.0)	Naive-Bayes [8] Precision (Rec. 1.0)	Max-Entropy [8] Precision (Rec. 1.0)	MDFSA Precision	MDFSA Recall
0.8068	0.8227	0.8275	0.8617	0.9987

Table 1: Results obtained on the full version of the Blitzer dataset.

The results demonstrated that the modeled fuzzy membership functions may be exploited effectively for computing the polarities of concepts used in different domains.

References

1. Pang, B., Lee, L., Vaithyanathan, S.: Thumbs up? sentiment classification using machine learning techniques. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), Philadelphia, Association for Computational Linguistics (July 2002) 79–86
2. Blitzer, J., Dredze, M., Pereira, F.: Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In Carroll, J.A., van den Bosch, A., Zaenen, A., eds.: ACL, The Association for Computational Linguistics (2007)
3. Bollegala, D., Weir, D.J., Carroll, J.A.: Cross-domain sentiment classification using a sentiment sensitive thesaurus. IEEE Trans. Knowl. Data Eng. **25**(8) (2013) 1719–1731
4. Xia, R., Zong, C., Hu, X., Cambria, E.: Feature ensemble plus sample selection: Domain adaptation for sentiment classification. IEEE Int. Systems **28**(3) (2013) 10–18
5. Ponomareva, N., Thelwall, M.: Semi-supervised vs. cross-domain graphs for sentiment analysis. In Angelova, G., Bontcheva, K., Mitkov, R., eds.: RANLP, RANLP 2011 Organising Committee / ACL (2013) 571–578
6. Tsai, A.C.R., Wu, C.E., Tsai, R.T.H., jen Hsu, J.Y.: Building a concept-level sentiment dictionary based on commonsense knowledge. IEEE Int. Systems **28**(2) (2013) 22–30
7. Chang, C.C., Lin, C.J.: Libsvm: A library for support vector machines. ACM TIST **2**(3) (2011) 27
8. McCallum, A.K.: Mallet: A machine learning for language toolkit. <http://mallet.cs.umass.edu> (2002)

⁴ Detailed results and tool demo are available at http://dkmtools.fbk.eu/moki/demo/mdfsa/mdfsa_demo.html

AMSL

Creating a Linked Data Infrastructure for Managing Electronic Resources in Libraries

Natanael Arndt¹, Sebastian Nuck¹, Andreas Nareike^{1,2}, Norman Radtke^{1,2},
Leander Seige², and Thomas Riechert³

¹ AKSW, Institute for Applied Informatics (InfAI) e.V.
Hainstraße 11, 04109 Leipzig, Germany
`{lastname}@infai.org`

² Leipzig University Library
Beethovenstr. 6, 04107 Leipzig, Germany
`{lastname}@ub.uni-leipzig.de`

³ Leipzig University of Applied Science (HTWK)
Gustav-Freytag-Str. 42A, 04251 Leipzig, Germany
`thomas.riechert@htwk-leipzig.de`

1 Introduction

The library domain is currently undergoing changes relating to not only physical resources (e.g. books, journals, CDs/DVDs) but to the rapidly growing numbers of electronic resources (e.g. e-journals, e-books or databases) which are contained in their collections as well. Since those resources are not limited to physical items anymore and can be copied without loss of information, new licensing and lending models have been introduced. Current licensing models are pay-per-view, patron-driven-acquisition, short term loan and big deal. In addition, with changing markets, user expectations and publication formats, new models will be introduced in the future.

The existing infrastructure is not prepared for managing those new electronic resources, licensing and lending models. Software which is developed to meet the current requirements (cf. section 2) is likely to be outdated in a couple of years, due to the changing modeling requirements. To develop a future-proof managing system, a highly flexible data model and software which is adaptable to a changed data model is needed. The targeted user group are librarians with limited or no understanding of Semantic Web, Linked Data and RDF techniques.

We present AMSL [1], an *Electronic Resource Management System* which is based on the generic and collaborative RDF resource management system OntoWiki [2,3]. Using RDF as data model makes the application flexible and thus future-proof, while the changing modeling requirements can be met by adjusting the used RDF vocabulary resp. application profile. To adapt the generic OntoWiki to the needs of the domain experts, we have added some components to support domain specific use cases, while still keeping them agnostic to

the used RDF vocabulary. The current status of the project including links to the source code and a live demonstration is available at the project web-page: <http://amsl.technology/>.

2 State of the Art

Software currently in use in libraries is mainly concerned with managing print resources, such as the *Integrated Library System* LIBERO (<http://www.libero.com.au/>), which is used at Leipzig University Library. But such software is not prepared for managing electronic resources with complex licensing and access models. Recently, some commercial vendors have started providing *Next Generation Library Systems* which constitute a new approach to Electronic Resource Management. Due to their commercial nature, it is difficult to evaluate how flexible the used data models are and which requirements are met [4, 5]. Since the data models are closed, they can not be extended by the customer as requirements change, whereas the contract and subscription terms often change with every new business year. Further, it is more difficult to integrate external knowledge bases in contrast to Linked Data and the Linked Open Data Cloud [6].

Another approach is to avoid specific library software but to use generic resource or document management systems, such as Wikis. OntoWiki was developed as a Wiki system for collaboratively creating semantic data in RDF and OWL knowledge bases. Over the time, OntoWiki has evolved towards a highly extensible development framework for knowledge intensive applications [3].

3 The AMSL Electronic Resource Management System

The AMSL Electronic Resource Management System is based on the OntoWiki Application Framework. It provides the basic functionality for managing resources i.e. creating, editing, querying, visualizing, linking and exporting RDF/OWL knowledge bases. With the Linked Data Server and Linked Data Wrapper/Importer components, it can publish and consume Linked Data according to the rules [6]. An Application Programming Interface allows the development of powerful third party extensions.

A core part of our application is to use an expressive **data model** as application profile which comprises different RDF and OWL vocabularies. The expressiveness of the data model helps to move design decisions from the program code to the easily adoptable vocabulary definitions, which increases the flexibility of the whole system. For expressing the data model we combine well known existing vocabularies, such as DCMI Metadata Terms (<http://purl.org/dc/terms/>), Dublin Core Metadata Element Set (<http://purl.org/dc/elements/1.1/>), The Bibliographic Ontology (<http://purl.org/ontology/bibo/>), Academic Institution Internal Structure Ontology (<http://purl.org/vocab/aiiso/schema#>) and The Friend of a Friend RDF vocabulary (<http://xmlns.com/foaf/0.1/>), to keep the data compatible to already existing components. With AMSL, we further introduce the *Vocabulary for Library ERM*

(BIBRM, <http://vocab.ub.uni-leipzig.de/bibrm/>). It provides terms for expressing licensing and access models and is aligned to the ideas of the Electronic Resource Management Initiative (ERMI) [4]. If new requirements arise in the future, the data model has to be changed and if necessary new vocabulary terms can be added.

We have developed **data templates** to provide a user interface for resource creation to the domain experts. Since our system does not require technical RDF knowledge of its users, the data templates provide a form based editing interface, which further supports the created resources to be compliant to the defined application profile. The template definition itself is expressed in RDF as well, to achieve the required extensibility, without need to change the software.

To support the work-flows for managing meta-data coming with electronic resources (such as contacts, contracts, packages, agreements and licenses), special **import and integration components** are developed. Publicly available Linked Data and SPARQL services, which have evolved in the library domain in the past years (e.g. title information, ISSN-history and authority files), are necessary for the electronic resource management and are imported using the existing Linked Data import process.

Being able to restore changes made on triples is one of the existing features of OntoWiki. In the AMSL project further requirements for reproducibility, consistency and increased clarity of resource changes were formulated. To meet these requirements the present **versioning** system was extended towards a ChangeSet ontology⁴. The versioning metadata is stored in a SQL database containing the ChangeSet elements. Hence, the underlying data model is now capable of expressing the same amount of information as a ChangeSet. Additions and removals concerning the same triple are aggregated to a change statement. Even though the versioning metadata is not stored in form of triples, the information could be queried and transferred into a ChangeSet knowledge base for further purposes. Moreover multiple retrieval capabilities for querying extended versioning information have been added in form of an OntoWiki extension.

The present search function of the OntoWiki is based on a conventional SPARQL search, using a `bif:contains` filter on labels. To improve search speed and provide additional features like a fuzzy search, the Elastic Search search engine was integrated as an OntoWiki extension. This **full-text search** makes use of a class based index structure. Hence, it provides a faster access to pre-indexed resources. The underlying index structure is built up by indexing classes of the knowledge base which contains information that need to be accessed frequently. That is, classes containing properties such as `dc:title` are more meaningful to be searched with a full-text search than classes that only include properties like e.g. `bibo:issn` which contain numerical sequences. Additionally to the auto-completion features search function, an extended search supports an enhanced fuzzy search which provides the possibility of restricting the result set to the previously defined classes and is more robust against typing mistakes.

⁴ <http://vocab.org/changeset/schema.html>

4 Conclusion and Future Work

We demonstrate a novel approach to build up an electronic resource management system for libraries by using generic RDF resource management technology. The used generic components are extended and complemented by some domain adaptable components to provide a customized interface to domain experts. Consequently using Linked Data and RDF further enables and supports libraries to build up a Linked Data infrastructure for exchange of meta data across libraries as well as across institutions using the services provided by library.

5 Acknowledgments

The presented software system was developed in the AMSL project for developing an Electronic Resource Management System based on Linked Data technology (<http://amsl.technology/>). We want to thank our colleagues Lydia Unterdörfel, Carsten Krahl and Björn Muschall from Leipzig University Library, Jens Mittelbach from Saxon State and University Library Dresden (SLUB) and our fellows from Agile Knowledge Engineering and Semantic Web (AKSW) research group especially Henri Knochenhauer for their support, helpful comments and inspiring discussions. This work was supported by the European Union and Free State of Saxony by a grant from the European Regional Development Fund (ERDF) for the project number 100151134 (SAB index).

References

1. Nareike, A., Arndt, N., Radtke, N., Nuck, S., Seige, L., Riechert, T.: Amsl – managing electronic resources for libraries based on semantic web. In: Workshop on Data Management and Electronic Resource Management in Libraries (DERM 2014) : INFORMATIK 2014. (September 2014)
2. Heino, N., Dietzold, S., Martin, M., Auer, S.: Developing semantic web applications with the ontowiki framework. In Pellegrini, T., Auer, S., Tochtermann, K., Schaffert, S., eds.: Networked Knowledge - Networked Media. Volume 221 of Studies in Computational Intelligence. Springer, Berlin / Heidelberg (2009) 61–77
3. Frischmuth, P., Martin, M., Tramp, S., Riechert, T., Auer, S.: OntoWiki - An Authoring, Publication and Visualization Interface for the Data Web. Semantic Web Journal (2014)
4. Jewell, T.D., Anderson, I., Chandler, A., Farb, S.E., Parker, K., Riggio, A., Robertson, N.D.M.: Electronic resource management – report of the dlf erm initiative. Technical report, Digital Library Federation, Washington, D.C. (2004) <http://old.diglib.org/pubs/dlf102/>.
5. Jewell, T., Aipperspach, J., Anderson, I., England, D., Kasprowski, R., McQuillan, B., McGear, T., Riggio, A.: Making good on the promise of erm: A standards and best practices discussion paper. Technical report, NISO ERM Data Standards and Best Practices Review Steering Committee, One North Charles Street, Suite 1905, Baltimore, MD 21201 (January 2012) http://www.niso.org/apps/group_public/document.php?document_id=7946&wg_abbrev=ermreview.
6. Berners-Lee, T.: Linked Data. Design issues, W3C (June 2009) <http://www.w3.org/DesignIssues/LinkedData.html>.

Extending an ontology alignment system with BIOPORTAL: a preliminary analysis*

Xi Chen¹, Weiguo Xia¹, Ernesto Jiménez-Ruiz², Valerie Cross¹

¹ Miami University, Oxford, OH 45056, United States

² University of Oxford, United Kingdom

1 Introduction

Ontology alignment (OA) systems developed over the past decade produced alignments by using lexical, structural and logical similarity measures between concepts in two different ontologies. To improve the OA process, string-based matchers were extended to look up synonyms for source and target concepts in background or external knowledge sources such as general purpose lexicons, for example, WordNet.³ Other OA systems such as SAMBO [8] and ASMOV [6] applied this approach but with specialized background knowledge, i.e. the UMLS Metathesaurus,⁴ for the anatomy track of the Ontology Alignment Evaluation Initiative⁵ (OAEI). Then a composition-based approach was proposed to use background knowledge sources such as Uberon⁶ and the Foundational Model of Anatomy⁷ (FMA) as intermediate ontologies [5] for the anatomy track. Here source concepts and target concepts are first mapped to the intermediate background ontology. If source and target concepts map to an exact match in the intermediate ontology, a mapping can be made between them. Other OA systems also followed with a composition-based approach using Uberon [1, 2].

One issue on the use of background knowledge sources is determining the best knowledge source on which to use these various alignment techniques. Previous OA systems using specialized knowledge sources have pre-selected specific biomedical ontologies such as Uberon for the anatomy track.

As a coordinated community effort, BIOPORTAL [3, 4] provides access to more than 370 biomedical ontologies, synonyms, and mappings between ontology entities via a set of REST services.⁸ By tapping into this resource, an OA system has access to the full range of these ontologies, including Uberon and many of the ontologies integrated in the UMLS Metathesaurus. Since BioPortal has not been exploited in the context of the OAEI, this paper examines two practical uses of BIOPORTAL as a generalized yet also specialized background knowledge source for the biomedical domain. We provide a preliminary investigation of the results of these two uses of BIOPORTAL in the OAEI's anatomy track using the LogMap system [7].

* This research was financed by the Optique project with grant agreement FP7-318338

³ <http://wordnet.princeton.edu/>

⁴ <http://www.nlm.nih.gov/research/umls>

⁵ <http://oaei.ontologymatching.org>

⁶ <http://obophenotype.github.io/anatomy/>

⁷ <http://sig.biostr.washington.edu/projects/fm/AboutFM.html>

⁸ <http://data.bioontology.org/documentation>

Algorithm 1 Algorithm to assess border-line mappings using BIOPORTAL

Input: $m = \langle e_1, e_2 \rangle$: mapping to assess; τ_1, τ_2 : thresholds; **Output:** true/false

```
1: Extract set of similar entities  $E_1$  from BIOPORTAL for entity  $e_1$ 
2: Extract set of similar entities  $E_2$  from BIOPORTAL for entity  $e_2$ 
3: if  $E_1 \neq \emptyset$  and  $E_2 \neq \emptyset$  then
4:   if  $\text{JaccardIndex}(E_1, E_2) > \tau_1$  then
5:     return true
6:   Extract mappings  $M_1$  from BIOPORTAL for entities in  $E_1$ 
7:   Extract mappings  $M_2$  from BIOPORTAL for entities in  $E_2$ 
8:   if  $M_1 \neq \emptyset$  and  $M_2 \neq \emptyset$  and  $\text{JaccardIndex}(M_1, M_2) > \tau_2$  then
9:     return true
10: return false
```

2 BIOPORTAL as an Oracle

Over the last few years, OA systems have made only minor improvements based on alignment performance measures of precision, recall, and F-score. This experience provides evidence that a performance upper bound is being reached using OA systems which are completely automatic. To increase their performance, some OA systems (e.g. LogMap) have included a semi-automatic matching approach which incorporates user interaction to assess borderline alignments (i.e. non “clear cut” cases with respect to their confidence values). For example, LogMap identifies 250 borderline mappings in the OAEI’s anatomy track when its interactive mode is active.

The research presented in this paper investigates replacing the human expert with an automated expert or “oracle” that relies on specialized knowledge sources in the biomedical domain. BIOPORTAL provides access to different resources including a wide variety of ontologies, classes within ontologies and mappings between the classes of different ontologies. For example, BIOPORTAL allows to search for ontology classes whose labels have an exact match with a given term. The oracle can use this capability to assist in determining whether a borderline mapping produced by an OA system should be included in the final alignment output or not (i.e. increasing its confidence).

Algorithm 1 shows the implemented method to assess a given mapping m between entities e_1 and e_2 using BIOPORTAL as an oracle.

3 BIOPORTAL as a Mediating Ontology Provider

Mediating ontologies are typically pre-selected specifically for the OA task. For example the top systems in the OAEI’s anatomy track used Uberon as (pre-selected) mediating ontology [5, 1, 2]. Limited research, however, has addressed the challenge of automatically selecting an appropriate mediating ontology as background knowledge [10, 9]. This research investigates using BIOPORTAL as a (dynamic) provider of mediating ontologies instead of relying on a few preselected ontologies.

Unlike [10] and [9], due to the large number of ontologies available in BIOPORTAL, we have followed a *fast-selection approach* to identify a suitable set of mediating

Algorithm 2 Algorithm to identify mediating ontologies from BIOPORTAL

Input: $\mathcal{O}_1, \mathcal{O}_2$: input ontologies; LM: a lexical matcher; N: stop condition**Output:** Top-5 (candidate) mediating ontologies \mathcal{MO}

- 1: Compute exact mappings \mathcal{M} between \mathcal{O}_1 and \mathcal{O}_2 using the lexical matcher LM
 - 2: Extract representative entity labels \mathcal{S} from \mathcal{M}
 - 3: **for each** $label \in \mathcal{S}$
 - 4: Get ontologies from BIOPORTAL that contains an entity with label $label$ (search call)
 - 5: Add to \mathcal{MO} the ontologies that provides synonyms for $label$ (record positive hits **I**)
 - 6: Record number of synonyms (**II**)
 - 7: Record ontology information: # of classes (**III**), depth (**IV**) and DL expressiveness (**V**)
 - 8: **stop condition:** if after N calls to BIOPORTAL \mathcal{MO} did not change then stop *iteration*
 - 9: **return** Top-5 ontologies from \mathcal{MO} according to the number of positive hits and synonyms
-

Table 1: Top 5 mediating (BIOPORTAL) ontologies for the OAEI’s anatomy track

#	Ontology	% pos. hits (I)	Avg. # syn. (II)	# classes (III)	Depth (IV)	DL exp. (V)
1	SNOMED CT	60%	5.1	401,200	28	<i>AL\mathcal{E}R</i>
2	UBERON	63%	3.3	12,091	28	<i>SRIQ</i>
3	MeSH	34%	5.0	242,262	16	<i>AL</i>
4	EFO	16%	5.1	14,253	14	<i>SROLF</i>
5	CL (Cell Onto.)	22%	3.3	5,534	19	<i>SH</i>

ontologies from BIOPORTAL (see Algorithm 2⁹). The fast-selection approach identifies entity labels that appear in the input ontologies and searches to find ontologies in BIOPORTAL that include those labels and contain synonyms for them. The algorithm stops if the number of identified mediating ontologies does not change after a specified number N of (search) calls to BIOPORTAL or when there are no more labels to check.

Table 1 shows the identified top-5 mediating ontologies for the OAEI’s anatomy track (with N=25 as stop condition). The ranking is based on the number of labels (i.e. search calls to BIOPORTAL) for which an ontology is able to provide synonyms (positive hits, **I**) and the average number of provided synonyms per positive hit (**II**). Additionally, information about the ontology is also given (**III–V**).

4 Preliminary evaluation

We have conducted a preliminary evaluation of the use of BIOPORTAL as a background knowledge provider (e.g. oracle and mediating ontology provider) in the OAEI’s anatomy track and with LogMap as OA system. For this purpose, we have extended LogMap’s matching process to (i) use Algorithm 1 as an oracle within its interactive mode (see Figure 3 in [7]); and (ii) use a mediating ontology \mathcal{MO} as in Algorithm 3.

The results¹⁰ are summarized in Table 2. Last column shows the original scores produced by LogMap (without BIOPORTAL). As expected, the best results in terms of

⁹ In the close future, we plan to combine this algorithm with the ontology recommender provided by BIOPORTAL: <https://bioportal.bioontology.org/recommender>

¹⁰ SNOMED and MeSH have been discarded as mediating ontologies. SNOMED is not available to download, and we were unable to download MeSH due to a time-out given by BIOPORTAL.

Algorithm 3 Use of a mediating ontology with LogMap

Input: $\mathcal{O}_1, \mathcal{O}_2$: input ontologies; \mathcal{MO} : mediating ontology; **Output:** \mathcal{M} : output mappings;

- 1: $\mathcal{M}_1 := \text{LogMap}(\mathcal{O}_1, \mathcal{MO})$
 - 2: $\mathcal{M}_2 := \text{LogMap}(\mathcal{MO}, \mathcal{O}_2)$
 - 3: $\mathcal{M}_C := \text{ComposeMappings}(\mathcal{M}_1, \mathcal{M}_2)$
 - 4: $\mathcal{M} := \text{LogMap}(\mathcal{O}_1, \mathcal{O}_2, \mathcal{M}_C)$
 - 5: **return** \mathcal{M}
-

Table 2: Results of LogMap with/without BIOPORTAL as background knowledge

Mode \ Score	LogMap - BIOPORTAL				LogMap
	Oracle	$\mathcal{MO}_{\text{Uberon}}$	\mathcal{MO}_{CL}	$\mathcal{MO}_{\text{EFO}}$	
Precision	0.915	0.899	0.907	0.914	0.913
Recall	0.846	0.927	0.867	0.846	0.846
F-score	0.879	0.913	0.886	0.879	0.878

F-score has been obtained using Uberon as mediating ontology. Using CL as mediator also improves the results with respect to those obtained by LogMap, although the improvement does not have an impact as big as with Uberon. There is not significant improvement using EFO as mediating ontology. Using BIOPORTAL as an oracle leads to a small increase in precision, but recall remains the same.

This preliminary evaluation has shown the potential of using BIOPORTAL as background knowledge. In the close future we plan to conduct an extensive evaluation involving more challenging datasets (e.g. OAEI’s largebio track) and other OA systems, and combining several mediating ontologies.

References

1. Cruz, I.F., Stroe, C., Caimi, F., Fabiani, A., Pesquita, C., Couto, F.M., Palmonari, M.: Using AgreementMaker to Align Ontologies for OAEI 2011. In: 6th OM Workshop (2011)
2. Faria, D., Pesquita, C., Santos, E., Palmonari, M., Cruz, I.F., Couto, F.M.: The Agreement-MakerLight Ontology Matching System. In: OTM Conferences. pp. 527–541 (2013)
3. Fridman Noy, N., Shah, N.H., Whetzel, P.L., Dai, B., et al.: BioPortal: ontologies and integrated data resources at the click of a mouse. *Nucleic Acids Research* 37, 170–173 (2009)
4. Ghazvinian, A., Noy, N.F., Jonquet, C., Shah, N.H., Musen, M.A.: What four million mappings can tell you about two hundred ontologies. In: Int’l Sem. Web Conf. (ISWC) (2009)
5. Gross, A., Hartung, M., Kirsten, T., Rahm, E.: Mapping composition for matching large life science ontologies. In: 2nd International Conference on Biomedical Ontology (ICBO) (2011)
6. Jean-Mary, Y.R., Shironoshita, E.P., Kabuka, M.R.: Ontology Matching with Semantic Verification. *Journal of Web Semantics* 7(3), 235–251 (2009)
7. Jiménez-Ruiz, E., Cuenca Grau, B., Zhou, Y., Horrocks, I.: Large-scale interactive ontology matching: Algorithms and implementation. In: European Conf. on Art. Int. (ECAI) (2012)
8. Lambrix, P., Tan, H.: A System for Aligning and Merging Biomedical Ontologies. *Journal of Web Semantics* 4(3), 196–206 (2006)
9. Quix, C., Roy, P., Kensch, D.: Automatic selection of background knowledge for ontology matching. In: Proc. of the Int’l Workshop on Sem. Web Inf. Management (2011)
10. Sabou, M., d’Aquin, M., Motta, E.: Exploring the Semantic Web as Background Knowledge for Ontology Matching. *J. Data Semantics* 11, 156–190 (2008)

How much navigable is the Web of Linked Data?*

Valeria Fionda¹, Enrico Malizia²

¹ Department of Mathematics, University of Calabria, Italy

² DIMES, University of Calabria, Italy

Abstract. Millions of RDF links connect data providers on the Web of Linked Data. Here, navigability is a key issue. This poster provides a preliminary quantitative analysis of this fundamental feature.

1 Motivation

Linked Data are published on the Web following the Linked Data principles [2]. One of them states that RDF links must be used to allow clients to navigate the Web of Linked Data from dataset to dataset. In particular, RDF links allow: *(i)* data publishers to connect the data they provide to data already on the Web; and *(ii)* clients to discover new knowledge by traversing links and retrieving data. Hence, navigability is a key feature. The Web of Linked Data is growing rapidly and both the set of data providers and the structure of RDF links continuously evolve. Is this growth taking place preserving the basic navigability principle?

In this poster we try to answer this question by analyzing the pay-level domain (PLD) networks extracted from the last three Billion Triple Challenge datasets. In addition, we also analyze the sameAs network obtained by considering only `owl:sameAs` links. Some recent works analyzed sameAs networks [1, 3] to provide some statistics on the deployment status and use of `owl:sameAs` links [3] and to evaluate their quality [1]. However, to the best of our knowledge, this is the first attempt to use the PLD and sameAs networks to perform a quantitative analysis of the navigability of the Web of Linked Data.

2 Methodology

Navigability indices. We model the Web of Linked Data as a directed graph $G = \langle V, E \rangle$ where $V = \{v_1, \dots, v_n\}$ is the set of vertices and $E \subseteq V \times V$ is the set of edges. The vertices of G represent the pay-level domains that identify data publishers in the Web of Linked Data. The edges of G represent directed links between *different* pay-level domains (i.e., there are no loops) and are *ordered*

*V. Fionda's work was supported by the European Commission, the European Social Fund and the Calabria region. E. Malizia's work was supported by the ERC grant 246858 (DIADEM), while he was visiting the Department of Computer Science of the University of Oxford.

pairs of vertices (v_i, v_j) , where v_i is the source vertex and v_j is the target one. Intuitively, an edge (v_i, v_j) models the fact that there is at least one URI having PLD v_i by dereferencing which an RDF link to a URI having PLD v_j is obtained.

We denote by $v_i \rightsquigarrow_G v_j$ the existence of a path from v_i to v_j in G (otherwise we write $v_i \not\rightsquigarrow_G v_j$). For a graph $G = \langle V, E \rangle$, $G^* = \langle V, E^* \rangle$ is the *closure* of G where $(v_i, v_j) \in E^*$ if and only if $v_i \neq v_j$ and $v_i \rightsquigarrow_G v_j$. We define the *reachability matrix* $R_G \in \{0, 1\}^{n \times n}$ such that $R_G[i, j] = 1$ if and only if $(v_i, v_j) \in E^*$ (i.e., $v_i \rightsquigarrow_G v_j$). Moreover, we define the *distance matrix* $D_G \in \mathbb{N}^{n \times n}$ such that $D_G[i, j]$ is the length of the shortest path between v_i and v_j ($D_G[i, j] = \infty$ if $v_i \not\rightsquigarrow_G v_j$). When G is understood we simply write $v_i \rightsquigarrow v_j$, $R[i, j]$, and $D[i, j]$.

To evaluate the navigability of the Web of Linked Data we use two indices. The first one is the *reachability index* $\eta(G)$, corresponding to the *edge density* of G^* . The reachability index is the probability that between any two given vertices of G there exists a path. In particular, $\eta(G) = \frac{1}{n(n-1)} \sum_{v_i, v_j \in V, v_i \neq v_j} R[i, j] = \frac{|E^*|}{n(n-1)}$. This index takes into account only the reachability between vertices and implies that $\eta(G_1) = \eta(G_2)$ for any pair of graphs $G_1 = \langle V, E_1 \rangle$ and $G_2 = \langle V, E_2 \rangle$ such that $G_1^* = G_2^*$, even if $E_1 \subset E_2$ (or $E_2 \subset E_1$).

To take into account differences in graph topologies, we use the *efficiency index* $\tilde{\eta}(G)$ [4]. This index exploits the distance matrix D to weight the reachability by the (inverse of the) length of the shortest path between vertices. Given a graph G , $\tilde{\eta}(G) = \frac{1}{n(n-1)} \sum_{v_i, v_j \in V, v_i \neq v_j} \frac{R[i, j]}{D[i, j]}$, where $\frac{R[i, j]}{D[i, j]} = 0$ when $v_i \not\rightsquigarrow v_j$. It can be shown that for any graph G , $\tilde{\eta}(G) \leq \eta(G)$, and given two graphs $G_1 = \langle V, E_1 \rangle$ and $G_2 = \langle V, E_2 \rangle$ such that $E_1 \subset E_2$ then $\eta(G_1) \leq \eta(G_2)$, while $\tilde{\eta}(G_1) < \tilde{\eta}(G_2)$. The index $\tilde{\eta}(\cdot)$ has been used in literature to measure how efficiently small-world networks exchange information [4].

Intuitively, the closer $\eta(G)$ is to 1, the more G is similar to a strongly connected graph; on the other hand, the closer $\tilde{\eta}(G)$ is to 1, the more G is similar to a complete graph. Note that, $\tilde{\eta}$ combines information on reachability *and* information about the distances between pairs of vertices and it is not simply the arithmetic mean of the inverse of the shortest paths lengths.

Datasets. To perform our analysis we used the Billion Triple Challenge (BTC) datasets of 2010, 2011 and 2012³. Unfortunately, the BTC dataset for 2013 was not crawled and a dataset for 2014 was not available to the date of submission. We decided to use the pay-level domain (PLD) granularity to build our networks, where the PLD of a URI is a sub-domain (generally one level below) of a generic public top-level domain, for which users usually pay for. PLD domains in the Web of Linked Data are often in one-to-one correspondence with Linked Open Data datasets. We extracted the PLD network from each BTC dataset by considering each RDF quad and adding an edge between the PLD of the context URI and the PLD of the subject and object. In particular, we extracted two PLD networks: the first one (denoted by ALL) considers all types of links and the second one (denoted by SA) considers only `owl:sameAs` links.

³<http://km.aifb.kit.edu/projects/btc-X/>, X={2010,2011,2012}

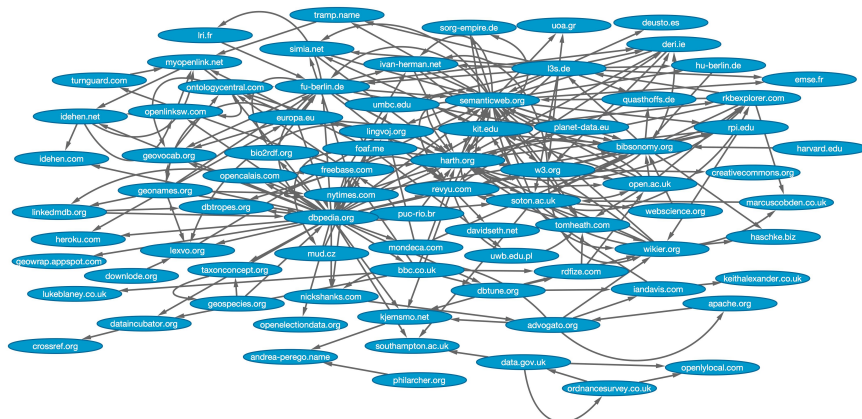


Fig. 1. The largest internal connected component of the PLD sameAs network extracted from the BTC2012 dataset.

Since BTC datasets are obtained by crawling the LOD cloud starting from a set of seed URIs, we also extracted from each network the largest internal connected component obtained by ignoring the PLDs “on the border” (i.e., those without any outgoing edge) that are probably those where the crawling stopped. We denote by ALL-I and SA-I are internal subnetworks extracted from ALL and SA, respectively. Fig. 1 shows the SA-I network of the BTC 2012 dataset.

3 Evaluation

Table 1 reports our results. The table shows that η and $\tilde{\eta}$ on the complete network, for both ALL and SA, decrease in 2011 with respect to 2010 and are still decreasing for the ALL network even in 2012. Moreover, the values obtained for both η and $\tilde{\eta}$ are very small. For example, $\eta(\text{ALL}) = 4.899 \cdot 10^{-4}$ for the BTC 2012 dataset means that given a random pair of PLDs from the ALL network the probability that they are connected by a path is less than 0.5%. Translated to the Web, this means that starting from a given a URI and following RDF links only a very small portion of the Web of Linked Data can be reached. However, an explanation for such a behavior could be that the BTC datasets are obtained by crawling the Web and it is reasonable to think that PLDs at the “border” of the network are those at which the crawling process stopped. If this is the case, some links can actually be missing and our indices can be biased. In general, a decrease over time of η and $\tilde{\eta}$ on the full Web of Linked Data highlight a decrease in its navigability. Nevertheless, since in our case the BTC datasets are used as representative samples of the full Web of Linked Data, this decrease can be related to the fact that in 2012 and 2011 the crawler retrieved triples spanning more data providers than in 2010 and a large portion of them is on the

Index/Year Network	η			$\tilde{\eta}$		
	2010	2011	2012	2010	2011	2012
All	0.034	0.002	$4.899 \cdot 10^{-4}$	0.009	$5.98 \cdot 10^{-4}$	$1.719 \cdot 10^{-4}$
SA	0.134	0.018	0.059	0.037	0.005	0.016
All-I	0.312	0.887	0.867	0.089	0.319	0.326
SA-I	0.400	0.658	0.497	0.131	0.223	0.187

Table 1. Summary of the analysis carried out.

border. Indeed, in general, both η and $\tilde{\eta}$ decrease if the proportion of the nodes on the border increase with respect to the total size of the network.

For this reason we decided to analyze the internal largest connected components ALL-I and SA-I. As for ALL-I, on one hand it can be observed that the difference in the values of both indices between 2011 and 2012 is negligible. There is, on the other hand, a big increase in both indices for the 2011 network with respect to the 2010 one. It is evident, from these results, that the Web of Linked Data gained a lot in navigability from 2010 to 2011, according to the ALL-I sample, while the navigability remained almost unchanged in 2012. A similar trend can be identified also on the SA-I network, apart from a noticeable decrease in the navigability of the 2012 network compared to the 2011 one. Our results show that, for example, in 2012 given a random pair of PLDs from the ALL-I network the probability that they are connected by a path is greater than 86% with an efficiency of 0.326. It is worth to point out that, in a distributed environment as the Web, efficiency is a fundamental property that, besides reachability, is related to the number of dereferences that must be performed to move from a source PLD to a target one. Roughly speaking, lower values of efficiency for the same value of reachability translate in more traffic on the network.

4 Conclusions

Navigability is a key feature of the Web of Linked Data. We introduced some indices to quantitatively measure the connectivity of Linked Data providers and the efficiency of their connections. The results obtained show that, as hoped, the navigability of the Web of Linked Data is increasing with its growth. However, in order not to be biased toward a certain interpretation, it is important to stress that the results obtained could have been influenced by the crawling strategy used to build the BTC datasets used in our analysis. We plan to perform our analysis in the following years to monitor and hopefully confirm this trend.

References

1. G. Bartolomeo and S. Salsano. A spectrometry of linked data. In *LDOW*, 2012.
2. T. Berners-Lee. Linked data design issues, 2006.
3. L. Ding, J. Shinavier, Z. Shanguan, and D. McGuinness. SameAs Networks and Beyond: Analyzing Deployment Status and Implications of owl:sameAs in Linked Data. In *ISWC*, 2010.
4. V. Latora and M. Marchiori. Efficient behavior of small-world networks. *Phys. Rev. Lett.*, 87(19):198701, 2001.

A Framework for Incremental Maintenance of RDF Views of Relational Data

Vânia M. P. Vidal¹, Marco A. Casanova², José M. Monteiro¹, Narciso Arruda¹,
Diego Sá¹, and Valéria M. Pequeno³

¹ Federal University of Ceará, Fortaleza, CE, Brazil

{vvidal, jmmfilho, narciso, diego}@lia.ufc.br

² Pontifical Catholic University of Rio de Janeiro, RJ, Brazil

casanova@inf.puc-rio.br

³ DMIR, INESC-ID Porto Salvo, Portugal

vmp@inesc-id.pt

Abstract. A general and flexible way to publish relational data in RDF format is to create RDF views of the underlying relational data. In this paper, we demonstrate a framework, based on rules, for the incremental maintenance of RDF views defined on top of relational data. We also demonstrate a tool that automatically generates, based on the mapping between the relational schema and a target ontology, the RDF view exported from the relational data source and all rules required for the incremental maintenance of the RDF view.

Keywords: RDF View Maintenance, RDB-to-RDF, Linked Data

1 Introduction

The Linked Data initiative [1] promotes the publication of previously isolated databases as interlinked RDF triple sets, thereby creating a global scale dataspaces, known as the Web of Data. However, the full potential of linked data depends on how easy it is to publish data stored in relational databases (RDBs) in RDF format. This process is often called RDB-to-RDF.

A general way to publish relational data in RDF format is to create RDF views of the relational data. The contents of views can be materialized to improve query performance and data availability. However, to be useful, a materialized view must be continuously maintained to reflect dynamic source updates.

In this demo, we show a framework, based on rules, for the incremental maintenance of external RDF views defined on top of relational data. Figure 1 depicts the main components of the framework. Briefly, the administrator of a relational data-base, using *Rubya* (**R**ules **by** assertion), should create RDF views and define a set of rules using *Rubya* - Figure 1(a). These rules are responsible for: (i) computing the view maintenance statements necessary to maintain a materialized view \mathbf{V} with respect to base updates; and (ii) sending the view maintenance statements to the *view controller* of \mathbf{V} - Figure 1(b). The rules can be implemented using triggers. Hence, no middleware system is required. The

view controller for the RDF view has the following functionality: (i) receives the view maintenance updates from the RDB server and (ii) applies the updates to the view accordingly.

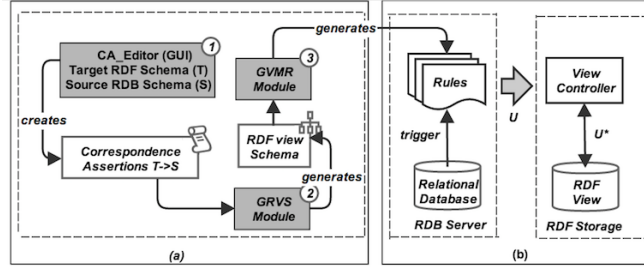


Fig. 1. Suggested Framework.

Our approach is very effective for an externally maintained view because: the view maintenance rules are defined at view definition time; no access to the materialized view is required to compute the view maintenance statements propagated by the rules; and the application of the view maintenance statements by the *view controller* does not require any additional queries over the data source to maintain the view. This is important when the view is maintained externally [4], because accessing a remote data source may be too slow.

The use of rules is therefore an effective solution for the incremental maintenance of external views. However, creating rules that correctly maintain an RDF view can be a complex process, which calls for tools that automate the rule generation process. In Section 2, we further detail the *Rubya* tool that, based on the mapping between the relational schema and a target ontology, automatically generates the RDF view exported from the relational data source and the set of rules required for the incremental maintenance of the RDF view.

The demo video is available at <http://tiny.cc/rubya>. First, the video shows, with the help of a real-word application, the process of defining the RDF view and generating the maintenance rules with *Rubya*. Then, it shows some practical examples of using the rules for incremental maintenance of a materialized RDF view. For more information see <http://www.arida.ufc.br/ivmf/>.

2 Generating Rules with Rubya

Figure 1 highlights the main components of *Rubya*. The process of defining the RDF view and generating the maintenance rules with *Rubya* consists of three steps:

STEP 1 (Mapping specification): Using the correspondence assertions editor of *Rubya*, the user loads the source and target schema and then he can

draw correspondence assertions (CAs) to specify the mapping between the target RDF schema and the source relational schema. The demo video shows how the *CA_Editor* helps the user graphically to define CAs.

A CA can be: (i) a class correspondence assertion (CCA), which matches a class and a relation schema; (ii) an object property correspondence assertion (OCA), which matches an object property with paths (list of foreign keys) of a relation schema; or (iii) a datatype property correspondence assertion (DCA), which matches a datatype property with attributes or paths of a relation schema. CAs have a simple syntax and semantics and yet suffice to capture most of the subtleties of mapping relational schemas into RDF schemas. Figure 2 shows some examples of correspondence assertions between the relational schema *ISWC_REL* and the ontology *CONF_OWL*. CCA1 matches the class *foaf:Person* with the relation *Persons*. We refer the reader to [4, 5] for the details and motivation of the mapping formalism.

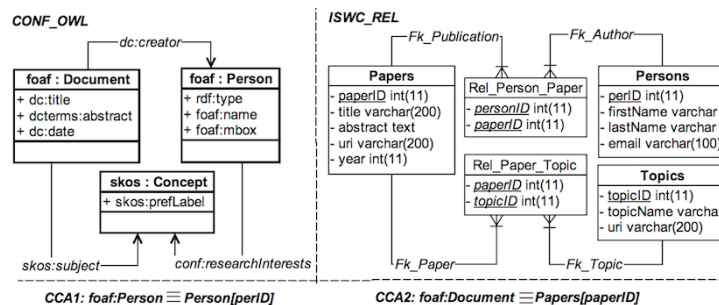


Fig. 2. *CONF_OWL* and *ISWC_REL* schemas and some examples of CAs.

STEP 2 (RDF view creation): The GRVS module automatically generates the RDF view schema, which is induced by the correspondence assertions defined in Step 1. The vocabulary of the RDF view schema contains all the elements of the target RDF schema that match an element of the source relational schema. **STEP 3 (Rule generation):** The GVMR module automatically generates the set of rules required to maintain the RDF view defined in Step 2. The process of generating the rules for a view *V* consists of the following steps: (a) Obtain, based on the CAs of *V*, the set of all relations in the relational schema that are relevant to *V*. (b) For each relation *R* that is relevant to *V*, three rules are generated to account for insertions, deletions and updates on *R*.

Two procedures, *GVU_INSERTonR* and *GVU_DELETEonR*, are automatically generated, at view definition time, based on the CAs of *V* that are relevant to *R*. Note that an update is treated as a deletion followed by an insertion, as usual. *GVU_INSERTonR* takes as input a tuple r_{new} inserted in *R* and returns the updates necessary to maintain the view *V*. *GVU_DELETEonR* takes as input a tuple r_{old} deleted from *R* and returns the updates necessary to maintain

the view \mathbf{V} . In [4], we present the algorithms that compile $GVU_INSERTonR$ and $GVU_DELETEonR$ based on the CAs of \mathbf{V} that are relevant to R .

Once the rules are created, they are used to incrementally maintain the materialized RDF view. For example, Figure 3 shows the process to update a RDF view when an insertion occurs on Papers. When an insertion occurs on Papers, a corresponding trigger is fired. The trigger computes the view maintenance statements U , and sends it to the view controller. The view controller computes the view updates U^* , and applies it to the view state.

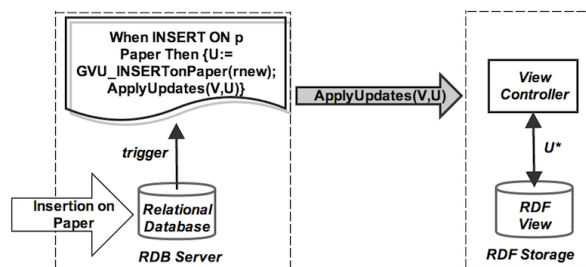


Fig. 3. Using the rules generated by *Rubya* when insertions occurs on Papers.

3 Conclusions

In this paper, we present *Rubya*, a tool for incremental maintenance of external RDF views defined on top of relational data. There is significant work on reusing relational data in terms of RDF (see a survey in [3]). Karma [2], for example, is a tool to semi-automatically create mapping from a source to a target ontology. In our tool, the user defines mappings between a source and a target ontology using a GUI. The novelty of our proposal is that we generate rules to maintain the RDF views.

References

1. Berners-lee, T., design issues: Linked data, <http://www.w3.org/DesignIssues/LinkedData.html>
2. Knoblock, C.A., et al.: Semi-automatically Mapping Structured Sources into the Semantic Web. In: ESWC, pp. 375–390. Springer-Verlag, Berlin, Heidelberg (2012)
3. Spanos, D.E., Stavrou, P., Mitrou, N.: Bringing Relational Databases into the Semantic Web: A Survey. *Semantic Web Journal* **3**(2), 169–209 (2012)
4. Vidal, V.M.P., Casanova, M.A., Cardoso, D.S.: Incremental Maintenance of RDF Views of Relational Data. In: OTM 2013 Conferences, pp. 572–587. Austria (2013)
5. Vidal, V.M.P., Casanova, M.A., Neto, L.E.T., Monteiro, J.M.: A Semi-Automatic Approach for Generating Customized R2RML Mappings. In: SAC, pp. 316–322 (2014)

Document Relation System Based on Ontologies for the Security Domain

Janine Hellriegel¹, Hans Georg Ziegler¹, and Ulrich Meissen¹

¹ Fraunhofer Institute for Open Communication Systems (FOKUS),
Kaiserin Augusta Allee 31, 10589 Berlin, Germany

{ Janine.Hellriegel, Hans.Georg.Ziegler,
Ulrich.Meissen }@fokus.fraunhofer.de

Abstract. Finding semantic similarity or semantic relatedness between unstructured text documents is an ongoing research field in the semantic web area. For larger text corpuses often lexical matching – the matching of shared terms – is applied. Related semantic terms and concepts are not considered in this solution. Also documents that use heterogeneous perspectives on a domain could not be set into a relation properly. In this paper, we present our ongoing work on a flexible and expandable system that handles text documents with different points of view, languages and level of detail. The system is presented in the security domain but could be adapted to other domains. The decision making process is transparent and the result is a ranked list.

Keywords: Document Relation, Security, Ontology, Semantic Relatedness

1 Introduction

The amount of available information in the Internet is growing day by day. It is difficult to keep an overview of relevant data in a domain, especially if different kinds of views on the same topic are considered. An expert is using different words and level of detail in contrast to a normal user, but they describe exactly the same concept. Having a database consisting of documents authored from people with different levels of expertise, language skills and ambitions imposes a big challenge on a semantic search algorithm. The usage of long texts as search input enables a wider range of search terms, which is the foundation to detect a larger spectrum of documents. The relevant results are documents related to the input query text document. A basic method to compare two text documents is the vector space model [2], which relates the text similarity to the amount of similar words. However, semantically related words are not considered. Knowledge based similarity measures use larger document corpuses and external networks like WordNet or Wikipedia to analyze co-occurrences and relations. An overview of these techniques is presented in [3] but most of the methods just work for a couple words as search query. Although all documents affiliate to one domain (e.g. the security domain) lexical matching and knowledge-based measure don't retrieve a sufficient number of related documents. Another measure, the

Ontology based matching includes concepts and heterogeneous relations. Wang [7] proposes a system to relate documents using the concepts found in WordNet. But the measurement step still depends on words and heterogeneous concepts could not be related. In the security and safety domain only specialized ontologies exist [5], [6], that mainly focus on the security of information systems. An attempt to combine different ontologies was made by [1] but could not express the diversity of the domain also addressing e.g. security of citizens, infrastructures or utilities. As the mentioned references show, a system that searches for related text documents in a clear and traceable way is not yet developed. At the moment no ontology exists that would match the terminology of the whole security domain. Therefore a new, more general, ontology as well as a general system are developed.

2 System of Semantic Related Documents

The fundament to measure semantic relatedness between two documents are terms. A terminology is built, which is used to compare all documents quickly and determine their relations. The whole system is divided into three steps. Figure 1 displays an overview of the whole system.

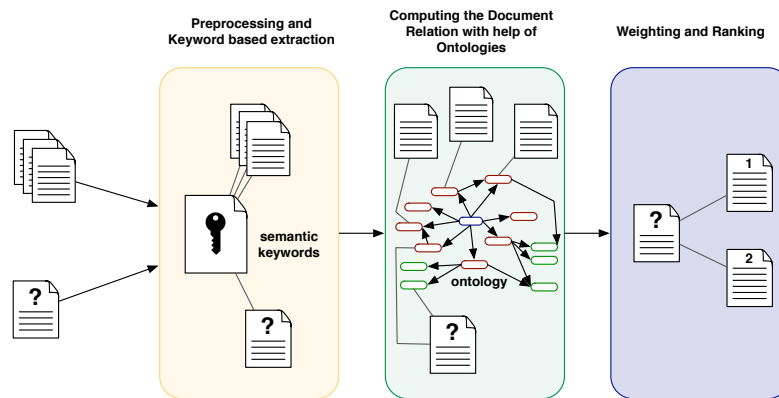


Fig. 1. System overview with three steps to determine document relatedness

A possible scenario is the goal to find related work and potential partners for a project idea. In the first step predefined keywords are extracted from the project descriptions and organization profiles as well as the query text containing the project idea to discover terms that characterize the documents. Each document is now represented with the detected keywords. In the second step the text documents are classified in the ontology according to their keywords in order to discover further relations. If the keyword *maritime borders* appears in the query document, all relations from this keyword to others, like *border surveillance*, are used. The ontology helps to discover related keywords and therefore related documents. With the help of a weighting algorithm a ranked list of related documents is the result in the last step.

2.1 Preprocessing and Keyword based extraction

In order to extract the valuable terms from the documents, a manually created keyword list is used and their term frequency for each document is determined. Comparing the occurrence of the keywords gives a first measure for the relatedness. The more terms the texts have in common, the more related they are. However, different views and special relations are not yet taken into account. In order to extract the keywords all documents are preprocessed with a tokenization on term bases. Further, stemming algorithms are used to transform all terms in the documents as well as all keywords to their base form. The keyword list was developed by early-warning system experts together with civil protection and police specialists. It contains about 500 English words relevant to the security domain but still could be modified or extended. Automatic keyword extraction algorithms are not suitable since they produce too much noise and could not hold up to the quality of the keywords list. All keywords can be translated semi-automatically. Therefore the system supports different languages. Synonyms, categories and other semantic relevant words are added by using BabelNet [4]. From the term *video surveillance* the terms *surveillance camera*, *cctv*, *video home security system* are derived. In total a keyword list with over 4000 terms has been produced. In this way it is ensured, that only domain related terms are found.

2.2 Computing the Document Relation with help of Ontologies

In the case when related documents don't contain identical or similar terms, an ontology or terminological net can be used in order to improve the calculation of the relatedness. The relation between a technical and a user view could only be determined over a shared concept. Using the heterogeneous paths between the terms in the graph-based knowledge representation, new relations between the documents are revealed. Not only the distances in the terminological net are considered, also the type of relation like *is-a* or *part-of* between the terms determines the relatedness of the text documents. In this way, for each detected keyword in the query document, related keywords could be found. Texts containing the related keywords are most likely to correlate with the query document. A new ontology in the security domain is manually built at the moment, containing the original 500 keywords, relations from BabelNet and a taxonomy created by security researchers. The taxonomy is loosely based on a project categorization for the recent FP 7 Cordis security call [8].

2.3 Weighting and Ranking

A ranked list of texts related to the query document is the result of the system. Two measurements are used to rank the results, first is the weighting of the original keywords and second is the type of relation between the keywords. Not all retrieved terms are equally important to distinguish the texts. The term *security* is important but very general and can be found in a lot of documents. Due to the low entropy of the term, it does not help to find unique relations. In contrast, the term *body scanner* is more useful to find related documents. A term weighting is applied with the tf-idf

statistic [2] to identify significant terms. As document corpus the FP 7 Cordis security call project descriptions are used. Secondly the relation between two specific keywords (*body scanner* and *metal detector*) is ranked higher than a relation between a specific keyword and a more general keyword (*body scanner* and *airport security*).

3 Conclusion and Future Work

With the presented system, a ranked list of related documents can be retrieved. Regardless what kind of view or level of detail they contain. The system describes a general sequence of functions and could be adapted to other domains if a corresponding keyword list and ontology are available. In the music domain e.g. artist profiles could be related to genre or instrument descriptions. The system is based on a simple method but achieves good results because it works close to the domain. In addition, it allows the evaluation of the results and to understand why documents are identified as related. The system is still work in progress, the next steps are to complete the development of the ontology and to evaluate the chosen keywords. Further evaluations concerning the accuracy as well as user satisfaction have to be performed.

References

1. Liu, Shuangyan, Duncan Shaw, and Christopher Brewster: Ontologies for Crisis Management: a Review of State of the Art in Ontology Design and Usability. In: Proceedings of the Information Systems for Crisis Response and Management Conference (2013)
2. Manning, Christopher D., Prabhakar Raghavan, and Hinrich Schütze: Introduction to Information Retrieval. Vol. 1. Cambridge university press Cambridge (2008)
3. Mihalcea, Rada, Courtney Corley, and Carlo Strapparava: Corpus-based and Knowledge-based Measures of Text Semantic Similarity. In: AAAI, 6:775–80 (2006)
4. Navigli, Roberto, and Simone Paolo Ponzetto: BabelNet: Building a Very Large Multilingual Semantic Network. In: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, 216–25. Association for Computational Linguistics (2010)
5. Ramanauskaite, Simona, Dmitrij Olfier, Nikolaj Goranin, and Antanas Čenys: Security Ontology for Adaptive Mapping of Security Standards. In: International Journal of Computers Communications & Control 8, no. 6 (2013)
6. Souag, Amina, Camille Salinesi, and Isabelle Comyn-Wattiau: Ontologies for Security Requirements: A Literature Survey and Classification. In: Advanced Information Systems Engineering Workshops, 61–69. Springer (2012)
7. Wang, James Z., and William Taylor: Concept Forest: A New Ontology-assisted Text Document Similarity Measurement Method. In: Web Intelligence, IEEE/WIC/ACM International Conference On, 395–401. IEEE (2007)
8. FP7 Cordis Project, http://cordis.europa.eu/fp7/security/home_en.html

Acknowledgement

This work has received funding from the Federal Ministry of Education and Research for the security research project “fit4sec” under grant agreement no. 13N12809.

Representing Swedish Lexical Resources in RDF with *lemon*

Lars Borin¹ and Dana Dannélls¹ and Markus Forsberg¹ and John P. McCrae²

¹ Språkbanken, University of Gothenburg

{lars.borin, dana.dannells, markus.forsberg}@svenska.gu.se

² Cognitive Interaction Technology Center of Excellence, University of Bielefeld
jmccrae@cit-ec.uni-bielefeld.de

Abstract. The paper presents an ongoing project which aims to publish Swedish lexical-semantic resources using Semantic Web and Linked Data technologies. In this article, we highlight the practical conversion methods and challenges of converting three of the Swedish language resources in RDF with *lemon*.

Keywords: Language Technology, Lexical Resources, Lexical Semantics.

1 Introduction

For state-of-the-art lexical resources, availability as Linked Open Data (LOD) is a basic requirement for their widest possible dissemination and use in research, education, development of products and services. In addition, in order to provide language support to individuals requiring augmentative and alternative communication (AAC) we need linguistic resources suitably organized and represented, e.g., sign language material, symbol and image libraries adapted to multiple cognitive levels, as well as textual support in many languages. So far, in Sweden, these resources have been developed as separate and uncoordinated efforts, either commercially or by non-profit organizations targeting specific groups and needs. In the long run, this is an exclusive and expensive way of proceeding, leading to limited usefulness. In the project, we aim to link Concept Coding Framework (CCF) technology and some symbol sets, to a common LOD format for languages resources (LRs) to be developed together with Språkbanken (The Swedish Language Bank),³ which will be a great step forward.

There are ongoing international initiatives aiming to define suitable formats for publishing linguistic content according to linked open data principles [1]. Integrating linguistic content on the Web by using these principles is central for many language technology applications. It requires harmonization on different levels in particular on the metadata level. In this paper we present our first attempt to publish three Swedish lexical-semantic resources in RDF with *lemon* [2].

³ <<http://spraakbanken.gu.se/>>

2 *lemon*

lemon (Lexicon Model for Ontologies) is a model for associating linguistic information with ontologies,⁴ in particular Semantic Web ontologies. The model builds on existing models for incorporating multilingual knowledge in ontologies, and for the representation of lexical resources [3]. *lemon* is built around the principal of *semantics by reference* [4]. It separates the lexical layer, that is the words and their morphology and syntactic behaviour, and the semantic layer in the ontology, which describes the domain-specific meaning of that entry. The model of *lemon* is based around lexical entries, which connect to ontology entities, by means of an object called *LexicalSense*, which refers to one meaning of a word, or correspondingly a pair consisting of the word and the meaning. In this sense, the model of *lemon* is primarily semasiological, i.e. organized around words, as opposed to the onomasiological resources, such as SALDO, which are primarily built around senses. However, the usage of the sense object and the distributed nature of the RDF graph model, means that from a linked data viewpoint this distinction is of less relevance, and *lemon* proves to be an effective model for the lexical resources discussed here.

The *lemon* model has since 2011 been the focus of the W3C OntoLex community group,⁵ and as such significant developments on both the model and its applications are still active. In particular, *lemon* has already been used successfully for the representation of several existing lexical resources, most notably WordNet [5], UBY [6] and BabelNet [7]. Furthermore, the use of *lemon* has already proved to be a key component in systems for tasks such as question answering [8], natural language generation [9] and information extraction [10].

3 Converting the Swedish Lexical Resources into RDF with *lemon*

Språkbanken at the Department of Swedish, University of Gothenburg, Sweden maintains a very large collection of lexical resources for both modern and historical Swedish. Currently there exist 23 different lexical resources with over 700,000 lexical entries. Within the time frame of our project we so far considered three of the modern lexicons, which are also freely available in Lexical Markup Framework (LMF) [11]. As we will show in this chapter, the form of these lexical resources varies substantially. We minimize this variation with *lemon*. Since *lemon* is built on LMF, it allows easy conversion supported by EXtensible Stylesheet Language XSL Transformation mechanism.⁶

SALDO, the new version of the Swedish Associative Thesaurus [12], is a semantically organized lexicon containing morphological and lexical-semantic information for more than 130,000 Swedish lexical entries of which 13,000 are verbs.⁷ It is the largest freely available electronic Swedish lexical resource for language technology, and is the pivot of all the Swedish lexical resources maintained at Språkbanken. SALDO entries

⁴ <<http://lemon-model.net>>

⁵ <<http://www.w3.org/community/ontolex/>>

⁶ <<http://www.w3.org/Style/XSL/>>

⁷ <<http://spraakbanken.gu.se/saldo>>

are arranged in a hierarchical structure capturing semantic closeness between senses indicated by a unique sense identifier, in *lemon* this unique identifier is represented with the object *lemon:LexicalSense*. A challenge here was how to represent SALDO's *lemgram* which is a pairing of the word base form and its inflectional paradigm. *Lemgram* is represented with the object *lemon:LexicalEntry*. The base form is described with a lemma value of the lexical entry and is represented with the object *lemon:Form*. The inflectional paradigm is described with a form value combined with a digit, and is also represented with the object *lemon:Form*. We defined our objects for capturing the paradigm patterns and the morphosyntactic tags.

Swedish FrameNet (SweFN), created as a part of a large project called SweFN++ [13], is a lexical-semantic resource that has been expanded from and constructed in line with Berkeley FrameNet (BFN) [14].⁸ It is defined in terms of semantic frames. Each frame is unique and is evoked by one or more target word(s) called *lexical unit* (LU) which carries valence information about the possible syntactic and semantic realizations of frame elements (FEs). Frames are represented with the object *lemon:LexicalSense*. The LUs evoked by a frame are linked to SALDO entities with the property *lemon:isSenseOf*. The property *lemon:SemArg* links to FEs objects. There are two types of FEs: *core* and *peripheral*, these are represented with the object *lemon:Argument* and are linked to either *uby:core* or *uby:peripheral* with the property *uby:coreType*. A challenge here was how to represent the syntactic and semantic realizations of FEs that are illustrated with annotated example sentences. In the LMF file they are annotated with extra, non-standardized tags. Our solution was to define our object *karpHash:example* to represent the annotated example sentences for each FE.

Lexin is a bilingual dictionary,⁹ originally developed for immigrants by the Swedish national agency for education.¹⁰ It contains detailed linguistic information for 15 languages including sentence and expression examples, sentence constructions, explanations through comments, synonyms, etc. A lexical entry in Lexin is represented with *lemon:LexicalSense*. Entries are linked to SALDO entries with *owl:sameAs* property. In addition, we defined the objects *spraakbanken:translation* and *spraakbanken:synonym* to represent translation equivalents of sentences in our RDF model.

4 Summary

We described the effort of transforming three Swedish lexical resources into LOD using Semantic Web and Linked Data technologies.¹¹ Deciding on how to transform the lexical resource attributes to *lemon* features has been carried out manually for each resource. Once the transformation is decided, the integration is conducted automatically. Publishing lexical resources in Swedish as RDF data is valuable for a variety of use cases. One of the many benefits of having this semantically interlinked content is to

⁸ <<http://spraakbanken.gu.se/eng/resource/swefn>>

⁹ <<http://spraakbanken.gu.se/eng/resource/lexin>>

¹⁰ <<http://www.skolverket.se/>>

¹¹ The published resources can be accessed here: <<http://spraakbanken.gu.se/rdf>>

enhance accessibility and availability on the web, in particular for language technology applications.

Acknowledgements

The research presented here has been conducted with funding by VINNOVA (Swedish Governmental Agency for Innovation Systems; grant agreement 2013-04996), and by the University of Gothenburg through its support of the Centre for Language Technology.¹²

References

1. Chiacros, C., Nordhoff, S., Hellmann, S., eds.: *Linked Data in Linguistics. Representing and Connecting Language Data and Language Metadata*. Springer (2012)
2. McCrae, J., Spohr, D., Cimiano, P.: Linking lexical resources and ontologies on the semantic web with lemon. In: *The Semantic Web: Research and Applications*. (2011) 245–259
3. Cimiano, P., Buitelaar, P., McCrae, J., Sintek, M.: Lexinfo: A declarative model for the lexicon-ontology interface. *Web Semantics: Science, Services and Agents on the World Wide Web* **9**(1) (2011)
4. Buitelaar, P. In: *Ontology-based Semantic Lexicons: Mapping between Terms and Object Descriptions*. Cambridge University Press (2010) 212–223
5. McCrae, J.P., Fellbaum, C., Cimiano, P.: Publishing and linking WordNet using RDF and lemon. In: *Proceedings of the 3rd Workshop on Linked Data in Linguistics*. (2014)
6. Eckle-Kohler, J., McCrae, J., Chiacros, C.: lemonUby-a large, interlinked, syntactically-rich resource for ontologies. *Semantic Web Journal*, submitted. (2014)
7. Ehrmann, M., Vannela, D., McCrae, J.P., Cecconi, F., Cimiano, P., Navigli, R.: Representing Multilingual Data as Linked Data: the Case of BabelNet 2.0. In: *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC)*. (2014)
8. Unger, C., Cimiano, P.: Pythia: Compositional meaning construction for ontology-based question answering on the semantic web. In Munoz, R., ed.: *Natural Language Processing and Information Systems: 16th International Conference on Applications of Natural Language to Information Systems*. Volume 6716., Springer (2011) 153–160
9. Cimiano, P., Lüker, J., Nagel, D., Unger, C.: Exploiting ontology lexica for generating natural language texts from RDF data. In: *Proceedings of the 14th European Workshop on Natural Language Generation*. (2013) 10–19
10. Davis, B., Badra, F., Buitelaar, P., Wunner, T., Handschuh, S.: Squeezing lemon with GATE. In: *Proceedings of the First Workshop on the Multilingual Semantic Web*. (2011) 74
11. Francopoulo, G., George, M., Calzolari, N., Monachini, M., Bel, N., Pet, M., Soria, C., et al.: Lexical markup framework (LMF). In: *International Conference on Language Resources and Evaluation LREC*. (2006)
12. Borin, L., Forsberg, M., Lönngrén, L.: SALDO: a touch of yin to WordNet’s yang. *Language Resources and Evaluation* **47**(4) (2013) 1191–1211
13. Borin, L., Dannélls, D., Forsberg, M., Toporowska Gronostaj, M., Kokkinakis, D.: The past meets the present in Swedish FrameNet++. In: *Proceedings of the 14th EURALEX International Congress*. (2010) 269–281
14. Fillmore, C.J., Johnson, C.R., Petruck, M.R.L.: Background to Framenet. *International Journal of Lexicography* **16**(3) (2003) 235–250

¹² <<http://www.clt.se>>

QASM: a Q&A Social Media System Based on Social Semantic

Zide Meng¹, Fabien Gandon¹, and Catherine Faron-Zucker²

INRIA Sophia Antipolis Méditerranée, 06900 Sophia Antipolis, France¹
Univ. Nice Sophia Antipolis, CNRS, I3S, UMR 7271, 06900 Sophia Antipolis, France²

Abstract. In this paper, we describe the QASM (Question & Answer Social Media) system based on social network analysis to manage the two main resources in CQA sites: users and contents. We first present the QASM vocabulary used to formalize both the level of interest and the expertise of users on topics. Then we present our method to extract this knowledge from CQA sites. Finally we show how this knowledge is used both to find relevant experts for a question and to search for similar questions. We tested QASM on a dataset extracted from the popular CQA site StackOverflow.

Keywords: Community Question Answering, Social Media Mining, Semantic Web

1 Introduction

Community Question Answering (CQA) services provide a platform where users can ask expert for help. Since questions and answers can be viewed and searched afterwards, people with similar questions can also directly find solutions by browsing this content. Therefore, effectively managing these content is a key issue. Previous research works on this topic mainly focus on expert detection [2], similar question retrieval [1]. In this paper, we describe QASM (Question & Answer Social Media), a system based on social network analysis (SNA) to manage the two main resources in CQA sites: users and contents. We first present the QASM vocabulary used to formalize both the level of interest and the expertise of users on topics. Then we present our method to extract this knowledge from CQA sites. Our knowledge model and knowledge extraction method is an extension of our work presented in [3] on social media mining for detecting topics from question tags in CQA sites. Finally we show how this knowledge is used both to find relevant experts for routing questions (users interested and experts in the question topics) and to find answers to questions by browsing CQA content and by identifying relevant answers to similar questions previously posted. We tested QASM on a dataset extracted from the popular CQA site StackOverflow.

2 QASM System Description

2.1 Overview

Figure 1 presents an overview of QASM. We first use the SIOC ontology¹ to construct an RDF dataset from social media data extracted from a CQA site. Then we use social media mining techniques to extract topics, interests and expertise levels from this dataset. We formalize them with the QASM schema and enrich our RDF dataset with this knowledge. As a result, we provide an integrated and enriched Q&A triple store which contains both user interests, levels of expertise and topics learned from question tags. Finally, we linked our dataset with DBpedia (through named entity identification).

Based on the QASM RDF dataset, we can provide the users of the Q&A site with two services to find relevant experts for a question and to search for similar questions. We detail them in the following subsections.

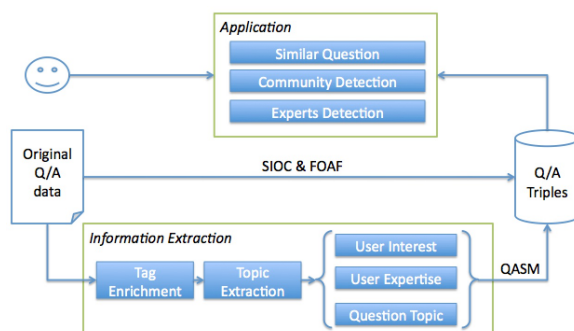


Fig. 1. Overview of QASM

2.2 QASM Vocabulary

The QASM vocabulary² enables to model the level of user interests and expertise and topics of questions and answers from Q&A sites. Figure 2 provides an overview of it. It reuses both the SIOC ontology and the Weighting ontology³.

- `qasm:Topic` represents a set of tags related to a specified topic. In our models, tags belong to instances of `qasm:Topic`, we also consider different tags have different weights for each topic.

¹ <http://sioc-project.org/ontology>

² It is available online at <http://ns.inria.fr/qasm/qasm.html>

³ <http://smy.sourceforge.net/wo/spec/weightingontology.html>

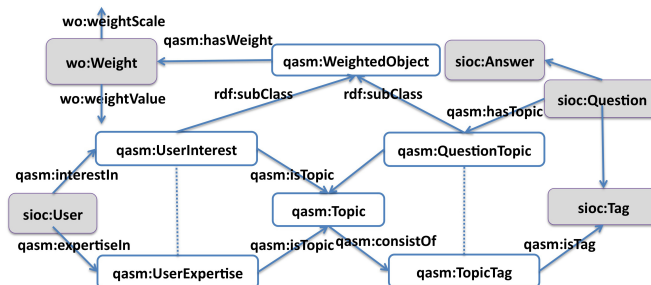


Fig. 2. Overview of the QASM vocabulary

- `qasm:WeightedObject` is used to describe the weight that a specified subject has with regard to a specified object. This class has four subclasses which represent question topics, users’ interests, users’ expertise and tag topics respectively. In fact, this class is used to model the distributions we extracted from the original data. For example, topic-tag distribution, user-interest distribution.
- `qasm:interestIn` is used to describe the user-interest distribution. This property is different from `foaf:interest` for its range. In FOAF people are interested in documents, while in QASM a user is interested in a topic to a certain degree (a weight).
- `qasm:expertiseIn` is used to describe the user-expertise distribution. A user has different weights for different topics.

2.3 Knowledge Extraction by Social Media Mining

Topics, interests and levels of expertise are implicit information in the available raw CQA data. We use social media mining techniques to extract this knowledge.

- Topics & User Interests In [3], we proposed a light-weight model to extract topics from question tags. The output of this model is a topic-tag distribution where each tag belonging to a topic is given a weight (probability) indicating to what extent the tag is related to the topic. A user answering a question acquires the tags attached to this question and can therefore be represented by a list of tags. Then we use the topic-tag distribution to compute a user-topic distribution indicating to what extent each user is related to a topic.
- User Expertise The users interested in a question may provide answers to it or comments to other answers. Each question or answer may get votes from other users and an answer may be chosen as the best answer. By exploiting the tags attached to a question and the topic-tag distribution, the users providing questions or answers with a high number of votes or the best answers can be considered as experts in the topics to which their questions belongs. Equation 1 defines how we use the vote information to compute users’ levels of expertise. $E_{u,k}$ denotes the expertise of user u on topic k , m denotes the number of answers provided by user u , $P_{i,k}$ denotes the weight

of tag t for topic k , Q_i and $A_{i,j}$ denote the votes on question i and its j^{th} answer, where A_j is the j^{th} answer provided by user u to question Q_i .

$$E_{u,k} = \sum_{i=1}^m P_{t,k} * \log(Q_i) * \log(A_{i,j}) \quad (1)$$

2.4 Experimental Evaluation

We first built an RDF dataset from Stackoverflow raw data which comprises 15327727 triples⁴. Then we randomly chose several questions and for each question we recorded 10 or 20 users provided by our system. Then for each question, we computed the proportion of the recorded users who actually answered it. Compared to [4], our results are much better.

Table 1. Preliminary results on question routing

	100	500	1000	average	[4]
precision@10	0.021	0.0188	0.0187	0.0195	0.0167
precision@20	0.016	0.0134	0.0134	0.0143	0.0118

3 Conclusion and Future Work

We presented QASM, a Q&A system combining social media mining and semantic web models and technologies to manage Q&A users and content in CQA sites. There are many potential future directions for this work. We are currently considering constructing a benchmark for Q&A system based on our Stackoverflow dataset. In a near future we will also enrich the linking of QASM with the LOD which may help to improve question routing and similar question search.

References

1. Anderson, A., Huttenlocher, D., Kleinberg, J., Leskovec, J.: Discovering value from community activity on focused question answering sites: a case study of stack overflow. In Proc. of the 18th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining (2012)
2. Yang, L., Qiu, M., Gottipati, S., Zhu, F., Jiang, J., Sun, H., Chen, Z.: CQArank: jointly model topics and expertise in community question answering. In Proc. of the 22nd ACM Int. Conf. on Information & Knowledge Management (2013)
3. Zide, M., Fabien, G., Catherine, F., Ge, S.: Empirical Study on Overlapping Community Detection in Question and Answer Sites. In Proc. of the Int. Conf. on Advances in Social Networks Analysis and Mining (2014)
4. Chang, S., Pal, A.: Routing questions for collaborative answering in community question answering. In Proc. of Int. Conf. on Advances in Social Networks Analysis and Mining (ASONAM) (2013)

⁴ It is available online at <https://wimmics.inria.fr/data>

A Semantic-Based Platform for Efficient Online Communication

Zaenal Akbar, José María García, Ioan Toma, Dieter Fensel

Semantic Technology Institute, University of Innsbruck, Austria
`firstname.lastname@sti2.at`

Abstract. To achieve an effective and efficient way of disseminating information to ever growing communication channels, we propose an approach that separates the information and communication channels and interlinks them with an intermediary component. The separation enables various dimensions to reuse the information and communication channels in transactional communication. In this paper we introduce our online communication platform, which is comprised of several components. The important roles of semantic web technologies to the platform are explained in detail, including a use case to show the contributions of semantic web in supporting the effectiveness and efficiency of information dissemination.

Keywords: semantic web, online communication, platform, information dissemination

1 Introduction

In today's internet era, the number and kinds of information dissemination channels are growing exponentially and changing constantly. Websites, e-mails, blogs and social media have become the mainstream means of communication. Nevertheless, information dissemination is not only about finding suitable channels, but also fitting the content to the available channels. These are the main challenges for effective and efficient information dissemination, and for online communication in general.

Our solution to overcoming these challenges is to decouple information from channels, defining separate models for each of them, and then interlinking them with an intermediary component [1]. Semantic technologies play important roles in our solution: analysis and understanding of the natural language statements, information modeling and sharing with common vocabularies, matchmaking information and channels using a rules-based approach [2].

In this paper, we focus on the information modeling (including annotations) part such that the matchmaking of information to appropriate channels can be performed efficiently. First, we present the overall architecture, then we discuss how semantics contribute to the solution and finally we show a use case, followed by the conclusion and future works.

2 The Online Communication Platform

Shown in Fig. 1, the online communication platform consists of several components which are grouped based on their conceptual functions:

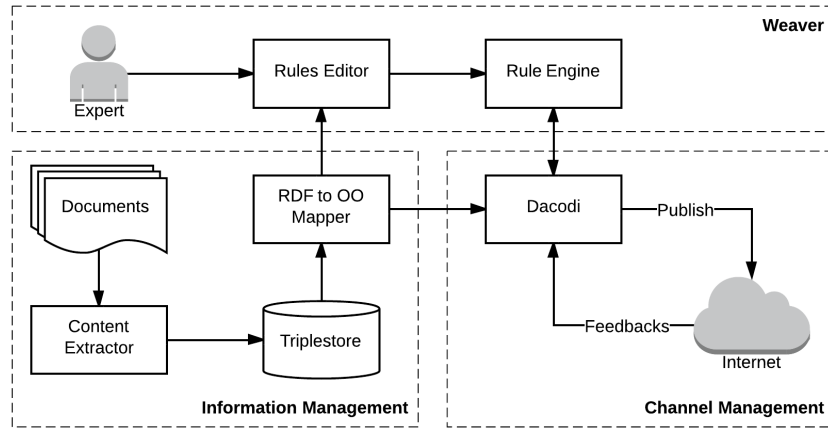


Fig. 1. The Online Communication platform architecture

Information Management is responsible for gathering the content from data sources (annotated and un-annotated) and representing them into the common vocabularies. First, the contents are extracted by a *Content Extractor* (implemented using Any23¹), then stored onto a *Triplestore* such as OWLIM². Further, an *RDF to OO Mapper* (implemented using RDFBeans³) maps the stored triples onto object-oriented models to be used by the other components. For annotated sources where the sources have been annotated with the selected vocabularies, the content can be extracted automatically. For un-annotated sources, a manual mapping is required to inter-relate the database items (i.e. table fields) to relevant terms in the desired vocabularies.

Weaver is responsible for matching the information to appropriate channels through a rule based system. A *Rule Editor* enables experts to create and maintain rules through an integrated user interface and access-controlled rules repository. The rules are then matched to the facts in the working memory of the rule-based system by a *Rule Engine*. In our implementation we use Drools⁴.

Channel Management is responsible for distributing the information to the selected channels according to the defined rules. *Dacodi*⁵ offers various functionalities for distributing the content to the selected communication channels, as well as for collecting and analyzing feedback from those channels [3].

¹<http://any23.apache.org>

²<http://www.ontotext.com/owlim>

³<http://rdfbeans.sourceforge.net>

⁴<http://drools.jboss.org>

⁵<http://dacodi.sti2.at>

3 Applying Semantic Technologies to Online Communication

Semantic technologies contribute mainly to content modeling, namely in how to obtain content from distributed and heterogeneous sources (i.e. through annotation) and represent them in a common representation to make an efficient match between information and desired channels possible. The matching is not between content sources but between the common representation to channels.

To achieve a reusable and interoperable information model, we selected vocabularies (whole or partial) from the Linked Open Vocabularies ⁶:

1. Dublin Core ⁷, all metadata terms to support resource description
2. Friend of a Friend ⁸, a vocabulary to describe people, the links between them, the things they create and do
3. Good Relations ⁹, a vocabulary to describe e-commerce products and services
4. Schema.org ¹⁰, a collection of tags to markup a page in ways recognized by major search engines

These vocabularies are widely used, especially Schema.org which has been adopted by webmasters to increase their webpages' visibility in search engines.

We show these contributions in detail within the Tourismusverband (TVb) Innsbruck ¹¹ use case. As one of the big tourism boards in Austria, its goal is to achieve the highest visibility possible in search engines as well as to be present in various social channels [4]. It has a lot of content types (i.e. Place, Event, Trip) to be disseminated to numerous channels (i.e. Facebook, YouTube).

- a) The TVb Innsbruck content sources (i.e. Blog, services from touristic providers) were annotated with the selected terms of Schema.org.

```
<div itemscope itemtype="http://schema.org/Event">
  <span itemprop="name">Farmer's Market</span>
  <div itemprop="startDate" datetime="2014-07-14T07:00">
    14.07.2014 - 01.01.2015</div>
  <time itemprop="endDate" datetime="2015-01-01T12:00"/>
  <span itemprop="location" itemscope
    itemtype="http://schema.org/PostalAddress">Location:
    <span itemprop="streetAddress" content="Markthalle">
      Markthalle (Herzog-Siegmond-Ufer 1-3, AT-6020, Innsbruck)</span>
    <meta itemprop="streetAddress" content="Herzog-Siegmond-Ufer 1-3"/>
    <meta itemprop="addressRegion" content="Innsbruck"/>
    <meta itemprop="postalCode" content="6020"/>
    <meta itemprop="addressCountry" content="AT"/></span>
</div>
```

In this example, information about *Event* is annotated with the term *Event* from Schema.org by using microdata format ¹².

⁶<http://lov.okfn.org>

⁷<http://dublincore.org>

⁸<http://www.foaf-project.org>

⁹<http://purl.org/goodrelations/>

¹⁰<http://schema.org>

¹¹<http://www.innsbruck.info>

¹²<http://www.w3.org/TR/microdata/>

- b) The publication rules were defined to guide the publication of extracted contents to selected channels.

```
rule "Event Publication Rule"
when item : Event()
then insert(new ItemToBePublishedIn(item, facebookWall))
      insert(new ItemToBePublishedIn(item, youtube))
end
```

In this rule, each time a new *Event* was found in the extracted contents, it was then prepared to be published to the `facebookWall`, `youtube` (instances of TVb's Facebook and YouTube accounts respectively).

4 Evaluation, Conclusions and Future Work

In order to evaluate our work, we compared the number of visitors to the TVb's website before and after annotating the content. Compared to the same period in 2013, the number of visitors increased by 8.63% between Jan-Feb 2014, which may be caused by the annotation. Also, the platform is currently being tested by 6 people at TVb as a substitution to their social media dissemination tool.

The platform was comprised of several components and used semantic web technologies to integrate various information sources, extracting and representing the content into common vocabularies to enable efficient matchmaking to appropriate channels using a rules-based approach. There are four vocabularies currently supported and in the future, we would like to add more vocabularies (i.e. Schema.org Action, SIOC¹³) to enhance the channel management, in order to improve the feedback collection, for example.

Acknowledgements We would like to thank all members of the OC working group¹⁴ for their valuable feedback. This work was partly funded by the EU FP7 under grants no. 600663 (Previda), 257641 (PlanetData) and 284860 (MSEE).

References

1. Fensel, A., Toma, I., García, J.M., Stavrakantonakis, I., Fensel, D.: Enabling customers engagement and collaboration for small and medium-sized enterprises in ubiquitous multi-channel ecosystems. *Computers in Industry* **65**(5) (2014) 891–904
2. Akbar, Z., García, J.M., Toma, I., Fensel, D.: On using semantically-aware rules for efficient online communication. In Bikakis, A., Fodor, P., Roman, D., eds.: *Rules on the web. From theory to applications*. Volume 8620 of LNCS. Springer (2014) 37–51
3. Toma, I., Fensel, D., Oberhauser, A., Fuchs, C., Stanciu, C., Larizgoitia, I.: Sesa: A scalable multi-channel communication and booking solution for e-commerce in the tourism domain. In: *The 10th International Conference on e-Business Engineering (ICEBE)*. (Sept 2013) 288–293
4. Akbar, Z., Fensel, A., Fensel, D., Fuchs, C., Garcia, J., Juen, B., Lasierra, N., Stanciu, C.V., Toma, I., Tymaniuk, S.: *Tvb innsbruck semantic pilot analysis*. White paper, Semantic Technology Institute, University of Innsbruck (May 2014) <http://oc.sti2.at/TR/TVBInnsbruck>.

¹³<http://sioc-project.org>

¹⁴<http://oc.sti2.at>

SHAX: A Semantic Historical Archive eXplorer

Michael Feldman¹, Shen Gao¹, Marc Novel², Katerina Papaioannou¹, and
Abraham Bernstein¹

¹ Department of Informatics, University of Zurich, Zurich, Switzerland

² Swiss Federal Research Institute WSL, Birmensdorf, Switzerland

Abstract. Newspaper archives are some of the richest historical document collections. Their study is, however, very tedious: one needs to physically visit the archives, search through reams of old, very fragile paper, and manually assemble cross-references. We present SHAX, a visual newspaper-archive exploration tool that takes large, historical archives as an input and allows interested parties to browse the information included in a chronological or geographic manner so as to re-discover history.

We used SHAX on a selection of the *Neue Zürcher Zeitung* (NZZ)—the longest continuously published German newspaper in Switzerland with archives going back to 1780. Specifically, we took the highly noisy OCRed text segments, extracted pertinent entities, geolocation, as well as temporal information, linked them with the Linked Open Data cloud, and built a browser-based exploration platform.

This platform enables users to interactively browse the 111906 newspaper pages published from 1910 to 1920 and containing historic events such as World War I (WWI) and the Russian Revolution. Note that SHAX is neither limited to this newspaper nor to this time-period or language but exemplifies the power in combining semantic technologies with an exceptional dataset.

1 Introduction

During the past decade, many newspapers (most notably the New York Times³ but see also [1] for an overview) have digitalized their archive in order to make it searchable and publicly available. Usually, the scanned newspapers are converted into text via the use of Optical Character Recognition (OCR). The received output contains a great degree of noise and makes knowledge discovery from historical newspaper archives a challenging task. Approaches like data cleaning with specialized Information Retrieval (IR) tools are commonly used for this task but require substantial human involvement and domain-specific knowledge [4].

Alternatively, we develop a Semantic-Web based, data-driven approach, which effectively retrieves information from a large volume of newspaper issues. Our methodology was applied to a part of the digitalized archive of the *Neue Zürcher*

³ <http://open.blogs.nytimes.com/2013/07/11/introducing-the-new-timesmachine/>

Zeitung (NZZ) for the years ranging from 1910 to 1920 and is applicable to different news corpora in various languages. The interactive visualization of our results enables the user to browse and discover historical events with the related geographic information.

2 Dataset

The NZZ kindly provided us with a part of the archive covering the issues published from 1910 to 1920. This period covers historic events such as WWI and the Russian Revolution. The scanning, digitizing, and OCRing of the NZZ archive was conducted by the Fraunhofer Institute⁴. The dataset we use consists of 354 GB scanned PDFs and 111906 OCRed pages in XML format (one XML file per newspaper page).

One of the biggest problems when processing the data is noise. The OCR struggled with the Gothic font, which is used during the longest period in the archive, including the one under discussion. Additionally, during wartime, when printing resources were scarce, ink and paper quality decreased: some pages are simply not readable and others were printed on thin paper, causing the text of the backside to shimmer through the front side in the scans. The recognized text also contains unavoidable errors, such as different word-spelling due to language change. However, the names of high-frequency entities remain the same during this time period. These errors cannot affect our results considerably.

As a result only a part of the text was correctly recognized. Using a spell-checker, we found that only 64% of the words were correctly recognized. As we assume a random distribution of the errors, our results contain insignificant biases.

3 The Application

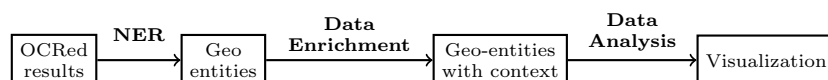


Fig. 1: The Semantic-Web based approach for analysing newspaper corpora

3.1 The Semantic Web Approach

Existing ways of dealing with historical corpora rely on Information Retrieval methods requiring a substantial amount of human effort. Based on the assumption that the locations of important historical events are explicitly mentioned in the newspaper, we develop a purely data-driven approach that leverages Semantic Web technologies to analyze the noisy dataset (Fig. 1).

Specifically, we first perform the Named Entity Recognition (NER) on the dataset with DBpedia Spotlight [3], trying out different confidence values to

⁴ <https://www.iais.fraunhofer.de/nzz.html>

improve the accuracy. We focus on the correlation between temporal and geographical information, hence, we only extract the geographic entities (e.g., the city of Sarajevo). Each entity is linked with its corresponding meta-data as well as information retrieved from DBpedia and GeoNames. For example, we query the longitude and latitude from DBpedia and use GeoNames to find its country code by reverse geo-indexing. The result of this process includes tuples in the following format: (entity name, longitude, latitude, country code, DBpedia link, date of mention, issue ID). Finally, we perform data analysis on the results on a monthly basis by aggregating on the country code or the entity name. In both cases, we compute the sum of counts in every group.

3.2 The Interactive Visualization

In this section, we briefly introduce the functions of our exploration platform which is available at <https://files.ifl.uzh.ch/ddis/nzz-demo/WebContent/>.

Function 1: Country Mentions over time A choropleth-map of Europe was generated for each year based on the country counts. As shown in Fig. 2(a) and 2(b), the color intensity of a country is in proportion to its counts (i.e. the darker the color, the more the counts). By navigating through the years, the way the colors change provides an overview of the popularity of each country. For example, the Balkan countries are mentioned more often at the beginning of WWI. In order to avoid biases, such as countries being mentioned extensively due to higher geographical proximity to Switzerland or due to larger population, the annual counts of each country were also normalized by relative distance ([5]) to Zürich and population estimated in 1910.

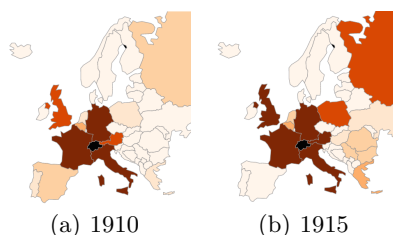


Fig. 2: Color Changing of EU in 1910 and 1915

Function 2: Linking countries, issues and historical events By constructing an inverse index that links the countries to the issues where they are mentioned, users can further explore the reasons behind the change in the colors. By clicking on a country, they can see the historical events it was involved in Fig. 3 as well as the relevant newspaper’s PDFs. The historical events presented are systematically extracted from DBpedia [2] by querying the category “Event” with the corresponding country. A newspaper’s PDF is considered relevant if the country or a place within its borders was mentioned.

Function 3: Entity Mentions over time A more detailed analysis of entity mentions is visualized using the word cloud (Fig. 4(a)) and trend line (Fig.

Date	Event	Wikipedia Link
13-3-1915	Battle of Neuve Chapelle	http://en.wikipedia.org/wiki/Battle_of_Neuve_Chapelle
9-5-1915	Battle of Aubers Ridge	http://en.wikipedia.org/wiki/Battle_of_Aubers_Ridge
27-5-1915	Battle of Festubert	http://en.wikipedia.org/wiki/Battle_of_Festubert
14-10-1915	Battle of Loos	http://en.wikipedia.org/wiki/Battle_of_Loos
4-11-1915	Third Battle of Artois	http://en.wikipedia.org/wiki/Third_Battle_of_Artois
6-11-1915	Second Battle of Champagne	http://en.wikipedia.org/wiki/Second_Battle_of_Champagne

Fig. 3: Linking countries, issues and historical events

4(b)) of all geographic entities per year. It is possible to directly observe the changes in the popularity of each entity over time, as well as correlations among them. Additionally, we plot each entity as a bubble on the map based on its coordinates and number of mentions (Fig. 4(c)). Thus, users could discover the actual location of a historical event.

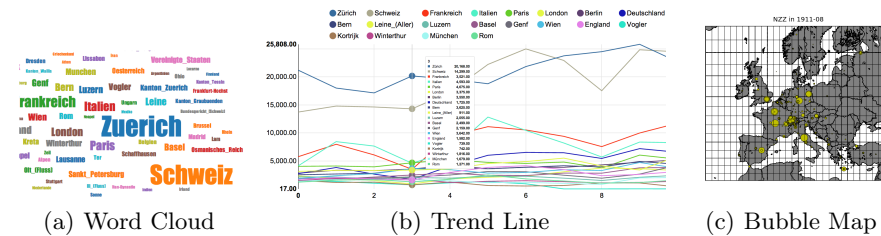


Fig. 4: Entity Mentions over time

4 Conclusion

SHAX is a browser-based exploration platform, which highlights the major role of Semantic Web tools in extracting entities mentioned in highly noisy newspaper archives. Moreover, it shows that interactive visualization is necessary not only in presenting the information within the newspaper in a user-friendly way, but also in discovering implicit knowledge from the corpus. In the future, we plan to apply our method on the whole 250 years of NZZ archives and try to extend it to explore other kinds of entities such as notable people.

Acknowledgements

We would like to thank the Neue Zürcher Zeitung, Thilo Haas, Thomas Scharrenbach, and Daniel Spicar.

References

- [1] Doerr, M., Markakis, G., Theodoridou, M., Tsikritzis, M.: Diathesis: Ocr based semantic annotation of newspapers (2007)
- [2] Hienert, D., Luciano, F.: Extraction of historical events from wikipedia. In: CoRR (2012)
- [3] Mendes, P.N., Jakob, M., García-Silva, A., Bizer, C.: Dbpedia spotlight: shedding light on the web of documents. In: Proc. of the 7th i-Semantics (2011)
- [4] Places, Á., Fariña, A., Luaces, M., Pedreira, Ó., Seco, D.: A workflow management system to feed digital libraries: proposal and case study. Multimedia Tools and Applications (2014)
- [5] Worboys, M.F.: Metrics and topologies for geographic space. In: Proc. 7th Intl. Symp. Spatial Data Handling (1996)

SemanTex: Semantic Text Exploration Using Document Links Implied by Conceptual Networks Extracted from the Texts*

Suad Aldarra¹, Emir Muñoz¹, Pierre-Yves Vandenbussche¹, and Vít Nováček²

¹ Fujitsu (Ireland) Limited

Airside Business Park, Swords, Co. Dublin, Ireland

E-mail: `Firstname.Lastname@ie.fujitsu.com`

² Insight @ NUI Galway (formerly known as DERI)

IDA Business Park, Lower Dangan, Galway, Ireland

E-mail: `vit.novacek@deri.org`

1 Introduction

Despite of advances in digital document processing, exploration of implicit relationships within large amounts of textual resources can still be daunting. This is partly due to the ‘black-box’ nature of most current methods for computing links (*i.e.*, similarities) between documents (*c.f.*, [1] and [2]). The methods are mostly based on numeric computational models like vector spaces or probabilistic classifiers. Such models may perform well according to standard IR evaluation methodologies, but can be sub-optimal in applications aimed at end users due to the difficulties in interpreting the results and their provenance [3, 1].

Our Semantic Text Exploration prototype (abbreviated as SemanTex) aims at finding implicit links within a corpus of textual resources (such as articles or web pages) and exposing them to users in an intuitive front-end. We discover the links by: (1) finding concepts that are important in the corpus; (2) computing relationships between the concepts; (3) using the relationships for finding links between the texts. The links are annotated with the concepts from which the particular connection was computed. Apart of being presented to human users for manual exploration in the SemanTex interfaces, we are working on representing the semantically annotated links between textual documents in RDF and exposing the resulting datasets for particular domains (such as PubMed or New York Times articles) as a part of the Linked Open Data cloud.

In the following we provide more details on the method and give an example of its practical application to browsing of biomedical articles. A video example of a specific SemanTex prototype to be demonstrated at the conference can be looked up at <http://goo.gl/zL81J2>.

* This work has been supported by the ‘KI2NA’ project funded by Fujitsu Laboratories Limited in collaboration with Insight @ NUI Galway.

2 Method

Extracting Conceptual Networks. For extracting links between concepts in the texts we use methods we introduced in [4]. The essentials of the method are as follows: (1) Extracting noun phrases that may refer to domain-specific concepts (using either a shallow parser for general texts or biomedical named-entity recognition tool for life sciences). (2) Computing co-occurrence relationships between the extracted noun phrases by means of point-wise mutual information (PMI). (3) Filtering out the relationships with the PMI scores below a threshold. (4) Computing (*cosine*) similarity relationships based on the co-occurrence ones.

Computing Paths between Documents. From a conceptual network, one can generate sets of paths leading out from every concept. To prevent a combinatorial explosion, we limit the paths by two factors: (1) the maximum path length; (2) the minimum product of the edge weights of the path. From the set of such paths associated with particular nodes, paths between the original documents (*i.e.*, text-to-text links semantically annotated by the concepts appearing on them) can be generated using inverted indices of concept-text provenance. For instance, imagine a text A contains concept x . Now assume that x is related to a concept y in text B via a path (x, u, v, y) . Then we can say the texts A and B are related by a (x, u, v, y) path.

Selecting the Most Relevant Paths. The critical part of the method is finding out which paths are most promising out of potentially huge numbers of them. For that we use multi-objective optimisation of several specific complexity, coherence and entropy measures introduced in [4]. We follow certain intuitive assumptions when selecting the path measures to optimise: (1) Paths leading through more complex environs are more informative for a user. (2) Paths surrounded by many highly balanced (*i.e.*, entropic) topics are more informative. (3) Coherent paths with gradual topical changes on the way are better (less chaotic, more focused progression from one topic to another en route to the linked text). (4) It is more interesting, considering an Information Retrieval point of view, when one ends up in a topically distant (incoherent) area (once the progress through the topics is gradual, *i.e.*, less random). The result of this step is a set of optimal (non-dominated) text-to-text paths that can be further ranked according to their combined score.

3 Usage

To demonstrate the SemanTex technology, we have applied it to the corpus of Parkinson's Disease article abstracts from PubMed which we experimented with in [4]. As can be seen in Figure 1, the front-end has been incorporated into a PubMed look-and-feel. Domain-specific concepts have been highlighted in the abstract display (the darker the shade, the more the concept is important for the given abstract). After clicking on any of the highlights, a separate 'Path Diagram' window is displayed where one can navigate paths leading from the selected concept. The nodes on the paths can be expanded with further connections

while the corresponding related articles are always displayed in the bottom of the window. Clicking on a related article leads to the article view. One can also explore articles related by a path to the currently browsed one. A diagram of paths that connect the articles via the concepts in them can be displayed as well.

Figure 1 illustrates SemanTex on an example of article about the correlation of caffeine consumption, risk of Parkinson’s disease and the related differences between men and women. When exploring the concept ‘*high caffeine consumption*’, one can continue to the ‘*hormones*’ and ‘*women*’ nodes. Expanding the ‘*women*’ link shows many concepts related to women’s health, such as ‘*oophorectomy*’ (removal of ovaries). There is a single article related to that concept, dealing with increased risk of parkinsonism in women who underwent oophorectomy before menopause. This shows how one can quickly explore a problem from many different viewpoints with SemanTex, linking an article dealing with the influence of particular hormonal levels on the development of Parkinson’s Disease in women with another article looking into higher risk of parkinsonism due to lower levels of estrogen caused by the pre-menopausal removal of ovaries.

When further exploring some articles related to the last one, one can see all the paths that connect them. For instance, a study about Dutch elderly people is linked to the oophorectomy article by means of four paths, all involving concepts clearly related to common geriatric ailments. This illustrates the possibility of smooth topical progression in exploring the articles.

4 Conclusions

In this work, we have presented SemanTex, an application that discovers implicit, semantically annotated links within a corpus of textual resources. We have implemented a sample prototype of the technology deployed on PubMed articles using the standard PubMed look-and-feel. This was to show how we can easily add value to many traditional applications involving exploration of large numbers of textual resources. In a similar way, we will implement SemanTex versions for New York Times and Wikipedia articles within an evaluation trial of the technology. Last but not least, we plan to generate RDF representations of the semantically annotated links between texts computed by particular instances of SemanTex and expose them as a part of the LOD cloud.

References

1. Manning, C.D., Raghavan, P., Schütze, H.: Introduction to Information Retrieval. Cambridge University Press (2008)
2. Lin, J., Wilbur, W.J.: PubMed related articles: a probabilistic topic-based model for content similarity. *BMC Bioinformatics* **8**(1) (2007)
3. Grefenstette, E.: Analysing document similarity measures. Master’s thesis, University of Oxford (2009)
4. Nováček, V., Burns, G.A.: SKIMMR: Facilitating knowledge discovery in life sciences by machine-aided skim reading. *PeerJ* (2014) In press, see <https://peerj.com/preprints/352/> for a preprint.

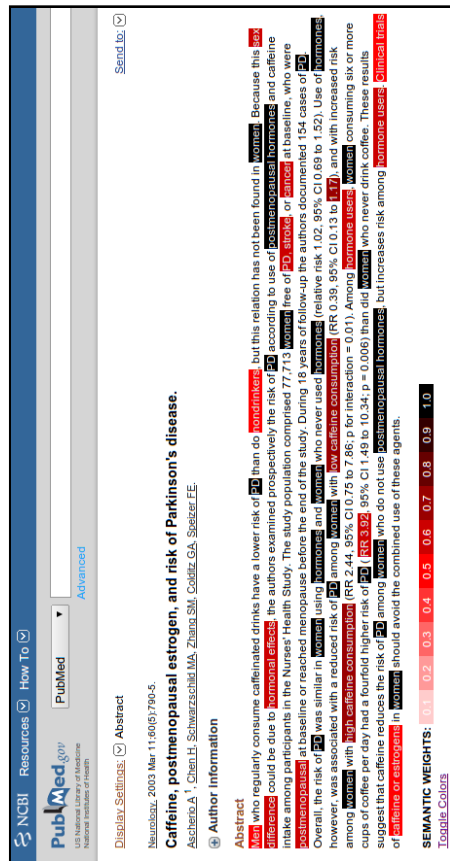
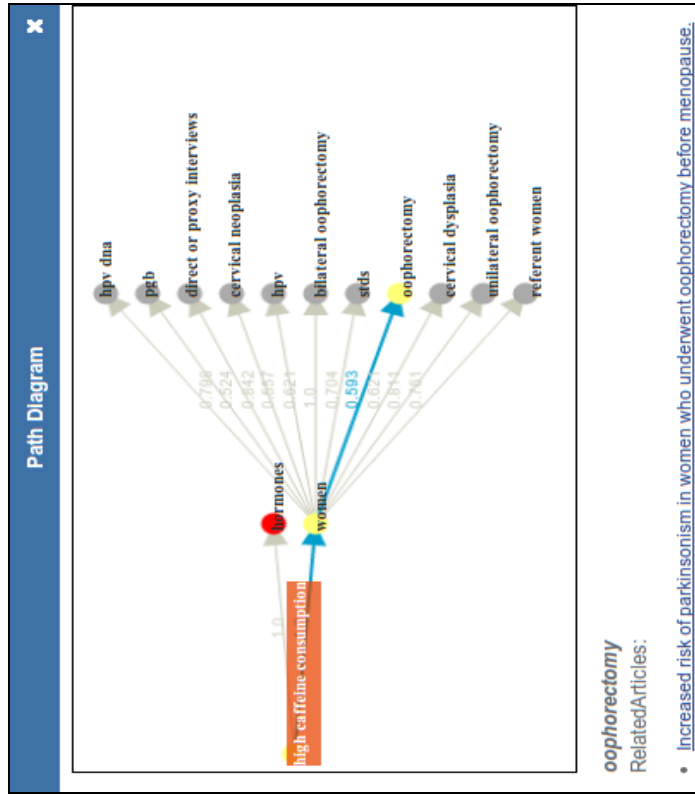


Fig. 1. SemanTex usage example - Parkinson's Disease

Towards a Top-K SPARQL Query Benchmark

Shima Zahmatkesh, Emanuele Della Valle, Daniele Dell’Aglio, and Alessandro Bozzon

DEIB - Politecnico of Milano, Delft University of Technology
shima.zahmatkesh@polimi.it, emanuele.dellavalle@polimi.it,
daniele.dellaglio@polimi.it, a.bozzon@tudelft.nl

Abstract. The research on optimization of top-k SPARQL query would largely benefit from the establishment of a benchmark that allows comparing different approaches. For such a benchmark to be meaningful, at least two requirements should hold: 1) the benchmark should resemble reality as much as possible, and 2) it should stress the features of the top-k SPARQL queries both from a syntactic and performance perspective. In this paper we propose Top-k DBPSB: an extension of the DBpedia SPARQL benchmark (DBPSB), a benchmark known to resemble reality, with the capabilities required to compare SPARQL engines on top-k queries.

Keywords: Top-k Query, SPARQL, Benchmark

1 Problem Statement

Top-k queries – queries returning the top k results ordered according to a user-defined scoring function – are gaining attention in the Database [1] and Semantic Web communities [2–6]. Order is an important property that can be exploited to speed up query processing, but state-of-the-art SPARQL engines such as Virtuoso [7], Sesame [8], and Jena [9], do not exploit order for query optimisation purposes. Top-k SPARQL queries are managed with a *materialize-then-sort* processing schema [1] that computes all the matching solutions (e.g., thousands) even if only a limited number k (e.g., ten) are requested.

Recent works [2–5] have shown that an efficient *split-and-interleave* processing schema [1] could be adopted to improve the performance of top-k SPARQL queries. To the best of our knowledge, a consistent comparison of those works does not exist. As often occurs, the main cause for this fragmentation resides in the partial lack of a SPARQL benchmark covering top-k SPARQL queries. To foster the work on top-k query processing within the Semantic Web community, we believe that it is the right time to define a top-k SPARQL benchmark.

Following well known principles of benchmarking [10], we formulate the research question of this work as: *is it possible to set up a benchmark for top-k SPARQL queries, which resembles reality as much as possible and stresses the features of top-k queries both from a syntactic (i.e., queries should contain rank-related clauses) and performance (i.e., the query mix should insist on characteristics of top-k queries which stress SPARQL engine) perspective?*

2 Methodology

In this poster, we describe our ongoing work on Top-k DBpedia SPARQL Benchmark (Top-k DBPSB). It extends the methodology, proposed in DBPSB [11], to build SPARQL benchmarks that resemble reality. It uses the same dataset, performance metrics, and test driver of DBPSB, but it extends the *query variability* feature of DBPSB by proposing an algorithm to automatically create top-k queries from the 25 auxiliary queries of the DBPSB and its datasets.

In order to present Top-k DBPSB, we need to introduce some terminology:

- **Definition 1:** *Rankable Data Property* is a RDF property whose range is in `xsd:int`, `xsd:long`, `xsd:float`, `xsd:integer`, `xsd:decimal`, `xsd:double`, `xsd:date`, `xsd:dateTime`, `xsd:time`, or `xsd:duration`.
- **Definition 2:** *Rankable Triple Pattern* is a triple pattern that has a Rankable Data Property in the property position of the pattern.
- **Definition 3:** When a variable, in the object position of a Rankable Triple Pattern, appears in a scoring criteria of the scoring function, we call it *Scoring Variable* and we call *Rankable Variable* the one appearing in the subject position.

Figure 1 shows an overview of the process followed by Top-k DBPSB to generate top-k SPARQL queries from the Auxiliary Queries and datasets of DBSBM. The process consists of four steps: 1) finding rankable variables, 2) computing maximum and minimum values for each rankable variable, 3) generating scoring functions, and 4) generating top-k queries.

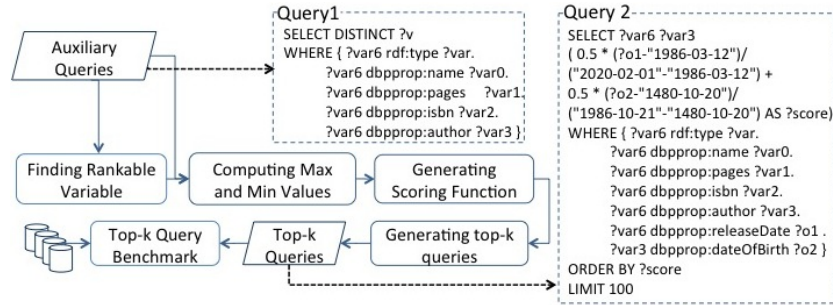


Fig. 1: Top-k DBPSB generates top-k SPARQL queries starting from the Auxiliary queries and datasets of DBPSB (a SPARQL benchmark known to resemble reality).

In the first step, Top-k DBPSB looks for all rankable variables, in each auxiliary query of each DBPSB query template, which could be used as part of a ranking criterion in a scoring function. For each DBPSB auxiliary query, Top-k DBPSB first checks if the variables in the query fit the definition of scoring variable. To find additional rankable variable Top-k DBPSB considers all variables in the query and tries also to find rankable triple pattern related to them.

For the sake of simplicity, Top-k DBPSB generates scoring function as a weighted sum of normalized ranking criteria, therefore, for each rankable variable, Top-k DBPSB needs to compute its maximum and minimum value. So, for each DBPSB auxiliary query and each rankable triple pattern identified in the previous step, Top-k DBPSB generates a query to find those values.

After having collected those values, for each rankable triple pattern, Top-k DBPSB creates a new top-k query template. For instance, Query 2 of Figure 1 shows such a query as generated by the Top-K DBPSB. In order to obtain an executable query, for each scoring variable that appears in scoring function Top-K DBPSB adds the related rankable triple pattern to the body of the query.¹

3 Experiments

Given that we extended DBPSB we give for positively answered the first part of our research question (i.e., Top-k DBPSB resembles reality). In this section, we provide preliminary evidence that the query variability feature of Top-k DBPSB positively answers also the second part of our research question. We do so by operationalising our research question in three hypotheses:

- H.1 The more are the number of the Rankable Variables, the longer is the average execution time.
- H.2 The more are the number of the Scoring Variables in the scoring function, the longer is the average execution time.
- H.3 The value of the LIMIT clause has not any significant impact on the average execution time.

In order to evaluate our hypothesis, we carry out our experiments by using the SPARQL engines Virtuoso, and Jena. After preparing the experimental environment, we load the DBpedia dataset with the scale factors of 10% on the SPARQL engines. In order to evaluate the performance of SPARQL engines, we use the DBpedia SPARQL Benchmark test driver and modify it for Top-k queries. We execute the generated Top-k SPARQL queries against these two SPARQL engines. The information gains from the execution of randomly generated query against the SPARQL engines is used to evaluate our hypotheses.

As a result we got that most of the queries templates generated by Top-k DBPSB are compatible with our hypotheses H.2 and H.3. On the contrary, Top-k DBPSB does not generate adequate query templates to test the hypothesis H.1: only 6 queries templates out of the 25 in DBPSB can be use in validating H.1 while the others have only one rankable variable. Further investigation is needed to refine the hypothesis H.1 and better qualify the cases that stress SPARQL engines when answering top-k queries.

¹ The implementation code is available online at https://bitbucket.org/sh_zahmatkesh/top-k-dbpsb

4 Conclusion

In this work, we presented Top-k DBPSB, an extension of DBPSB [11] that adds the possibility to automatically generate top-k queries. Top-k DBPSB satisfies the requirement of resembling the reality extending DBPSB which automatically derives datasets and queries from DBpedia and its query logs. Moreover, we provide experimental evidence that Top-k DBPSB also satisfies the requirement to stress the distinguishing features of top-k queries.

In order to support the claim that we positively answered to the research question presented in Section 1, we experimentally showed that the query variability provided by Top-k DBPSB stresses the SPARQL engines. To this end, we formulated three hypotheses and we empirically demonstrated that when the number of scoring variables increases the average execution time also does (hypothesis H.2) and that the average execution time is independent from the value used in the LIMIT clause (hypothesis H.3). Counterintuitively, hypothesis H.1 is not confirmed by our experiments.

References

1. Ilyas, I.F., Beskales, G., Soliman, M.A.: A survey of top-k query processing techniques in relational database systems. *ACM Computing Surveys (CSUR)* **40**(4) (2008) 11
2. Magliacane, S., Bozzon, A., Della Valle, E.: Efficient execution of top-k sparql queries. In: *The Semantic Web—ISWC 2012*. Springer (2012) 344–360
3. Wagner, A., Duc, T.T., Ladwig, G., Harth, A., Studer, R.: Top-k linked data query processing. In: *The Semantic Web: Research and Applications*. Springer (2012) 56–71
4. Cheng, J., Ma, Z., Yan, L.: f-sparql: a flexible extension of sparql. In: *Database and Expert Systems Applications*, Springer (2010) 487–494
5. Cedeno, J.P.: A Framework for Top-K Queries over Weighted RDF Graphs. PhD thesis, Arizona State University (2010)
6. Siberski, W., Pan, J.Z., Thaden, U.: Querying the semantic web with preferences. In: *ISWC. (2006)* 612–624
7. Erling, O., Mikhailov, I.: Rdf support in the virtuoso dbms. In Auer, S., Bizer, C., Müller, C., Zhdanova, A.V., eds.: *CSSW*. Volume 113 of LNI, GI (2007) 59–68
8. Broekstra, J., Kampman, A., van Harmelen, F.: Sesame: A generic architecture for storing and querying rdf and rdf schema. In Horrocks, I., Hendler, J.A., eds.: *International Semantic Web Conference*. Volume 2342 of *Lecture Notes in Computer Science*, Springer (2002) 54–68
9. Owens, A., Seaborne, A., Gibbins, N., mc schraefel: Clustered tdb: A clustered triple store for jena. Technical report, Electronics and Computer Science, University of Southampton (2008)
10. Gray, J., ed.: *The Benchmark Handbook for Database and Transaction Systems (2nd Edition)*. Morgan Kaufmann (1993)
11. Morsey, M., Lehmann, J., Auer, S., Ngomo, A.C.N.: Dbpedia sparql benchmark - performance assessment with real queries on real data. In Aroyo, L., Welty, C., Alani, H., Taylor, J., Bernstein, A., Kagal, L., Noy, N.F., Blomqvist, E., eds.: *International Semantic Web Conference (1)*. Volume 7031 of *Lecture Notes in Computer Science*, Springer (2011) 454–469

Exploring type-specific topic profiles of datasets: a demo for educational linked data

Davide Taibi¹, Stefan Dietze², Besnik Fetahu², Giovanni Fulantelli¹

¹Istituto per le Tecnologie Didattiche, Consiglio Nazionale delle Ricerche, Palermo, Italy
{davide.taibi, giovanni.fulantelli}@itd.cnr.it

²L3S Research Center, Hannover, Germany
{dietze, fetahu}@l3s.de

Abstract. This demo presents the dataset profile explorer which provides a resource type-specific view on categories associated with available datasets in the Linked Data cloud, in particular the ones of educational relevance. Our work utilises type mappings with dataset topic profiles to provide a type-specific view on datasets and their categorisation with respect to topics, i.e. DBpedia categories. Categories associated with each dataset are shown in an interactive graph, generated for the specific profiles only, allowing for more representative and meaningful classification and exploration of datasets.

Keywords: Dataset profile, Linked Data for Education, Linked Data Explorer

1 Motivation

The diversity of datasets in the Linked Data (LD) cloud has increased in the last few years, and identifying a dataset containing resources related to a specific topic is, at present, a challenging activity. Moreover, the lack of up-to-date and precise descriptive information has exacerbated this challenge. The mere keywords-based classification derived from the description of the dataset owner is not sufficient, and for this reason, it is necessary to find new methods that exploit the characteristics of the resources within the datasets to provide useful hints about topics covered by datasets and their subsequent classification.

In this direction, authors in [1] proposed an approach to create structured metadata to describe a dataset by means of topics, where a weighted graph of topics constitutes a dataset profile. Profiles are created by means of a processing pipeline¹ that combines techniques for datasets resource sampling, topic extraction and topic ranking. Topics have been associated to dataset by using named entity recognition (NER) techniques and a score, representing the relevance of a topic for a dataset, has been calculated using algorithms to evaluate node relevance in network such as PageRank, K-Step Markov, and HITS.

¹ <http://data-observatory.org/lod-profiles/profiling.htm>

The limitations of such an approach are related mainly to the following aspects. First, the meaning of individual topics assigned to a dataset can be extremely dependent on the type of resources they are attached to. Also, the entire topic profile of a dataset is hard to interpret if categories from different types are considered at the same time. As an example of the first issue, the same category (e.g. "Technology") might be associated to resources of very different types such as "video" (e.g. in the Yovisto Dataset²) or "research institution"(e.g. in the CNR dataset³). Concerning the second issue, the single topic profile attached for instance to bibliographic datasets (such as: the LAK dataset⁴ or Semantic Web Dog Food⁵) - in which people ("authors"), organisations ("affiliations") and documents ("papers") are represented – is characterized by the diversity of its categories (e.g. DBpedia categories: *Scientific_disciplines*, *Data_management Information_science* but also *Universities_by_country*, *Universities_and_colleges*). Indeed, classification of datasets in the LD Cloud is highly specific to the resource types one is looking at. While one might be interested in the classification of "persons" listed in one dataset (for instance, to learn more about the origin countries of authors in DBLP), another one might be interested in the classification of topics covered by the documents (for instance disciplines of scientific publications) in the very same dataset.

The approach we propose in this demo to overcome the limitations described above relies on filtering the topic profiles defined in [1] according to the types of the resources. This results in a type-specific categorisation of datasets, which considers both the categories associated with one dataset and the resource types these are associated with.

However, the schemas adopted by the datasets of the LD cloud are heterogeneous, thus making difficult to compare the topic profiles across datasets. While there are many overlapping type definitions representing the same or similar real world entities, such as "documents", "people", "organization", type-specific profiling relies on type mappings to improve the comparability and interpretation of types and consequently, profiles. For this aim the explicit mappings and relations declared within specific schemas (as an example *foaf:Agent* has as subclasses: *foaf:Group*, *foaf:Person*, *foaf:Organization*) as well as across schemas (for instance through *owl:equivalentClass* or *rdfs:subClassOf* properties) are crucial.

While relying on explicit type mappings we have based our demo on a set of datasets where explicit schema mappings are available from earlier work [2]. This includes education-related datasets identified by the LinkedUp Catalog⁶ in combination with the dataset profiles generated by the Linked Data Observatory⁷. While the latter provides topic profiles for all selected datasets, the LinkedUp Catalog contains explicit schema mappings which were manually created for the most frequent types in the dataset. Specifically, the profile explorer proposed in this demo aims at providing a resource type-specific view on categories associated with the datasets in the LinkedUp Catalog. In

² <http://www.yovisto.com/>

³ <http://data.cnr.it/>

⁴ <http://lak.linkededucation.org>

⁵ <http://data.semanticweb.org>

⁶ <http://data.linkededucation.org/linkedup/catalog/>

⁷ <http://data-observatory.org/lod-profiles>

this initial stage a selection of 23 dataset of the catalog have been considered, as representative of datasets including different resource types related to several topics. Type mappings across all involved datasets link "documents" of all sorts to the common *foaf:Document* class, "persons" and "organisations" to the common *foaf:Agent* class, and course and module to the *aiiso:KnowledgeGrouping*⁸ class. Categories associated with each dataset are shown in an interactive graph, generated for the specific types only, allowing for more representative and meaningful classification and exploration of datasets (Figure 1).

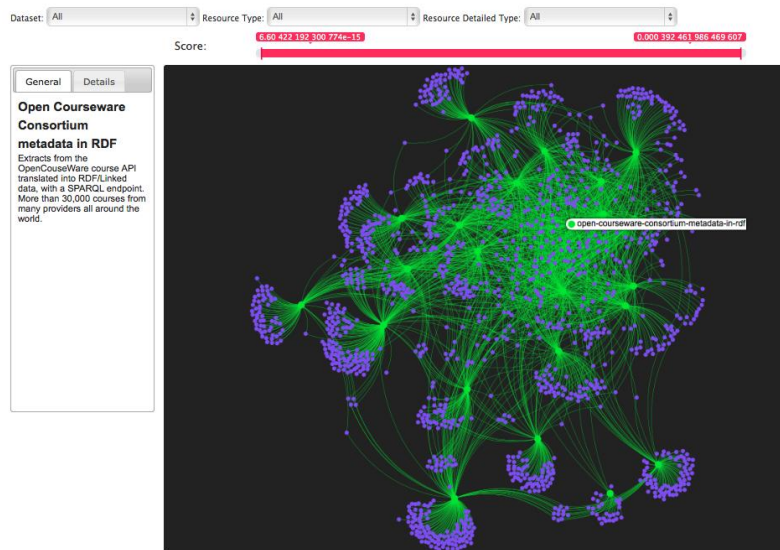


Fig. 1. A screenshot of the demo

2 The Dataset Profile Explorer

The dataset explorer is available at: <http://data-observatory.org/led-explorer/>. The explorer is composed of three panels: the panel at the center of the screen shows the network of datasets and categories, the panel on the left shows general and detailed descriptions about datasets and categories, and at the top of these panels the selection panel allows users to apply specific filters on the network. In the central panel, green nodes represent datasets while blue nodes represent categories. An edge connects a dataset to a category if the category belongs to the dataset topic profile. In order to draw the network, the *sigma.js*⁹ library has been used and the nodes of the network have been displayed using the ForceAtlas2 layout. By clicking on a node (dataset or category), general and detailed descriptions are shown on the left panel. In the case of a dataset,

⁸ <http://purl.org/vocab/aiiso/schema#KnowledgeGrouping>

⁹ <http://sigma.js.org>

the general description reports the description of the dataset retrieved from the Datahub repository¹⁰. In the detailed description, the list of the top ten categories (and the related score) associated to the dataset is reported. In the case of a category, the description panel reports the list of datasets which have that category in their profile. The datasets including the category in their top ten list are highlighted in bold.

The selection panel at the top allows users to filter the results by means of three combo boxes, respectively related to: dataset, resource type, and resource *sub-type*. The list of dataset is composed by the dataset of the LinkedUp catalog. Regarding the resource type, the explorer is focused on three classes: *foaf:Document*, *foaf:Agent* and *aiiso:KnowledgeGrouping*. The *foaf:Document* is related to learning material such as: research papers, books, and so on; the *foaf:Agent* resource type has been included to take into account elements such as persons and organizations. The *aiiso:KnowledgeGrouping* is a type representing resources related to courses and modules. This initial set of resource type can be easily enlarged by means of configuration settings. The resource sub-type has been included with the aim of refining the results already filtered by resource type. Another filter that has been included into the explorer is related to the score of the relationships between datasets and categories. A slider bar allows users to filter results based on a specific range of the scores. As stated before, the scores have been calculated by the linked dataset profiling pipeline. The filters on datasets, resource types and resource sub-types can be combined and, as a result, only the portion of the network consistent with the filter selections is highlighted

3 Conclusion

In order to foster an effective use of the resources in the LD cloud, it is important to make explicit the topics covered by the datasets even in relation to the types of resources in the datasets. To this aim, we have developed a dataset profile explorer focused on the domain of educational related datasets. In this domain, topic coverage and the type of the resources assume a key role in supporting the search for content suitable for a specific learning course. The explorer allows users to navigate topic profiles associated with datasets with respect to the type of the resource in the dataset.

The explorer can be configured to be used with different datasets provided that the dataset topic profile is available, thus extending the application of the proposed approaches to several fields.

4 References

1. Fetahu, B., Dietze, S., Nunes, B. P., Taibi, D., Casanova, M. A., Generating structured Profiles of Linked Data Graphs, 12th International Semantic Web Conference (ISWC2013), Sydney, Australia, (2013).
2. D'Aquin, M., Adamou, A., Dietze, S., Assessing the Educational Linked Data Landscape, ACM Web Science 2013 (WebSci2013), Paris, France, May 2013.

¹⁰ <http://datahub.io>

T_EX-OWL: a Latex-Style Syntax for authoring OWL 2 ontologies

Matteo Matassoni¹, Marco Rospocher², Mauro Dragoni², and Paolo Bouquet¹

¹ The University of Trento, Via Sommarive 9, Trento, I-38123, Italy

² Fondazione Bruno Kessler—IRST, Via Sommarive 18, Trento, I-38123, Italy

Abstract. This paper describes a new syntax that can be used to write OWL 2 ontologies. The syntax, which is known as T_EX-OWL, was developed to address the need for an easy-to-read and easy-to-write plain text syntax. T_EX-OWL is inspired by L^AT_EX syntax, and covers all construct of OWL 2. We designed T_EX-OWL to be less verbose than the other OWL syntaxes, and easy-to-use especially for quickly developing small-size ontologies with just a text editor. The important features of the syntax are discussed in this paper, and a reference implementation of a Java-based parser and writer is described.

1 Introduction and Motivation

Since OWL became a World (W3C) Wide Web Consortium recommendation, there has been a steady stream of Web (OWL) Ontology Language ontology editing tools that have made their way to users' desktops. Most notably, Protégé-OWL [1], Swoop [2], TopBraid Composer,¹ and MoKi [3].

All of these tools offer a variety of presentation or rendering formats for class, property and individual descriptions and axioms. These formats range from the W3C officially required RDF/XML exchange syntax [4], to the optionals: Turtle [5], OWL/XML [6], the Functional-Style Syntax [7], the Manchester Syntax [8], with some non-standard W3C syntaxes, like a Description-Logic style syntax and the Open (OBO) Biomedical Ontologies format [9].

While the use of ontology editing tools is becoming more and more popular, there are still situations where users have to quickly write small-size ontology for testing or prototyping purposes, and directly writing the ontology with a text editor would be more effective (i.e., the overhead of learning the ontology editing tool's functionalities and features is more than the benefit obtained by using it). These situations quite frequently occur in academic context.

W3C chose RDF/XML as the primary exchange syntax for OWL 2; indeed, this syntax must be supported by all OWL 2 tools. However, the fact that XML is extremely verbose and hard to write by hand excludes this syntax for quickly authoring and editing ontologies in a concise manner.

W3C provides alternatives to RDF/XML for OWL 2; these include Turtle, OWL/XML, the Functional-Style Syntax and the Manchester Syntax. OWL/XML is an XML

¹ <http://topbraidcomposer.com/>.

serialization for OWL 2 that mirrors its structural specification. It is more human-readable and easier to parse than RDF/XML; however like RDF/XML, OWL/XML is still XML. Another syntax that follows OWL 2's structural specification is the Functional-Style Syntax. It is a human-readable and plain text syntax that removes the burden of XML, however like the previous two, the Functional-Style Syntax is also verbose. In fact, it has an excessive number of keywords, and typically requires the use of a large number of brackets, as its name might suggest. The Manchester Syntax is a user-friendly syntax that can be used to write OWL 2 ontologies. It is the least verbose OWL 2 syntax and when it is used to write ontology documents, it gathers together information about names in a frame-like manner as opposed to the others OWL 2 syntaxes. This nature at a first look may seem a great advantage for the Manchester Syntax, but on the other hand it makes this syntax unable of handling General (GCI) Concept Inclusions (i.e., the Manchester Syntax does not cover the expressivity of the whole OWL 2 language).

The OWL Latex-Style Syntax was created to deal with the above issues and provide users with a lightweight syntax that makes it easier to write ontologies. It has been designed primarily for writing ontology documents in simple textual editor. The syntax is discussed in detail through the rest of this paper.

2 OWL Latex-Style Syntax

The full specification of the \TeX -OWL syntax is available at <http://github.com/matax87/TexOwl/blob/master/docs/grammar.pdf>, together with several examples of using the various syntax constructs. The primary design consideration in developing \TeX -OWL was to produce a syntax that was concise, and quick and easy to read and write by hand. We took inspiration for developing the syntax from the \LaTeX format, given its popularity especially in academic environments. A previous attempt to develop a latex-like syntax was proposed in [11], but its syntax is restricted to a limited subset of OWL 1. Lessons learned from this previous experience were also taken into consideration. For example, keywords that represent datatypes and annotations (i.e., datatypes and annotations were hard-coded in the syntax) were removed in the new syntax to generalise them via IRIs.

It was also decided that although the syntax should be aligned as much as possible with the OWL specification, for example by using keywords derived from the Functional-Style Syntax specification, the main objective would be to strive for conciseness and a reduction in the amount of time it took users to write axioms. To this end, some new keywords were created and others changed in name or name length. Moreover, it was also decided that the syntax should match as much as possible the \LaTeX format peculiarities of using keyword and command that start with a backslash (\backslash) symbol, with required parameters inside curly braces and optional parameters inside square brackets.

Although the \TeX -OWL Syntax borrows ideas from the OWL Functional-Style Syntax, it is much less verbose. An OWL ontology written in \TeX -OWL starts with an optional preface and continues with the actual ontology document. The optional preface is where prefixes can be declared via the new keyword $\backslash\text{ns}$. This keyword can be used also to declare a default prefix, which will be used for interpreting sim-

ple IRIs.² The actual ontology document begins with `\begin{ontology}` and ends with `\end{ontology}` syntax. After the begin ontology statement, users can also provide an optional ontology IRI and a even more optional version IRI typing them inside square brackets: `[ontologyIRI, versionIRI]`. Inside the begin/end block, user can import other ontology documents, using the keyword `\import`, declare axioms and put ontology annotations, as shown in the example below:

```
\ns <http://www.mydomain.org/african#>
\begin{ontology}[<http://www.mydomain.org/african>]
  % Animals form a class
  animal \c
  % Plants form a class disjoint from animals
  animal \cdisjoint plant
  % Trees are a type of plant
  tree \cisa plant
  % Branches are parts of trees
  branch \cisa \oforall{is_part_of}{tree}
  % Leaves are parts of branches
  leaf \cisa \oforall{is_part_of}{branch}
  % Herbivores are exactly those animals that eat only plants or parts of plants
  herbivore \ceq (animal \cand \oforall{eats}{(plant \cor \oforall{is_part_of}{
    plant})})
  % Carnivores are exactly those animals that eat animals
  carnivore \ceq (animal \cand \oexists{eats}{animal})
  % Giraffes are herbivores, and they eat only leaves
  giraffe \cisa (herbivore \cand \oforall{eats}{leaf})
  % Lions are animals that eat only herbivores
  lion \cisa (animal \cand \oforall{eats}{herbivore})
  % Tasty plants are plants that are eaten both by herbivores and carnivores
  tasty_plant \cisa \candof{plant, \oexists{eaten_by}{herbivore}, \oexists{
    eaten_by}{carnivore}}
  eaten_by \oinv eats
  eats \odomain animal
\end{ontology}
```

3 Implementation

A Java based reference implementation of a \TeX -OWL parser and writer were created.³ They use the OWLAPI framework [10] and were developed as modules that can be integrated inside it. The parser was constructed using the Java (JavaCC) Compiler Compiler [12]. It can parse complete ontologies written in \TeX -OWL. The writer, which inside the OWLAPI is known as *renderer*, can serialize OWLAPI's ontology objects to files written in \TeX -OWL. Moreover, the implementation also includes converters, which can transform a \TeX -OWL ontology to any other OWL 2 syntaxes and vice versa.

4 Concluding Remarks

\TeX -OWL is a new OWL 2 syntax that was designed in response to a demand from users for a more concise syntax that can be easily used to quickly write small-size ontologies by hand. Key features of the syntax are that it is inspired by the \LaTeX syntax:

² Simple IRIs are equivalent to abbreviated IRIs where the default prefix is used and there is no need need of typing the colon (':') symbol.

³ The implementation is available from <http://github.com/matax87/TexOwl/>.

in particular the syntax uses the same format for parameters and keywords. The syntax is suited for use in simple textual editor tools. A reference implementation of a Java based parser and a writer have been produced, which may be integrated into any tool. The implementation also includes converters, which can transform $\text{T}_{\text{E}}\text{X}$ -OWL to other OWL 2 syntaxes and vice versa.

In order to evaluate $\text{T}_{\text{E}}\text{X}$ -OWL, two questionnaires were designed and sent to knowledge engineers with experience in authoring ontologies with the various OWL syntaxes. In the first questionnaire (accessible here: <http://goo.gl/Cjprtg>), all OWL 2 syntaxes and $\text{T}_{\text{E}}\text{X}$ -OWL's intuitiveness, conciseness, and understandability were compared using ten different examples of use. The second questionnaire (accessible here: <http://goo.gl/lbFu4R>) was focused on evaluating the usability of the new syntax for authoring a small ontology. Ten knowledge engineers participated to the first questionnaire, and five of them took part to the second one. Ratings were expressed according to the typical five-level Likert scale.

In summary, the results show that $\text{T}_{\text{E}}\text{X}$ -OWL is indeed the most concise syntax and is intuitive as the Manchester Syntax, which is the most intuitive among OWL 2 syntaxes. Moreover, users have found it easy to use $\text{T}_{\text{E}}\text{X}$ -OWL for authoring a small example ontology and that, in general, this syntax is better to use for writing ontologies by hands than other OWL 2 syntaxes.

References

1. Knublauch, H., Musen, M.A., Rector, A.L.: Editing description logics ontologies with the Protégé OWL plugin (2004)
2. Kalyanpur, A., Parsia, B., Hendler, J.: A tool for working with web ontologies (2005)
3. Chiara Ghidini, Marco Rospocher, Luciano Serafini: Modeling in a Wiki with MoKi: Reference Architecture, Implementation, and Usages International Journal On Advances in Life Sciences, IARIA, volume 4, 111-124 (2012)
4. Fabien, G., Schreiber, G.: Rdf 1.1 xml syntax specification (2014) <http://www.w3.org/TR/2014/REC-rdf-syntax-grammar-20140225/>.
5. Beckett, D.: New syntaxes for rdf. Technical report, Institute For Learning And Research Technology, Bristol (2004)
6. Motik, B., Parsia, B., Patel-Schneider, P.F.: Owl 2 web ontology language xml serialization (second edition) (2012) <http://www.w3.org/TR/2012/REC-owl2-xml-serialization-20121211/>.
7. Motik, B., Parsia, B.: Owl 2 web ontology language structural specification and functional-style syntax (second edition) (2012) <http://www.w3.org/TR/2012/REC-owl2-syntax-20121211/>.
8. Horridge, M., Drummond, N., Goodwin, J., Rector, A.L., Stevens, R., Wang, H.: The manchester owl Syntax (2006)
9. Motik, B., Parsia, B.: Obo flat file format 1.4 syntax and semantics [draft] (2011) <ftp://ftp.geneontology.org/go/www/obo-syntax.html>.
10. The owl api <http://owlapi.sourceforge.net>.
11. Latex2owl, <http://dkm.fbk.eu/index.php/Latex2owl>.
12. Sreeni, V., Sriram, S.: Java compiler compiler [tm] (javacc [tm]) - the java parser generator <http://javacc.java.net>.

Supporting Integrated Tourism Services with Semantic Technologies and Machine Learning

Francesca A. Lisi and Floriana Esposito

Dipartimento di Informatica, Università degli Studi di Bari “Aldo Moro”, Italy
{francesca.lisi,floriana.esposito}@uniba.it

Abstract. In this paper we report our ongoing work on the application of semantic technologies and machine learning to Integrated Tourism in the Apulia Region, Italy, within the *Puglia@Service* project.

1 Introduction

Integrated Tourism can be defined as the kind of tourism which is explicitly linked to the localities in which it takes place and, in practical terms, has clear connections with local resources, activities, products, production and service industries, and a participatory local community. Integrated Tourism thus needs ICTs that should go beyond the mere technological support for tourism marketing, differently from most approaches in eTourism research (see [1] for a comprehensive yet not very recent review). In this paper, we report our experience in supporting Integrated Tourism services with Semantic Technologies (STs) and Machine Learning (ML). The work has been conducted within *Puglia@Service*,¹ an Italian PON Research & Competitiveness project aimed at creating an innovative service infrastructure for the Apulia Region, Italy.

The paper is structured as follows. Section 2.1 shortly describes a domain ontology for Integrated Tourism, named *OnTourism*, which has been modeled for being used in *Puglia@Service*. Section 2.2 briefly presents a Web Information Extraction (WIE) tool, named WIE-ONTOUR, which has been developed for populating *OnTourism* with data automatically retrieved from the Web. Section 2.3 illustrates some of the Semantic Web Services (SWSes) which have been defined on top of *OnTourism* for supporting Integrated Tourism in Apulia. Section 3 outlines an application scenario for a ML tool, named FOIL- \mathcal{DL} , to better adapt the automated composition of these services to user demands. Section 4 concludes the paper with final remarks and directions of future work.

2 Semantic Technologies for Integrated Tourism

2.1 A Domain Ontology

Domain ontologies for tourism are already available, *e.g.* the *travel*² ontology is centered around the concept of *Destination*. However, it is not fully satisfactory

¹ <http://www.ponrec.it/open-data/progetti/scheda-progetto?ProgettoID=5807>

² <http://www.protege.cim3.net/file/pub/ontologies/travel/travel.owl>

from the viewpoint of Integrated Tourism because, *e.g.*, it lacks concepts modeling the reachability of places. In *Puglia@Service*, we have decided to build a domain ontology, named *OnTourism*,³ more suitable for the project objectives and compliant with the OWL 2 standard. It consists of 379 axioms, 205 logical axioms, 117 classes, 9 object properties, and 14 data properties, and has the expressivity of the DL $\mathcal{ALCOF}(\mathbf{D})$.

The main classes of the terminology are *Site*, *Place* and *Distance*. The first is the root of a taxonomy which covers several types of sites of interest (*e.g.*, *Hotel* and *Church*). The second models the places where sites are located at. The third, together with the object properties *hasDistance* and *isDistanceFor* and the data properties *hasLengthValue/hasTimeValue*, allows to represent the distance relation between sites with values in either length or time units. Distances are further specified according to the transportation means used (see, *e.g.*, the class *Distance_on_Foot*). Other relevant classes in the terminology are *Amenity* (with subclasses such as *Wheelchair_Access*) and *Service* (with subclasses such as *Bike.Rental*) that model, respectively, amenities and services available at the accommodations. Finally, the terminology includes the official 5-star classification system for hotel ranking.

2.2 Ontology Population with Web Information Extraction

WIE-ONTOUR is a wrapper-based WIE tool implemented in Java and conceived for the population of *OnTourism* with data concerning hotels and B&Bs available in the web site of TripAdvisor⁴. The tool is also able to compute distances of the extracted accommodations from sites of interest (*e.g.*, touristic attractions) by means of the Google Maps⁵ API. Finally, the tool supports the user in the specification of sites of interest.

Instantiations of *OnTourism* for the main destinations of urban tourism in Apulia have been obtained with WIE-ONTOUR. Here, we consider an instantiation for the city of Bari (the capital town of Apulia). It contains 34 hotels, 70 B&Bs, 106 places, and 208 foot distances for a total of 440 individuals. The distances are provided in time and length on foot and have been computed with respect to *Basilica di San Nicola* and *Cattedrale di San Sabino* (both instances of *Church* and located in Bari). The restriction to foot distances is due to the aforementioned preference of Integrated Tourism for eco-mobility.

2.3 Semantic Web Services

In *Puglia@Service*, we have defined several atomic services in OWL-S on top of the aforementioned domain ontologies, *travel* and *OnTourism*. For example, *city_churches_service* returns the churches (o.p. of type *Church*) located in a given city (i.p. of type *City*) whereas *near_attraction_accomodations_service* returns all the accommodations (o.p. of type *Accommodation*) near a given attraction (i.p.

³ <http://www.di.uniba.it/~lisi/ontologies/OnTourism.owl>

⁴ <http://www.tripadvisor.com/>

⁵ <http://maps.google.com/>

of type *Attraction*). Note that closeness can be defined on the basis of distance either in a crisp way (*i.e.*, when the distance value is under a fixed threshold) or in a fuzzy way (*i.e.*, through grades of closeness). In both ways, however, the computation should consider the transportation means used as well as the measure units adopted according to the *OnTourism* ontology.

In *Puglia@Service*, we intend to obtain composite services by applying methods such as [3]. For example, the sequence composed of *city_churches_service* and *near_attraction_accomodations_service* could satisfy, *e.g.*, the user request for accommodations around *Basilica di San Nicola*. Indeed, since Bari is a major destination of religious tourism in Apulia, it could effectively support the demand from pilgrims who prefer to find an accommodation in the neighborhood of places of worship so that they can practise their own religions at any hour of the day. Also, if the suggested accommodations are easy to reach (*i.e.*, at foot distance) from the site of interest, the service will bring benefit also to the city, by reducing the car traffic. In a more complex scenario, disabled pilgrims might need a wheelchair-accessible accommodation. The service composition mechanism should then append also *wheelchairaccess_accomodations_service*, so that the resulting composite service could be considered more compatible with the special needs of this user profile.

3 Towards Learning from Users' Feedback

In *Puglia@Service*, automated service composition will be enhanced by exploiting users' feedback. The idea is to apply ML tools in order to induce ontology axioms which can be used for discarding those compositions that do not reflect the preferences/expectations/needs of a certain user profile. Here, we illustrate this idea with an application scenario which builds upon the accommodation rating provided by TripAdvisor's users. More precisely, we consider the task of accommodation finding. This task strongly relies on a classification problem aimed at distinguishing good accommodations from bad ones according to the amenities available, the services offered, the location and the distance from sites of interest, etc. In order to address this classification problem, we need ML tools able to deal with the inherent incompleteness of Web data and the inherent vagueness of concepts such as the closeness. One such tool is FOIL- \mathcal{DL} [2], a ML system able to induce a set of fuzzy General Concept Inclusion (GCI) $\mathcal{EL}(\mathbf{D})$ axioms from positive and negative examples for a target class in any OWL ontology.

As an illustration of the potential usefulness of FOIL- \mathcal{DL} in the *Puglia@Service* context, we report here a couple of experiments concerning the filtering of results returned by the SWSes reported in the previous section for the case of Bari. We set up a learning problem with the class *Bad_Accommodation* as target of the learning process. Ratings from TripAdvisor users have been exploited for providing FOIL- \mathcal{DL} with positive and negative examples. Out of the 104 accommodations, 57 with a higher percentage (say, over 0.7) of positive users' feedback are asserted as instances of *Good_Accommodation*, whereas 15 with a lower percentage (say, under 0.5) are asserted as instances of *Bad_Accommodation*. The latter, of course, play the role of positive examples in our learning problem. Syntactic restrictions are imposed on the form of the learnable GCI axioms.

In the first experiment, we have not considered the distances of the accommodations from the sites of interest. With this configuration, FOIL- \mathcal{DL} returns just the following GCI with confidence 0.5:

```
Bed_and_Breakfast and hasAmenity some (Pets_Allowed) and hasAmenity some (Wheelchair_Access)
  subclass of Bad_Accommodation
```

The GCI suggests that B&Bs are not recommended even though they provide disabled facilities. It can be used to filter out from the result set of *wheelchairaccess_accommodations_service* those accommodations which are classified as bad.

In the second experiment, conversely, we have considered the distances of the accommodations from the sites of interest. With this configuration, FOIL- \mathcal{DL} returns the following GCI with confidence 1.0

```
hasAmenity some (Bar) and hasAmenity some (Wheelchair_Access) and
hasDistance some (isDistanceFor some (Bed_and_Breakfast) and isDistanceFor some (Church))
  subclass of Bad_Accommodation
```

The GCI strenghtens the opinion that B&Bs are not recommendable accommodations for disabled people whatever their distance from the churches is.

As a further experiment, we have restricted our analysis of accommodations in Bari to only B&Bs. Starting from 12 positive examples and 39 negative examples for *Bad_Accommodation*, FOIL- \mathcal{DL} returns the following two GCIs with confidence 0.154 and 0.067 respectively:

```
hasAmenity some (Pets_Allowed) and hasAmenity some (Wheelchair_Access) subclass of Bad_Accommodation
hasAmenity some (Bar) and hasAmenity some (Wheelchair_Access) subclass of Bad_Accommodation
```

which confirm that B&Bs should not be recommended to disabled tourists.

4 Conclusions and future work

In this paper we have reported our ongoing work on the use of STs and ML for Integrated Tourism in Apulia within the *Puglia@Service* project. Though developed for the purposes of the project, the technical solutions here described are nevertheless general enough to be reusable for similar applications in other geographical contexts. Notably, they show the added value of having ontologies and ontology reasoning (including also non-standard inferences like induction as exemplified by FOIL- \mathcal{DL}) behind a Web Service infrastructure.

For the future we intend to carry on the work on the application of FOIL- \mathcal{DL} to the automated service composition. Notably, we shall consider the problem of learning from the feedback provided by specific user profiles.

References

1. Buhalis, D., Law, R.: Progress in information technology and tourism management: 20 years on and 10 years after the internet - the state of eTourism research. *Tourism Management* 29(4), 609–623 (2008)
2. Lisi, F.A., Straccia, U.: A System for Learning GCI Axioms in Fuzzy Description Logics. In: Eiter, T., Glimm, B., Kazakov, Y., Kroetzsch, M. (eds.) Proc. of the 26th Int. Workshop on Description Logics. CEUR Workshop Proceedings, vol. 1014. CEUR-WS.org (2013)
3. Redavid, D., Iannone, L., Payne, T.R., Semeraro, G.: OWL-S atomic services composition with SWRL rules. In: An, A., Matwin, S., Ras, Z.W., Slezak, D. (eds.) Foundations of Intelligent Systems. LNCS, vol. 4994, pp. 605–611. Springer (2008)

Towards a Semantically Enriched Online Newspaper

Ricardo Kawase, Eelco Herder, Patrick Siehndel

L3S Research Center, Leibniz University Hannover, Germany
{kawase, herder, siehndel}@L3S.de

Abstract. The Internet plays a major role as a source of news. Many publishers offer online versions of their newspapers to paying customers. Online newspapers bear more similarity with traditional print papers than with regular news sites. In a close collaboration with Mediengruppe Madsack - publisher of newspapers in several German federal states, we aim at providing a semantically enriched online newspaper. News articles are annotated with relevant entities - places, persons and organizations. These annotations form the basis for an entity-based 'Theme Radar', a dashboard for monitoring articles related to the users' explicitly indicated and inferred interests.

1 Introduction

Traditional print media are nowadays replaced or complemented by online media. Most publishers of international, national and local newspapers use the Web as an additional communication channel. Many news sites that are connected to a print newspaper also offer online subscriptions or provide content as pay-per-view. A commonly used solution is subscription-based access to an online newspaper, which is a digital copy of the print newspaper, often with additional features for search, recommendation or archiving. However, in most cases, these additional features are based on content analysis, manual interlinking by the editors and collaborative filtering. In this paper, we present our work towards an semantically enriched online newspaper, which is a currently running collaboration between the L3S Research Center and Madsack GmbH & Co. KG.

1.1 Madsack

Madsack GmbH & Co. KG is a German media group with headquarters in Hannover, Germany. Its core business comprises the production of several regional newspapers in Lower Saxony, Schleswig-Holstein, Mecklenburg-Western Pomerania, Saxony, Hessen and Saxony-Anhalt. Madsack is the sixth largest publishing house in Germany¹; in the year 2012, the average circulation of their 18 paid newspapers amounted to 939,590 copies².

¹ http://www.media-perspektiven.de/uploads/tx_mppublications/05-2010_Roeper.pdf

² http://www.madsack.de/fileadmin/content/downloads/Geschaeftsbericht_MGM_2012-2013_web.pdf

The digital business of Madsack media group includes the distribution of editorial content (e-paper, mobile apps, usually using the brand of the corresponding daily newspapers), marketing services (e.g. programming of websites and apps) as well as collaborations with online marketplaces.

We focus on Madsack's e-paper product. The e-paper is a Web environment that allows subscribers to access the daily editions of the newspaper in digital format. The environment is restricted to paying subscribers, who are required to log in with their personal username and password. Once they are logged in, the website presents the reader current daily newspaper editions. The online newspaper holds the same design as the printed version. Every morning, except on Sundays (there are no editions printed on Sundays), a news daily edition is available on the website.

2 Enrichment

In order to effectively archive, categorize and publish news articles, most larger media companies have documentation departments that assign labels, categories or terms to news articles. Due to the increasingly large amount of items and the need for the term assignment to be quick [3], automatic semantic annotation is increasingly considered as an alternative for human annotation. Several established off-the-shelf tools for knowledge extraction and semantic annotation are readily available, including DBpedia Spotlight [4], AIDA [7], Open Calais, Wikimeta and Zemanta [2]. Wikipedia Miner [6] directly makes use of the evolving Wikipedia structure; the toolkit includes search and annotation services and provides links to relevant Wikipedia articles. In a comparison of entity-annotation systems [1], Wikipedia Miner consistently scored high in terms of recall and F1.

We semantically enriched the news articles by identifying entities and types. For this purpose, we use the Wikipedia Miner[5] service as an annotation tool. First, detected words are disambiguated using machine learning algorithms that take the context of the word into account. This step is followed by the detection of links to Wikipedia articles (which will later be aligned with DBpedia entities). By using a predefined threshold, we ensured that only those words that are relevant for the whole document are linked to articles. The goal of the whole process is to annotate a given news article in the same way as a human would link a Wikipedia article. We set up a local deployment of Wikipedia Miner and trained the models on top of a German Wikipedia dump from February, 2014³.

After annotating the content of the news articles, with the identified entities in hand, we query DBpedia in order to gather further information regarding the entities. Specifically, we explore their relationships through the predicate **rdf:type** to extract the type of the entity given by DBpedia's ontology (dbpedia-owl). Although several different types are identified, we selected the three most relevant types for news articles: **dbpedia-owl:Place**, **dbpedia-owl:Person** and **dbpedia-owl:Organisation**. These three types were reported by Madsack's editorial staff to be the most relevant for their readers. Additionally, as we describe in Section 3, it is important to avoid an overload of features and information to the readers. Thus, we aim at having just a few and very useful facets that can improve relevant news retrieval.

³ <http://dumps.wikimedia.org/dewiki/20140216/>

Hi ricardo [Logout](#) [Artikel für mich](#) [meine Artikel](#) [meine Themenradar](#) [Search](#)

MY THEMENRADAR ▼

Adidas | Angela Merkel | Apple | Bad Waltersdorf | Berlin | Bundesrat | Euro-Schuldenkrise | Griechenland | Grüne | Hannover | Hannover 96 | Hände | Kiew | Lady Gaga | London | Manchester | Mirko Slomka | Staatsanwaltschaft | Sänger

GENERAL SUGGESTIONS ▼

Bundesregierung | Bundeswertpapiere | Bühne | England | Euro-Krise | Europäische Union | Fußball-Bundesligisten | Fußballspiel | Graz | Natürlich | Nokia | Präsident | Rendite | Rentenmarkt | Rösler | Stürmer | Ukraine | Verfügung

SEMANTICS

PLACES ▼

MY THEMENRADAR: Bad Waltersdorf | Berlin | Griechenland | Hannover | Kiew | London

SUGGESTIONS: Graz | Ukraine

COMPANIES ▼

MY THEMENRADAR: Adidas | Apple

SUGGESTIONS: Nokia

PEOPLE ▼

MY THEMENRADAR: Angela Merkel | Lady Gaga | Mirko Slomka

BOOKS

<< Politik >>

POLITIK

MY THEMENRADAR: Griechenland

SUGGESTIONS: Euro-Krise | Bundesregierung | Europäische Union | Präsident | Rösler | Stürmer

<< Magazin >>

MAGAZIN

MY THEMENRADAR: Sänger

SUGGESTIONS: Bühne

Fig. 1. Madsack's e-paper prototype interface.

3 User Interface

From the end users' (the readers') perspective, the main innovation of the e-paper is the so called 'Themenradar' (Theme Radar). The Theme Radar provides users with shortcuts to news articles that pertain to their particular interests - as explicitly indicated by subscribing to an entity (which represents a theme or topic), combined with the entities that most often occur in the articles that the user read so far. Augmenting the 'Theme Radar' with the assistance of semantically enriched data is, in fact, one of our main goals in this collaboration.

Figure 1 depicts the first prototype of the 'Theme Radar'. It consists of a dashboard of the readers' interests. The 'Theme Radar' works as a semantically enhanced topic dashboard that enables readers to get suggestions for themes and topics, to manage their

topics and to get personalized news recommendations based on entity co-occurrences, linked data relations and the aforementioned semantic properties types.

Based on the users' activity logs, the system automatically builds the 'Theme Radar'. Top entities of interest are presented to the users in their 'Theme Radar' with additional suggested entities and recommended articles (based on entity co-occurrence). In the interface, these entities are grouped by type and also by 'Book' (Books are sections within the newspaper, as predefined by the editors - such as 'Sports' and 'Politics'). Additionally, the users can manually add entities to their profiles, which get higher weights in the news recommendation process.

4 Conclusions

In this paper, we presented our work towards a semantically enriched online newspaper, which - to the best of our knowledge - is the first of its kind in a fully commercial setup. We are currently on a stage of interface designing which, in a commercial product, requires the validation and approval from several stakeholders. As future work, we plan to evaluate the quality of the annotations with user feedback and to perform an analysis of online reading behavior, with a focus on the semantic aspects. Building upon these steps, we plan to develop an entity-based news recommender that fulfills Madsack's online readers' interests, and to evaluate them in practice.

5 Acknowledgement

We would like to thank the Madsack Online GmbH & Co. KG team for the collaboration opportunity and the support during the implementation presented in this work.

References

1. M. Cornolti, P. Ferragina, and M. Ciaramita. A framework for benchmarking entity-annotation systems. In *Proceedings of the 22nd international conference on World Wide Web*, pages 249–260. International World Wide Web Conferences Steering Committee, 2013.
2. A. Gangemi. A comparison of knowledge extraction tools for the semantic web. In *The Semantic Web: Semantics and Big Data*, pages 351–366. Springer, 2013.
3. A. L. Garrido, O. Gómez, S. Ilarri, and E. Mena. An experience developing a semantic annotation system in a media group. In *Natural Language Processing and Information Systems*, pages 333–338. Springer, 2012.
4. P. N. Mendes, M. Jakob, A. García-Silva, and C. Bizer. Dbpedia spotlight: shedding light on the web of documents. In *Proceedings of the 7th International Conference on Semantic Systems*, pages 1–8. ACM, 2011.
5. D. Milne and I. H. Witten. Learning to link with wikipedia. In *CIKM '08: Proceeding of the 17th ACM conference on Information and knowledge management*, pages 509–518, New York, NY, USA, 2008. ACM.
6. D. Milne and I. H. Witten. An open-source toolkit for mining wikipedia. *Artificial Intelligence*, 194:222–239, 2013.
7. M. A. Yosef, J. Hoffart, I. Bordino, M. Spaniol, and G. Weikum. Aida: An online tool for accurate disambiguation of named entities in text and tables. *Proceedings of the VLDB Endowment*, 4(12):1450–1453, 2011.

Identifying Topic-Related Hyperlinks on Twitter

Patrick Siehndel, Ricardo Kawase, Eelco Herder and Thomas Risse

L3S Research Center, Leibniz University Hannover, Germany
{siehndel, kawase, herder, risse}@L3S.de

Abstract. The microblogging service Twitter has become one of the most popular sources of real time information. Every second, hundreds of URLs are posted on Twitter. Due to the maximum tweet length of 140 characters, these URLs are in most cases a shortened version of the original URLs. In contrast to the original URLs, which usually provide some hints on the destination Web site and the specific page, shortened links do not tell the users what to expect behind them. These links might contain relevant information or news regarding a certain topic of interest, but they might just as well be completely irrelevant, or even lead to a malicious or harmful website. In this paper, we present our work towards identifying credible Twitter users for given topics. We achieve this by characterizing the content of the posted URLs to further relate to the expertise of Twitter users.

1 Introduction

The microblogging service Twitter has become one of the most popular and most dynamic social networks available on the Web, reaching almost 300 million active users [1]. Due to its popularity and dynamics, Twitter has been topic of various areas of research. Twitter clearly trades content size for dynamics, which results in one major challenge for researchers - tweets are too short to put them into context without relating them to other information. Nevertheless, these short messages can be combined to build a larger picture of a given user (user profiling) or a given topic. Additionally, tweets may contain hyperlinks to external additional Web pages. In this case, these linked Web pages can be used for enriching tweets with plenty of information.

An increasing number of users post URLs on a regular basis, and there are more than 500 million Tweets every day¹. With such a high volume, it is unlikely that all posted URLs link to relevant sources. Thus, measuring the quality of a link posted on Twitter is an open question [3].

In many cases, a lot can be deduced just by the URL of a given Web page. For example, if the URL domain is from a news provider, a video hosting website or a social network, the user already knows more or less what to expect after clicking on it. However, regular URLs are, in many cases, too long to fit in a single tweet. Consequently, Twitter automatically reduces the link length using shortening services. This leads to the problem that the user's educated guess of what is coming next is completely gone. In this work, we focus on ameliorating these problems by identifying those tweets containing URLs that might be relevant for the rest of the community.

The reasonable assumption behind our method is that users who usually talk about a certain topic ('experts') will post interesting links about the same topic. The strong

¹ <https://blog.twitter.com/2013/new-tweets-per-second-record-and-how>

point in our method is that it is independent of the users' social graph. There is no need to verify the user's network or the retweet behavior. Thus, it can be calculated on the fly. To achieve our final goal, we divide our work in two main steps: the generation of user profiles [5] and the generation of URL profiles. In this paper, we focus on the latter step.

2 Methodology

In our scenario, we build profiles for Twitter users based on the content of their tweets. Besides the profiles for users we also generate profiles for the URLs posted by the users. One of the biggest challenges in this task is to find appropriate algorithms and metrics for building comparable profiles for users and websites. The method we developed to solve this task is based on the vast amount of information provided by Wikipedia. We use the articles and the related category information supplied by Wikipedia to define the topic and the expertise level inherent in certain terms. Our method consists of three main steps to create profiles for users and websites.

Extraction: In this step, we annotate the given content (all tweets of a user, or the contents of a Web page) using the Wikipedia Miner Toolkit [4]. The tool provides us with links to Wikipedia articles. The links discovered by Wikipedia Miner have a similar style to the links that can be found inside a Wikipedia article. Not all words that have a related article in Wikipedia are used as links, but only words that are relevant for the whole topic are used as links, if a detected article is relevant for the whole text is based on different features like the relatedness to other detected articles or generality of the article.

Categorization: In the second stage, Categorization, we extract the categories of each entity that has been mentioned in the users' tweets or in the posted URL. For each category, we follow the path through all parent categories, up to the root category. In most cases, this procedure results in the assignment of several top-level categories to an entity. Since the graph structure of Wikipedia contains also links to less relevant categories, we only follow links to parent categories which distance to the root is shorter or less than the one of the child category. For each category, a weight is calculated by first defining a value for the detected entity. This value is based on the distance of the entity to the root node. Following the parent categories, we divide the weight of each node by the number of sibling categories. This step ensures, that categories do not get higher values just because of a broader structure inside the graph. Based on this calculation, we give higher scores to categories that are deeper inside the category graph and more focused on one topic.

Aggregation: In the final stage, Aggregation, we perform a linear aggregation over all of the scores for a document, in order to generate the final profile for the user (or for the website). The generated profile displays the topics a user/website talks about, as well as the expertise in - or focus on - a certain topic.

3 Validation

As mentioned in Section 1, in this paper we focus our attention on the generation of URL profiles and the relation to the corresponding tweets and users. Thus, in order to validate our methodology, we crawled Twitter with a number of predefined queries (keywords) and collected all resulting tweets that additionally contain URLs. We have

Table 1. Statistics about the used dataset.

	Items	Annotations	Annotations per Item
Topic Tweets	83,300	88,530	1.06
Linked Wedsites	40,940	457,164	11.1
All Tweets	11,303,580	30,059,981	3.127

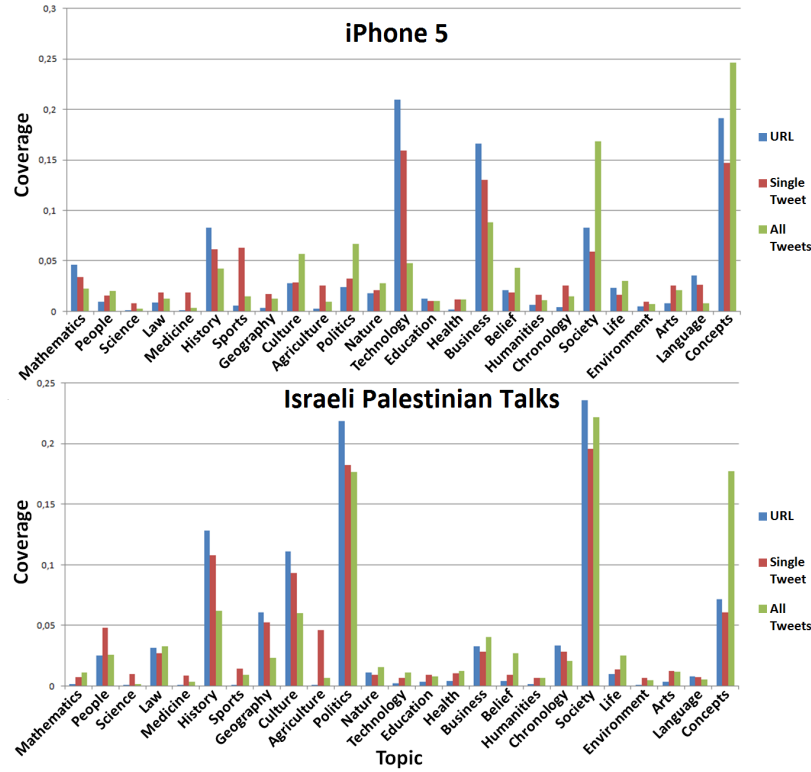


Fig. 1. Coverage of Wikipedia Categories based on the URL Content for each selected topic.

previously validated our approach by characterizing and connecting heterogeneous resources based on the aggregated topics [2]. Here, the goal is to qualitatively validate if the topic assignment given by our method in fact represents the real topics that are expected to be covered in a given query.

3.1 Dataset

The used dataset consists of around 83,300 tweets related to seven different topics. The idea behind this approach is, to collect a series of tweets that contain links and certain keywords relevant for one particular topic. Within these tweets, we found 40,940 different URLs. For each of these URLs, we tried to download and extract the textual content, which resulted in 26,475 different websites. Additionally we downloaded the last 200 posts for each user. The numbers of the dataset are shown in Table 1.

Table 2. Correlations between created profiles

	URL Content Single Tweet	URL Content User Tweets	Single Tweet User Tweets
Edward Snowden	0.995	0.968	0.961
Higgs Boson	0.812	0.628	0.496
Iphone 5	0.961	0.698	0.664
Israel Palastinian Talks	0.984	0.884	0.867
Nexus 5	0.968.	0.972	0.956
Obamacare	0.983	0.79	0.752
World Music Avars	0.921	0.718	0.614
All topics average	0.946	0.808	0.759

3.2 Topic Comparison

Figure 1 shows the generated profiles for two of the chosen example topics. The shown profiles are averaged over all users and show the profiles based on the content of the crawled web pages, based on the tweets containing the URLs and based on the complete user profile (last 200 Tweets, based on API restrictions). We can see that for the very specific topic ‘Israeli Palestinian Talks’ the generated profiles are very similar. For the topic ‘iPhone 5’ the profiles are less similar, since this topic or keyword is less specific it becomes much harder for a user to find the content he is looking for. A tweet like ‘*The new iPhone is really cool*’ together with a link may be related to many different aspects of the product. Table 2 displays the correlation between the different profiles for the chosen exemplifying topics. While users who write about topic like ‘Snowden’ or ‘Nexus phones’ seem to write about related topics in most of their tweets, this is not true for more general topics.

4 Conclusion

In this paper, we presented a work towards the identification of credible topic-related hyperlinks in social networks. Our basic assumption is that users who usually talk about a certain topic (‘experts’) will post interesting (and safe) links about the same topic. The final goal of our work requires to analyze the quality of the posted URLs. Here, we presented our profiling method with preliminary results of the URL profiles. As future work we plan to analyze the quality of profiles and URLs in order to provide a confidence and quality score for URLs.

5 Acknowledgment

This work has been partially supported by the European Commission under ARCOMEM (ICT 270239) and QualiMaster (ICT 619525)

References

1. Twitter now the fastest growing social platform in the world. <http://globalwebindex.net/thinking/twitter-now-the-fastest-growing-social-platform-in-the-world/>, Jan. 2013.
2. R. Kawase, P. Siehdnel, B. P. Nunes, E. Herder, and W. Nejdl. Exploiting the wisdom of the crowds for characterizing and connecting heterogeneous resources. In *HT*, 2014.
3. S. Lee and J. Kim. Warningbird: Detecting suspicious urls in twitter stream. In *NDSS*, 2012.
4. D. Milne and I. H. Witten. An open-source toolkit for mining wikipedia. *Artificial Intelligence*, 194:222–239, 2013.
5. P. Siehdnel and R. Kawase. Twikime! - user profiles that make sense. In *International Semantic Web Conference (Posters & Demos)*, 2012.

Capturing and Linking Human Sensor Observations with YouSense

Tomi Kauppinen^{1,2} and Evgenia Litvinova¹ and Jan Kallenbach¹

¹ Department of Media Technology
Aalto University School of Science, Finland

`evgenia.litvinova@aalto.fi`

`jan.kallenbach@aalto.fi`

² Cognitive Systems Group,
University of Bremen, Germany

`tomi.kauppinen@aalto.fi`

Abstract. Semantic technologies are prominent for gathering human sensor observations. Linked Data supports sharing and accessing of not just data but also vocabularies describing the data. Human sensor observations are often a combination of natural language and categorizable entries, thus calling for semantic treatment. Space and time serve as natural integrators of data in addition to concepts. In this paper we demonstrate YouSense tool which supports gathering of experiences about spaces (like generic buildings or office spaces). Our contribution is also a vocabulary for describing the experiences as RDF and tools for visualizing and making sense of the gathered user experiences.

1 Introduction

Understanding how people experience surroundings (like office spaces or conference venues) supports to further develop spaces and to modify them to meet user needs. This *Human Sensor Web* approach is rather different from monitoring just technical parameters such as the indoor temperature, humidity, or CO2 concentration, typically in use to assess the performance of buildings.

The promise of Linked Data and semantic technologies in this setting is that they can offer ways to share and access these human sensor observations in a radically novel ways. The crucial tasks are to figure out how to describe the experiences of people in a processable way, and also how to gather the observations. Finally, Information Visualization is a useful step in understanding if the gathered data has a story to tell [1,2].

In this paper we demonstrate and present YouSense, a mobile and web application for supporting sensing of the spaces by people. We also present the EXPERIENCE vocabulary, a Linked Data compatible collection of terms to enable creation of RDF about the gathered experiences. The resulting history data provides evidences about comfortable and problematic spaces for both users of buildings and building managers. Our contribution includes also a set of visualization tools to make sense of the human sensor observations, and their thematic, spatial and temporal characteristics.

The paper is structured as follows. Section 2 presents the tool for gathering the user experiences and discusses the vocabulary for encoding them. Section 3 demonstrates the use of the gathered data and outlines questions one can ask with the system. Section 4 provides future work research agenda and concluding remarks.

2 Gathering Observations for Semantic Descriptions

YouSense³ is a web application that can be used from both mobile and desktop browsers. In YouSense, a user makes a sentence to describe how he/she feels in the space (see Figure 1 for an example). The structure of the sentence is defined with the help of the EXPERIENCE⁴ Vocabulary. EXPERIENCE is a lightweight vocabulary providing terms for creating descriptions of experiences (as instances of *Experience*) about the environment, for example how cold or warm one feels in a particular context created by the location, time and activities an *Experiencer* is involved with. It thus supports for describing user experiences and feelings.

The image shows two side-by-side panels of the YouSense web application. The left panel, titled 'HOW DO YOU FEEL?', contains a form with the following fields: 'When?' with the value 'NOW', 'Where?' with the value 'in ROOM 3156', 'What?' with the value 'I feel _____', 'Why?' with the value 'because of _____', 'Activity' with the value 'I'm _____', and 'Then...' with the value 'I will _____'. At the bottom of this panel is a red button labeled 'SUBMIT FEEDBACK!'. The right panel, titled 'WHAT?', has a 'Back' button at the top left. It displays a text box with the sentence: 'Now in the room 3156 I feel VERY COLD, FRESH AIR because of ____, I'm ____, I will ____'. Below the text box are several colored buttons: 'OK' (blue), 'EXCELLENT' (blue), 'HOT' (blue), 'COLD' (orange), 'CHILLY' (blue), 'SMELLY' (blue), 'TOO BRIGHT' (blue), 'TOO DARK' (blue), 'STUFFED AIR' (blue), 'FRESH AIR' (orange), 'COZY' (blue), and 'ANNOYED' (blue). At the bottom of the right panel is a grey button labeled 'ENTER OWN'.

Fig. 1. YouSense in action.

The structure of EXPERIENCE was designed to include things people generally use when they describe situations in spaces. We conducted a diary study, where people were asked to report about their experience about spaces in a free form. By analyzing the results of the user study we designed the set of terms to be included in EXPERIENCE. For instance, while we supposed people would

³ Demonstration also available online at <http://yousense.aalto.fi>

⁴ <http://linkedearth.org/experience/ns/>

like to relate the experiences to spaces, and times, it was rather surprising that they also wanted to guess about the reason for the experience.

Below is a simple example use of the EXPERIENCE Vocabulary, in line with Figure 1 but a completed one. In this example there is a person (*Laura*) who has experience (*VeryCold*) and (*FreshAir*) in her office (*Room3156*) in June 2014 while performing certain activity (*Sitting*). The observer has also communicated about the (possible) reason (*Window is open*) for the experience (*VeryCold*). We also gather the action that the experiencer plans to do next (*Work*).

```
@prefix experience:<http://linkedearth.org/experience/ns#> .

example:exampleExperience_34
  a experience:Experience ;
  experience:hasExperiencer feelingdata:Laura ;
  experience:hasEventExperience feelingdata:VeryCold, feelingdata:FreshAir ;
  experience:hasLocation feelingdata:Room3156 ;
  experience:hasTime "2014-06-20T10:00+02:00" ;
  experience:hasActivity dbpedia:Sitting ;
  experience:hasReason "Window is open" ;
  experience:hasFollowingAction dbpedia:Work .
```

3 Experimenting with YouSense in Concrete Use Cases

The sensemaking part of the YouSense creates diagrams for each of the message parts (reasons, locations, times, people, spaces, activities) with reasoning support (partonomy, concept hierarchy, temporal abstraction). For instance, the adjectives people reported about the certain experiences (such as cold or warm) about spaces support understanding of spaces.

Figure 2 depicts the approach by presenting a set of experiences about spaces as a bubble visualization. The experiences are aggregated to positive (green) and negative (red) ones. The floor plan visualization on the right shows a heat map of these aggregated experiences by rooms. Zooming closer to a room allows to see more detailed information about collected experiences and the spatial configuration of the room. The idea is to retrieve patterns like “rooms facing the sea generally have more positive experiences than ones facing the inner yard” thus supporting to reveal the causes of the experiences.

We have experimented YouSense with selected spaces at the Aalto University to evaluate its usefulness. These spaces include the Design Factory⁵, Media Factory⁶ and the spaces of the Department of Media Technology and Department of Automation and Systems Technology. According to the experiments and discussions with building managers the following types questions arose as the ones needing to be answered by making sense of the gathered data.

- what is the air quality of this space?
- do people feel comfortable in this space?
- do people stay in this space to work and study?

⁵ <http://www.aaltodesignfactory.fi>

⁶ <http://mediafactory.aalto.fi>

An Update Strategy for the WaterFowl RDF Data Store

Olivier Curé¹, Guillaume Blin²

¹ Université Paris-Est, LIGM - UMR CNRS 8049, France
ocure@univ-mlv.fr

² Université de Bordeaux, LaBRI - UMR CNRS 5800, France
guillaume.blin@labri.fr

Abstract. The WaterFowl RDF Store is characterized by its high compression rate and a self-indexing approach. Both of these characteristics are due to its underlying architecture. Intuitively, it is based on a stack composed of two forms of Succinct Data Structures, namely bitmaps and wavelet trees. The ability to efficiently retrieve information from these structures is performed via a set of operations, *i.e.*, **rank**, **select** and **access**, which are used by our query processor. The nice properties, *e.g.* compactness and efficient data retrieval, we have observed on our first experimentations come at the price of poor performances when insertions or deletions are required. For instance, a naive approach has a dramatic impact on the capacity to handle ABox updates. In this paper, we address this issue by proposing an update strategy which uses an hybrid wavelet tree (using both pointer-based and pointerless sub-wavelet trees).

1 Introduction

Large amount of RDF data are being produced in diverse domains. Such a data deluge is generally addressed by distributing the workload over a cluster of commodity machines. We believe that this will soon not be enough and that in order to respond to the exponential production of data, the next generation of systems will distribute highly compressed data. One valuable property of such systems would be to perform some data oriented operations without requiring a decompression phase.

We have recently proposed the first building blocks of such a system, namely WaterFowl, for RDF data [1]. The current version corresponds to an in-memory, self-indexed, operating at the bit level approach which uses data structures with a compression rate close to theoretical optimum. These so-called Succinct Data Structures (SDS) support efficient decompression-free query operations on the compressed data. The first components we have developed for this architecture are a query processor and an inference engine which supports the RDFS entailment regime with inference materialization limited to **rdfs:range** and **rdfs:domain**. Both of these components take advantage of the SDS properties and highly performant operations. Of course, SDS are not perfect and a main limitation corresponds to their inability to efficiently handle update operations,

i.e., inserting or deleting a bit. In the worst case, one has to completely rebuild the corresponding SDS (bitmap or wavelet tree) to address such updates. Even if we consider that the sweet spot for RDF stores is OnLine Analytic Processing (OLAP), rather than OnLine Transactional Processing (OLTP), such a drawback is not acceptable when managing data sets of several millions of triples.

The main contribution of this paper is to present an update strategy that addresses instance updates. Due to space limitations, we do not consider updates at the schema level (*i.e.*, TBox). This approach is based on a set of heuristics and the definition of an hybrid wavelet tree using both pointer-based and pointerless sub-wavelet trees.

2 WaterFowl architecture

Figure 1 presents the three main components of the WaterFowl system: dictionary, query processing and triples storage. The former is responsible for the encoding and decoding of the entries of triple data sets. The main contribution here consists of encoding the concepts and properties of an ontology such that their respective hierarchy are using common binary prefixes. This enables us to perform prefix versions of the **rank**, **select** and **access** operations and thus prevents navigating the whole depth of the wavelet trees to return an answer.

The query processing component handles the classical operations of a database query processor and communicates intensively with the dictionary unit. A peculiarity of this query processor is to translate SPARQL queries into sequences of **rank**, **select** and **access** SDS operations over sequences. They are designed to (i) count the number of occurrences of a letter appearing in a prefix of a given length, (ii) find the position of the k^{th} occurrence of a letter and (iii) retrieve the letter at a given position in the sequence.

Finally, the triple storage component consists of two layers of bitmaps and wavelet trees³. It uses an abstraction of a set of triples represented as a forest where each tree corresponds to a single subject, *i.e.*, its root, the intermediate nodes are the properties of the root subject and the leaves are the objects of those subject-property pairs. The first layer encodes the relation between the subjects and the predicates of a set of triples. It is composed of a bitmap, to encode the subjects, and a wavelet tree, to encode the sequences of predicates of those subjects. Unlike the first layer, the second one has two bitmaps and two wavelet trees. B_o encodes the relation between the predicates and the objects; that is the edges between the leaves and their parents in the tree representation. Whereas, the bitmap B_c encodes the positions of ontology concepts in the sequence of objects. Finally, the sequence of objects obtained from a pre-order traversal in the forest is split into two disjoint subsequences; one for the concepts and one for the rest. Each of these sequences is encoded in a wavelet tree (resp. WT_{oc} and WT_{oi}). This architecture reduces sparsity of identifiers and enables the management of very large datasets and ontologies while allowing time and

³ Due to space limitations, we let the reader refer to [1] for corresponding definitions

space efficiency. More details on these components are proposed in [1]. That paper presents some evaluations where different wavelet tree implementations have been used: with and without pointers and one so-called matrix [3]. The characteristics of the first two motivated our update approach.

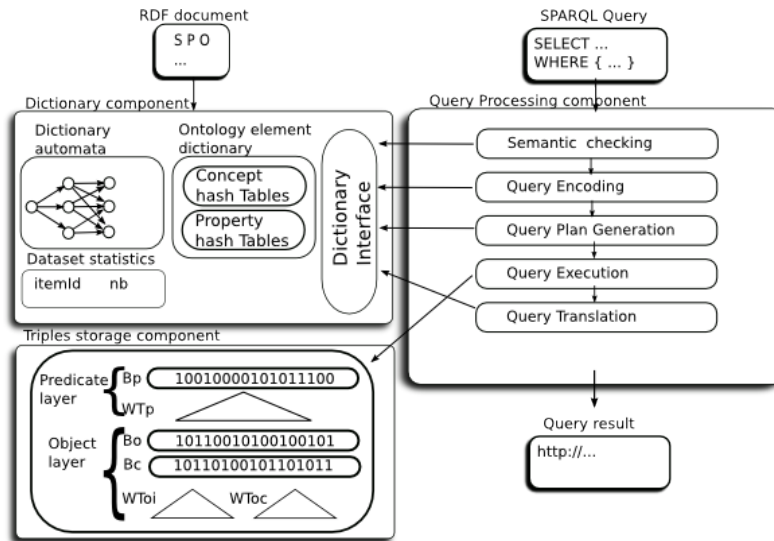


Fig. 1. WaterFowl's architecture

3 Update strategy

As previously mentioned, one of the benefits of using wavelet trees in our system is the ability to compress drastically the data while being able to query it without decompression. The main drawback of such SDS lies in its requirement to pre-compute and store some small but necessary extra informations. More formally, the bitmaps used in the inner construction of the wavelet trees requires $n + o(n)$ bits of storage space (the original bit array and an $o(n)$ auxiliary structure) to support **rank** and **select** in constant time. Note that while the implementation of **rank** is simple and practical, this is not the case for **select** which should be avoided whenever possible, *e.g.*, in our SPARQL query translations. Recently, some attempts were done in order to provide faster and smaller structure for **select** [4]. These precomputed auxiliary informations are used during query processing in order to get constant time complexity. This is why, by definition, the bitmaps are static and cannot be updated. In our context, it implies an immutable RDF store (which is quite restrictive).

In order to overcome this issue, we propose an update strategy using the inner tree structure of the wavelet trees. First, recall that there are mainly two

implementations of the wavelet trees: with and without pointers. On one hand, the implementation without pointers uses less memory space and provides better time performances. On the other hand, any modification to the represented sequence implies a full reconstruction of the wavelet tree; while, in the case where pointers are used, only a subset of the nodes (and the corresponding bitmaps) have to be rebuilt which in our first experiments is much faster than a total reconstruction. A second important point of our approach is based on the fact that in wavelet trees, each bit of an encoded entry specifies a path in the tree. That is, the instance data only influence the size of nodes but not their placement in the tree. Considering these two assumptions together with the huge amount of data we want to handle, our strategy supports a hybrid approach based on the natural definition of a tree. Indeed, a tree can be defined recursively as a node with a sequence of children which are themselves trees. Our hybrid wavelet tree is then defined as a node representing a wavelet tree without pointers of height k and a sequence of 2^k children which are themselves hybrid wavelet trees. In practice, considering a querying scenario composed of read (e.g., select) and write (e.g., add, delete, update) operations, for performance purpose the hybrid wavelet tree can adapt its composition to the scenario by minimizing and maximizing the numbers of pointers in the depth traversal of respectively a read and write operation. Note that this approach of cracking the database into manageable pieces is reminiscent to dynamic indexing solution presented in [2].

4 Conclusion

This poster exploits available wavelet tree implementations to address the issue of updating an ABox. We have already implemented a prototype of the WaterFowl system and of the updating system. They so far provide interesting performance results but we have yet to test with real use cases. This will enable us to observe practical modifications and to study their efficiency. These observations should provide directions for optimizations. Our future work, we will consist in the development of two new components. A first one will address updates at the schema level, *i.e.*, insertions or removals of concepts and properties of the underlying ontology. The second one will consider the partitioning of a data set over a machine cluster together with both ABox and TBox updates.

References

1. Olivier Curé, Guillaume Blin, Dominique Revuz, and David Célestin Faye. Waterfowl: A compact, self-indexed and inference-enabled immutable rdf store. In *ESWC*, pages 302–316, 2014.
2. Stratos Idreos, Martin L. Kersten, and Stefan Manegold. Database cracking. In *CIDR*, pages 68–78, 2007.
3. Gonzalo Navarro. Wavelet trees for all. *J. Discrete Algorithms*, 25:2–20, 2014.
4. Gonzalo Navarro and Eliana Provedel. Fast, small, simple rank/select on bitmaps. In Ralf Klasing, editor, *Experimental Algorithms*, volume 7276 of *Lecture Notes in Computer Science*, pages 295–306. Springer Berlin Heidelberg, 2012.

Linking Historical Data on the Web^{*}

Valeria Fionda¹, Giovanni Grasso²

¹ Department of Mathematics, University of Calabria, Italy
fionda@mat.unical.it

² Department of Computer Science, Oxford University, UK
giovanni.grasso@cs.ox.ac.uk

Abstract. Linked Data today available on the Web mostly represent snapshots at particular points in time. The temporal aspect of data is mostly taken into account only by adding and removing triples to keep datasets up-to-date, thus neglecting the importance to keep track of the evolution of data over time. To overcome this limitation, we introduce the LINKHISDATA framework to automatize the creation and publication of linked historical data extracted from the Deep Web.

1 Introduction

Most of the linked datasets today available on the Web offer a static view over the data they provide ignoring their evolution over time. Indeed, the temporal aspect is only considered by adding and removing information to keep datasets up-to-date [4]. However, while some information is intrinsically static because universally valid (e.g., the author of Harry Potter books is J. K. Rowling), a large portion of data are only valid from a particular point in time on (e.g., the first Harry Potter novel is available in Braille edition from May 1998) or are valid for a certain period (e.g., the price of the children paperback edition of the first Harry Potter book was £3.85 between the 10th and the 15th May 2014).

The possibility to include the temporal validity of RDF data opens the doors to the creation of new datasets in several domains by keeping track of the evolution of information over time (e.g., books prices, currency exchange rate) and publishing it as linked historical data. In turn, the availability of historical datasets would stimulate and enable a large number of applications (e.g., data analytics) and research areas [1]. Unfortunately, a recent work [6] showed that the amount of linked data with temporal information available on the Web is very small and to the best of our knowledge there are no proposals aimed at publishing historical dataset on the Web of Linked Data. In this paper we propose LINKHISDATA, a configurable framework to automatize the creation and publication of linked historical data extracted from the Deep Web. In particular, the focus is on extracting and making persistent transient information (e.g., the price of a book on a certain day) before becoming no longer accessible.

^{*} V. Fionda was supported by the European Commission, the European Social Fund and the Calabria region. G. Grasso was supported by the European Commission's Seventh Framework Programme (FP7/2007–2013) from ERC grant agreement DIA-DEM, no. 246858.

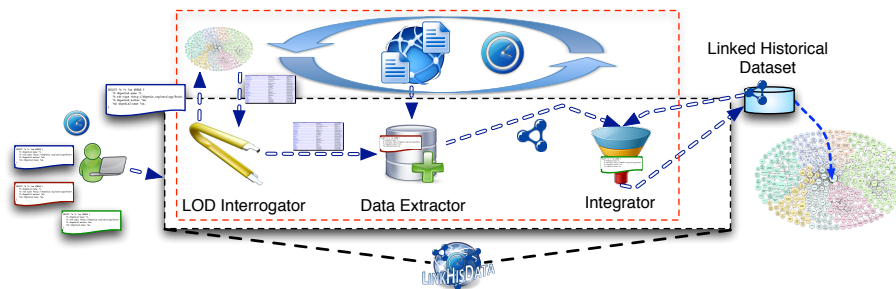


Fig. 1: An overview of the components of the LINKHISDATA framework.

2 The LINKHISDATA framework

LINKHISDATA (Linked Historical Data) is a configurable framework that builds on well-established semantic technologies and tools (e.g., SPARQL, RDF), as well as languages for Deep Web data extraction that have already been successfully employed by the LOD community (i.e., XPath [3, 5]).

Fig. 1 shows the LINKHISDATA architecture. Its main components are: the LOD Interrogator, the Data Extractor, the Integrator, and the Linked Historical Dataset (LHD). The LOD Interrogator uses the SPARQL query and the endpoint address provided in input by the user to retrieve from the Web of Linked Data entities' URIs and related information. These data feed the Data Extractor that runs the XPath wrapper, again provided in input by the user, to extract transient information from the Deep Web directly into RDF. XPath is a modern wrapping language able to execute actions (e.g., click, form filling) and schedule periodic extraction tasks. The query and the wrapper may share variable names so that the wrapper is instantiated with the actual values provided by the LOD Interrogator. The RDF data generated by the Data Extractor populates LHD and are published on the Web. A single execution of the extraction and publication process produces RDF data with temporal information that represent a snapshot at the time of extraction. To produce historical data, the whole process is repeated at the frequency set by the user and for each repetition the Integrator is responsible for the integration of the fresh triples (with temporal validity set to the time of extraction) with the data already stored in LHD (whose validity dates back to a prior extraction) by executing the (set of) SPARQL query provided in input. Different inputs supplied by the user configure LINKHISDATA to extract and publish historical datasets in different domains.

3 LINKHISDATA: Book Price Example

We instantiate the LINKHISDATA framework for the extraction and publication of linked historical data about books prices extracted from Barnes&Noble (www.bn.com). The next query retrieves from DBpedia books and relative attributes³:

```
SELECT ?b ?t ?ab ?an WHERE {
  ?b dbp:name ?t. ?b rdf:type dbo:Book. ?b dbp:author ?ab. ?ab foaf:name ?an. }
```

³ The prefixes used in the paper are taken from www.prefix.cc

```

1 doc("www.bn.com")//*[ @id='keyword' ]/{?t ?an}///*[ @id='quick-search' ]/{ "Books" /}
2 //*[ @id='quick-search-1' ]//button/{click /}/li#search-result
3 [jarowrinkler(/li#title, ?t)=1][jarowrinkler(/li#auth, ?an)>.7]/{click /}/
4 /html:<(schema:Book(isbn))> [.: <owl:sameAs(?b)>]
5 [./:[starts-with(., "ISBN")][1]/text()[2]: <schema:isbn=string(.)>]
6 [.:<schema:author(schema:Person(?authorName))> [.: <owl:sameAs(?a)>]]
7 [.: <schema:offers(schema:Offer lhd:HistoricalEntity)>]
8 [? ./:[@itemprop="price"][1]: <schema:price=substring(.,2)> ]
9 [? ./:[@itemprop="price"][1]: <schema:priceCurrency=substring(.,1,1)> ]
10 [? ..:<schema:validFrom = now()> ]

```

Fig. 2: OXPath RDF wrapper

This query returns a set of tuples containing the URI of a book (?b), its title (?t), the URI of its author (?ab) and the author’s name (?an). For instance, for the book “The Firm” of J. Grisham the retrieved tuple is $\langle \text{dbpedia:The_Firm_}(novel), \text{“The Firm”}, \text{dbpedia:John_Grisham}, \text{“John Grisham”} \rangle$. The remainder of the example uses the values of this tuple to instantiate the various components.

The tuples returned by the LOD Interrogator constitute the input for the Data Extractor which runs the OXPath wrapper shown in Figure 2 (for space reasons some parts are simplified or omitted). Here, the variables (e.g., ?b) refer to the corresponding ones in the LOD Interrogator SPARQL query. We assume some familiarity with XPath to illustrate the wrapper. It comprises three parts: (i) navigation to the pages containing relevant data, (ii) identification of data to extract, and (iii) RDF output production. In our example, we use types and properties from schema.org (e.g., Book, Offer, price). For instance, for our example tuple about “The Firm”, the wrapper produces the following RDF output:

```

lhd:b9780440245926 a schema:Book ; owl:sameAs dbpedia:The_Firm_(novel);
  schema:isbn "9780440245926"; schema:author lhd:John_Grisham;
  schema:offers lhd:off_123.
lhd:off_123 a schema:Offer,lhd:HistoricalEntity ; schema:price "9.21";
  schema:priceCurrency "$"; schema:validFrom "2014-06-28".
lhd:John_Grisham a schema:Person ; owl:sameAs dbpedia:John_Grisham.

```

The wrapper encodes part (i) in lines 1–2. Firstly, the website is loaded, then, the search form is filled with (“The Firm John Grisham”) i.e., book title and author name, plus the search is restricted to the book category. Finally, the submit button is clicked and all the books on the result page are selected by the expression //li.search-result. However, many of the results may not refer to the book of interest (e.g., also collections/sets of books containing it are retrieved) and it is absolutely crucial to identify the correct entities as these will be linked back to the original entities in DBpedia. We address this problem (part (ii)) by using the Jaro-Winkler distance [2] to match author name and book title. This metric has been widely used in record linkage and performs particularly well on person and entity’s names. Our wrapper (line 3) demands for a perfect matching on the title and a significant similarity (>0.7) on the author’s name to deal with different variations (e.g., “J. Grisham” or “John Ray Grisham, Jr”).

This strategy prevents our framework from overtaxing the site by extracting (a possibly huge quantity of) data on non-interesting books, that would be only later discarded by a costly post-processing linking phase. For example, for “The Firm” we correctly identify only 2 books out of the original 22 results.

Part (iii) is realized by visiting the detail page of each selected book via the action `{click/}`. An *RDF extraction marker* is used to create a `schema:Book` instance (line 4) and by explicitly linking it to the corresponding DBpedia URI via `owl:sameAs`. The unique URI for the created book instance relies on the functional dependency to its `isbn` (`schema:Book(isbn)`). This ensures that this URI will be always the same within the subsequent extractions, allowing to refer to the right entity in the integration phase. The wrapper also creates one linked data entity for the book author (line 6) and one for the offer (line 7), respectively. The former is of type `schema:Person`; its URI is created on the basis of the author name (*?an*) and is linked via `owl:sameAs` to the author on DBpedia. The latter is of type `schema:Offer` and `lhd:HistoricalEntity`, type used to mark entities with temporal validity. Its URI is randomly generated to ensure different URIs for consecutive extractions. Some properties of the offer are also extracted (e.g., the `schema:price`) and the `schema:validityFrom` is set to `now()`, the current date.

The Integrator takes in input, for each book (*?b*), the set of triples \mathcal{T} as produced by the Data Extractor and it is responsible for integrating them with those already present in LHD. In particular, it deals with entities having a temporal validity, e.g., book offers and their prices in our example. Each book in LHD may have associated several offers that represent the evolution of the price over time. However, only one of them provides the current selling price (i.e., the one not having a `schema:validThrough` triple). Therefore, for each book (*?b*), the Integrator instantiates the query template (provided by the user) to retrieve such current price (*?p*) and its corresponding offer (*?o*) from the historical dataset. If *?b* is not already present in the historical dataset, the triples in \mathcal{T} are added to it. Otherwise, the Integrator compares the current price (*?p*) with the price of the offer in \mathcal{T} (freshly extracted). If they differ, the price validity is updated by adding to the dataset both the triple (*?o*, `schema:validThrough`, `now()`) and the offer in \mathcal{T} . Together they provide a new piece of historical information.

References

1. O. Alonso, J. Strötgen, R. A. Baeza-Yates, and M. Gertz. Temporal information retrieval: Challenges and opportunities. In *TWAW*, volume 813, pages 1–8, 2011.
2. W.W. Cohen, P.D. Ravikumar, and S.E. Fienberg. A comparison of string distance metrics for name-matching tasks. In *IIWeb*, pages 73–78, 2003.
3. T. Furche, G. Gottlob, G. Grasso, C. Schallhart, and A.J. Sellers. OXPath: A language for scalable data extraction, automation, and crawling on the deep web. *VLDB J.*, 22(1):47–72, 2013.
4. T. Käfer, A. Abdelrahman, J. Umbrich, P. O’Byrne, and A. Hogan. Observing linked data dynamics. In *ESWC*, pages 213–227, 2013.
5. J. Lehmann, T. Furche, G. Grasso, A.C. Ngonga Ngomo, C. Schallhart, and C. et al. Unger. deqa: Deep web extraction for question answering. In *ISWC*, 2012.
6. A. Rula, M. Palmonari, A. Harth, S. Stadtmüller, and A. Maurino. On the diversity and availability of temporal information in linked open data. In *ISWC*, 2012.

User driven Information Extraction with LODIE

Anna Lisa Gentile and Suvodeep Mazumdar

Department of Computer Science, University of Sheffield, UK
{a.gentile, s.mazumdar}@sheffield.ac.uk

Abstract. Information Extraction (IE) is the technique for transforming unstructured or semi-structured data into structured representation that can be understood by machines. In this paper we use a user-driven Information Extraction technique to wrap entity-centric Web pages. The user can select concepts and properties of interest from available Linked Data. Given a number of websites containing pages about the concepts of interest, the method will exploit (i) recurrent structures in the Web pages and (ii) available knowledge in Linked data to extract the information of interest from the Web pages.

1 Introduction

Information Extraction transforms unstructured or semi-structured text into structured data that can be understood by machines. It is a crucial technique towards realizing the vision of the Semantic Web. Wrapper Induction (WI) is the task of automatically learning wrappers (or extraction patterns) for a set of homogeneous Web pages, i.e. pages from the same website, generated using consistent templates¹. WI methods [1,2] learn a set of rules enabling the systematic extraction of specific data records from the homogeneous Web pages. In this paper we adopt a user driven paradigm for IE and we perform on demand extraction on entity-centric webpages. We adopt our WI method [2,3] developed within the LODIE (Linked Open Data for Information Extraction) framework [4]. The main advantage of our method is that does not require manually annotated pages. The training examples for the WI method are automatically generated exploiting Linked Data.

2 State of the Art

Using WI to extract information from structured Web pages has been studied extensively. Early studies focused on the DOM-tree representation of Web pages and learn a template that wrap data records in HTML tags, such as [1,5,6]. Supervised methods require manual annotation on example pages to learn wrappers for similar pages [1,7,8]. The number of required annotations can be drastically reduced by annotating pages from a specific website and then adapting the learnt

¹ For example, a yellow page website will use the same template to display information (e.g., name, address, cuisine) of different restaurants.

rules to previously unseen websites of the same domain [9,10]. Completely unsupervised methods (e.g. RoadRunner [11] and EXALG [12]) do not require any training data, nor an initial extraction template (indicating which concepts and attributes to extract), and they only assume the homogeneity of the considered pages. The drawback of unsupervised methods is that the semantic of produced results is left as a post-process to the user. Hybrid methods [2] intend to find a tradeoff with these two limitations by proposing a supervised strategy, where the training data is automatically generated exploiting Linked Data. In this work we perform IE using the method proposed in [2,3] and follow the general IE paradigm from [4].

3 User-driven Information Extraction

In LODIE we adopt a user driven paradigm for IE. As first step, the user must define her/his information need. This is done via a visual exploration of linked data (Figure 1).

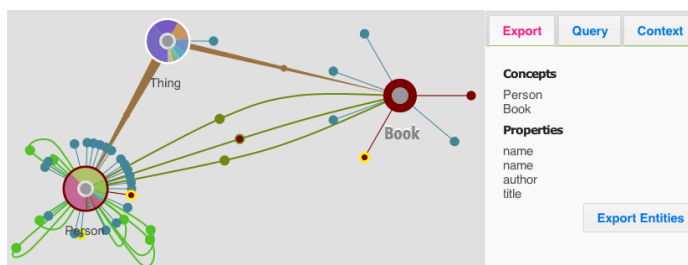


Fig. 1: Exploring linked data to define user need, by selecting concepts and attributes to extract. Here the user selected the concept *Book* and the attributes *title* and *author*. As *author* is a datatype attribute, of type *Person*, the attribute *name* is chosen.

The user can explore underlying linked data using the Affective Graphs visualization tool [13] and select concepts and properties she/he is interested in (a screenshot is shown in Figure 1). These concepts and properties get added to the side panel. Once the selection is finished, she/he can start the IE process. The IE starts with a dictionary generation phase. A dictionary $d_{i,k}$ consists of values for the attribute $a_{i,k}$ of instances of concept c_i . Noisy entries in the dictionaries are removed using a cleaning procedure detailed in [3]. As a running example we will assume the user wants to extract *title* and *author* for the concept *Book*. We retrieve from the Web k websites containing entity-pages of the concept types selected by the user, and save the pages $W_{c_i,k}$. Following the *Book* example, Barnes&Noble² or AbeBooks³ websites can be used, and pages collected in $W_{book,barnesandnoble}$ and $W_{book,abebooks}$.

For each $W_{c_i,k}$ we generate a set of extraction patterns for every attribute. In our example we will produce 4 sets of patterns, one per each website and

² <http://www.barnesandnoble.com/>

³ <http://www.abebooks.co.uk>

attribute. To produce the patterns we (i) use our dictionaries to generate brute-force annotations on the pages in $W_{c_i,k}$ and then (ii) use statistical (occurrence frequency) and structural (position of the annotations in the webpage) clues to choose the final extraction patterns.

Briefly, a page is transformed to a simplified page representation P_{c_i} : a collection of pairs $\langle \textit{xpath}^4, \textit{text value} \rangle$. Candidates are generated matching the dictionaries $d_{i,k}$ against possible *text values* in P_{c_i} (Figure 2).

```

/HTML[1]/BODY[1]/DIV[2]/DIV[2]/DIV[2]/DIV[1]/H2[1]/text()[1] breaking dawn
/HTML[1]/BODY[1]/DIV[2]/DIV[2]/DIV[2]/DIV[4]/DIV[1]/H2[1]/EM[1]/text()[1] breaking dawn
/HTML[1]/BODY[1]/DIV[2]/DIV[2]/DIV[2]/DIV[4]/TABLE[10]/TBODY[1]/TR[1]/TD[3]/B[1]/A[1]/text()[1] break-
ing dawn
/HTML[1]/BODY[1]/DIV[2]/DIV[2]/DIV[2]/DIV[4]/TABLE[1]/TBODY[1]/TR[1]/TD[3]/B[1]/A[1]/text()[1] break-
ing dawn
/HTML[1]/BODY[1]/DIV[2]/DIV[2]/DIV[2]/DIV[4]/TABLE[2]/TBODY[1]/TR[1]/TD[3]/B[1]/A[1]/text()[1] break-
ing dawn
/HTML[1]/BODY[1]/DIV[2]/DIV[2]/DIV[2]/DIV[4]/TABLE[3]/TBODY[1]/TR[1]/TD[3]/B[1]/A[1]/text()[1] break-
ing dawn
/HTML[1]/BODY[1]/DIV[2]/DIV[2]/DIV[2]/DIV[4]/TABLE[6]/TBODY[1]/TR[1]/TD[3]/B[1]/A[1]/text()[1] break-
ing dawn
/HTML[1]/BODY[1]/DIV[2]/DIV[2]/DIV[2]/DIV[4]/TABLE[8]/TBODY[1]/TR[1]/TD[3]/B[1]/A[1]/text()[1] break-
ing dawn
/HTML[1]/BODY[1]/DIV[2]/DIV[2]/DIV[3]/DIV[3]/UL[1]/LI[2]/A[1]/text()[1] the host
/HTML[1]/BODY[1]/DIV[2]/DIV[2]/DIV[3]/DIV[3]/UL[1]/LI[5]/A[1]/text()[1] new moon

```

Fig. 2: Example of candidates for book title for a Web page on the book “Breaking Dawn”, from the website AbeBooks.

Final patterns are chosen amongst the candidates exploiting frequency information and other heuristics. Details of the method can be found in [2,3]. In the running example, higher scoring patterns for extracting book title from AbeBooks website are shown in Figure 3.

```

/HTML[1]/BODY[1]/DIV[2]/DIV[2]/DIV[2]/DIV[1]/H2[1]/text()[1] 329.0
/HTML[1]/BODY[1]/DIV[2]/DIV[2]/DIV[2]/DIV[4]/DIV[1]/H2[1]/EM[1]/text()[1] 329.0

```

Fig. 3: Extraction patterns for book titles from AbeBooks website.

All extraction patterns are then used to extract target values from all $W_{c_i,k}$. Results are produced as linked data, using the concept and properties initially selected by the user for representation, and made accessible to the user via an exploration interface (Figure 4), implemented using Simile Widgets⁵.

A video showing the proposed system used with the running *Book* example can be found at <http://staffwww.dcs.shef.ac.uk/people/A.L.Gentile/demo/iswc2014.html>.

4 Conclusions and future work

In this paper we describe the LODIE approach to perform IE on user defined extraction tasks. The user is prompted a visual tool to explore available linked data and choose concepts for which she/he wants to mine additional material from the Web. We learn extraction patterns to wrap relevant websites and return structured results to the user.

⁴ <http://www.w3.org/TR/xpath/>

⁵ <http://www.simile-widgets.org/>

Information Extraction results

4000 Book filtered from 19948 originally (Reset All Filters)

EXTRACTED RESULTS [TABLE VIEW](#)

WEBSITE	ORIGINAL WEB PAGE (cached)	TITLE	AUTHOR
abebooks	http://lodie.co.uk/ontology/book/abebooks/1081.htm	the orchestra: orchestral techniques and combinations	prout, ebenezer
abebooks	http://lodie.co.uk/ontology/book/abebooks/0265.htm	the pathfinder: how to choose or change your career for a lifetime of satisfaction and success	lore, nicholas
abebooks	http://lodie.co.uk/ontology/book/abebooks/0876.htm	the shadow of the sun : my african life	hochschild, adam, theroux, paul, kapuscinski, ryszard; random house vintage books, and kapuscinski, ryszard
abebooks	http://lodie.co.uk/ontology/book/abebooks/0411.htm	mayflower : a story of courage, community, and war	mcculough, david willis, mann, charles c., goodwin, doris kearns, philbrick, nathaniel, and mcculough, david
abebooks	http://lodie.co.uk/ontology/book/abebooks/0493.htm	full catastrophe living: using the wisdom of your body and mind to face stress, pain, and illness	kabat-zinn, jon

Fig. 4: Exploration of results produced by the IE method

References

1. Kushmerick, N.: Wrapper Induction for information Extraction. In: IJCAI97. (1997) 729–735
2. Gentile, A.L., Zhang, Z., Augenstein, I., Ciravegna, F.: Unsupervised wrapper induction using linked data. In: Proc. of the seventh international conference on Knowledge capture. K-CAP '13, New York, NY, USA, ACM (2013) 41–48
3. Gentile, A.L., Zhang, Z., Ciravegna, F.: Self training wrapper induction with linked data. In: Proceedings of the 17th International Conference on Text, Speech and Dialogue (TSD 2014). (2014) 295–302
4. Ciravegna, F., Gentile, A.L., Zhang, Z.: Lodie: Linked open data for web-scale information extraction. In: SWAIE. (2012) 11–22
5. Muslea, I., Minton, S., Knoblock, C.: Hierarchical wrapper induction for semistructured information sources. *Autonomous Agents and Multi-Agent Systems* (2001) 1–28
6. Soderland, S.: Learning information extraction rules for semi-structured and free text. *Mach. Learn.* **34**(1-3) (February 1999) 233–272
7. Muslea, I., Minton, S., Knoblock, C.: Active Learning with Strong and Weak Views: A Case Study on Wrapper Induction. *IJCAI'03 8th international joint conference on Artificial intelligence* (2003) 415–420
8. Dalvi, N., Kumar, R., Soliman, M.: Automatic wrappers for large scale web extraction. *Proc. of the VLDB Endowment* **4**(4) (2011) 219–230
9. Wong, T., Lam, W.: Learning to adapt web information extraction knowledge and discovering new attributes via a Bayesian approach. *Knowledge and Data Engineering, IEEE* **22**(4) (2010) 523–536
10. Hao, Q., Cai, R., Pang, Y., Zhang, L.: From One Tree to a Forest : a Unified Solution for Structured Web Data Extraction. In: *SIGIR 2011*. (2011) 775–784
11. Crescenzi, V., Mecca, G.: Automatic information extraction from large websites. *Journal of the ACM* **51**(5) (September 2004) 731–779
12. Arasu, A., Garcia-Molina, H.: Extracting structured data from web pages. In: *Proc. of the 2003 ACM SIGMOD international conference on Management of data, ACM* (2003) 337–348
13. Mazumdar, S., Petrelli, D., Elbedweihy, K., Lanfranchi, V., Ciravegna, F.: Affective graphs: The visual appeal of linked data. *Semantic Web–Interoperability, Usability, Applicability*. IOS Press (to appear, 2014) (2013)

QALM: a Benchmark for Question Answering over Linked Merchant Websites Data

Amine Hallili¹, Elena Cabrio^{2,3}, and Catherine Faron Zucker¹

¹ Univ. Nice Sophia Antipolis, CNRS, I3S, UMR 7271, Sophia Antipolis, France
`amine.hallili@inria.fr`; `faron@unice.fr`

² INRIA Sophia Antipolis Méditerranée, Sophia Antipolis, France
`elena.cabrio@inria.fr`

³ EURECOM, Sophia Antipolis, France

Abstract. This paper presents a benchmark for training and evaluating Question Answering Systems aiming at mediating between a user, expressing his or her information needs in natural language, and semantic data in the commercial domain of the mobile phones industry. We first describe the RDF dataset we extracted through the APIs of merchant websites, and the schemas on which it relies. We then present the methodology we applied to create a set of natural language questions expressing possible user needs in the above mentioned domain. Such question set has then been further annotated both with the corresponding SPARQL queries, and with the correct answers retrieved from the dataset.

1 Introduction

The evolution of the e-commerce domain, especially the Business To Client (B2C), has encouraged the implementation and the use of dedicated applications (e.g. Question Answering Systems) trying to provide end-users with a better experience. At the same time, the user's needs are getting more and more complex and specific, especially when it comes to commercial products whose questions concern more often their technical aspects (e.g. price, color, seller, etc.). Several systems are proposing solutions to answer to these needs, but many challenges have not been overcome yet, leaving room for improvement. For instance, federating several commercial knowledge bases in one knowledge base has not been accomplished yet. Also, understanding and interpreting complex natural language questions also known as n-relation questions seems to be one of the ambitious topics that systems are currently trying to figure out.

In this paper we present a benchmark for training and evaluating Question Answering (QA) Systems aiming at mediating between a user, expressing his or her information need in natural language, and semantic data in the commercial domain of the mobile phone industry. We first describe the RDF dataset that we have extracted through the APIs of merchant sites, and the schemas on which it relies. We then present the methodology we applied to create a set of natural language questions expressing possible user needs in the above mentioned domain.

Such question set has then be further annotated both with the corresponding SPARQL queries, and with the correct answers retrieved from the dataset.

2 A Merchant Sites Dataset for the Mobile Phones Industry

This section describes the QALM (Question Answering over Linked Merchant websites) ontology (Section 2.1), and the RDF dataset (Section 2.2) we built by extracting a sample of data from a set of commercial websites.

2.1 QALM Ontology

The QALM RDF dataset relies on two ontologies: the Merchant Site Ontology (MSO) and the Phone Ontology (PO). Together they build up the QALM Ontology.⁴ MSO models general concepts of merchant websites, and it is aligned to the commercial part of the *Schema.org* ontology. MSO is composed of 5 classes: `mso:Product`, `mso:Seller`, `mso:Organization`, `mso:Store`, `mso:ParcelDelivery`, and of 29 properties (e.g. `mso:price`, `mso:url`, `mso:location`, `mso:seller`) declared as subclasses and subproperties of Schema.org classes and properties. We added to them multilingual labels (both in English and in French), that can be exploited by QA systems in particular for property identification in the question interpretation step. We relied on WordNet synonyms [2] to extract as much labels as possible. For example, the property `mso:price` has the following English labels: “price”, “cost”, “value”, “tariff”, “amount”, and the following French labels: “prix”, “coût”, “coûter”, “valoir”, “tarif”, “s’élever”.

PO is a domain ontology modeling concepts specific to the phone industry. It is composed of 7 classes (e.g. `po:Phone`, `po:Accessory`) which are declared as subclasses of `mso:Product`, and of 35 properties (e.g. `po:handsetType`, `po:operatingSystem`, `po:phoneStyle`).

2.2 QALM RDF Dataset

Our final goal is to build a unified RDF dataset integrating commercial product descriptions from various e-commerce websites. In order to achieve this goal, we analyze the web services of the e-commerce websites regardless of their type (either SOAP or REST). To feed our dataset, we create a mapping between the remote calls to the web services and the ontology properties, that we store in a separate file for reuse. In particular, we built the QALM RDF dataset by extracting data from eBay⁵ and BestBuy⁶ commercial websites through BestBuy Web service and eBay API. The extracted raw data is transformed into RDF triples by applying the above described mapping between the QALM ontology

⁴ Available at www.i3s.unice.fr/qalm/ontology

⁵ <http://www.ebay.com/>

⁶ <http://www.bestbuy.com/>

and the API/web service. For instance, the method `getPrice()` in the eBay API is mapped to the property `mso:price` in the QALM ontology. Currently, the QALM dataset comprises 500000 product descriptions and up to 15 millions triples extracted from eBay and BestBuy.⁷

3 QALM Question Set

In order to train and to evaluate a QA system mediating between a user and semantic data in the QALM dataset, a set of questions representing users requests in the phone industry domain is required. Up to our knowledge, the only available standard sets of questions to evaluate QA systems over linked data are the ones released by the organizers of the QALD (Question Answering over Linked Data) challenges.⁸ However such questions are over the English DBpedia dataset⁹, and therefore cover several topics. For this reason, we created a set of natural language questions for the specific commercial domain of the phone industry, following the guidelines described by the QALD organizers for the creation of their question sets [1]. More specifically, these questions were created by 12 external people (students and researchers in other groups) with no background in question answering, in order to avoid a bias towards a particular approach. To accomplish the task of question creation, each person was given *i*) the list of the product types present in the QALM dataset (mainly composed of IT products as phones and accessories); *ii*) the list of the properties of the QALM ontology presented as product features in which they could be interested in; and they were asked to produce *i*) both 1-relation and 2-relation questions, and *ii*) at least 5 questions each. The questions were designed to present potential user questions and to include a wide range of challenges such as lexical ambiguities and complex syntactical structures. Such questions were then annotated with the corresponding SPARQL queries, and the correct answers retrieved from the dataset, in order to consider them as a reliable goldstandard for our benchmark.

The final question set comprises 70 questions; it is divided into a training set¹⁰ and a test set of respectively 40 and 30 questions. Annotations are provided in XML format, and according to QALD guidelines, the following attributes are specified for each question along with its ID: *aggregation* (indicates whether any operation beyond triple pattern matching is required to answer the question, e.g., counting, filtering, ordering), *answertype* (gives the answer type: resource, string, boolean, double, date). We also added the attribute *relations*, to indicate whether the question is connected to its answer through one or more properties of the ontology (values: 1, n). Finally, for each question the corresponding SPARQL query is provided, as well as the answers this query returns. Examples 1 and 2 show some questions from the collected question set, connected to their answers through 1 property or more than 1 property of the ontology, respectively. In

⁷ Available at www.i3s.unice.fr/QALM/qalm.rdf

⁸ <http://greententacle.techfak.uni-bielefeld.de/~cunger/qald/>

⁹ <http://dbpedia.org>

¹⁰ Available at www.i3s.unice.fr/QALM/training_questions.xml

particular, questions 14 and 50 from Example 2 require also to carry out some reasoning on the results, in order to rank them and to produce the correct answer.

Example 1. 1-relation questions.

id=36. *Give me the manufacturers who supply on-ear headphones.*

id=52. *What colors are available for the Samsung Galaxy 5 ?*

id=61. *Which products of Alcatel are available online?*

Example 2. n-relations questions.

id=14. *Which cell phone case (any manufacturer) has the most ratings?*

id=50. *What is the highest camera resolution of phones manufactured by Motorola?*

id=58. *I would like to know in which stores I can buy Apple phones.*

4 Conclusions and Ongoing Work

This paper presented a benchmark to train and test QA systems, composed of *i*) the QALM ontologies; *ii*) the QALM RDF dataset of product descriptions extracted from eBay and BestBuy; and *iii*) the QALM Question Set, containing 70 natural language questions in the commercial domain of phones and accessories.

As for future work, we will consider aligning the QALM ontology to the *GoodRelations* ontology to fully cover the commercial domain, and to benefit from the semantics captured in this ontology. We also consider improving the QALM RDF dataset by *i*) extracting RDF data from additional commercial websites that provide web services or APIs; and *ii*) directly extracting RDF data in the Schema.org ontology from commercial websites whose pages are automatically generated with Schema.org markup (e.g. Magento, OSCommerce, Genesis2.0, Prestashop), to extend the number of addressed commercial websites.

In parallel, we are currently developing the SynchroBot QA system [3], an ontology-based chatbot for the e-commerce domain. We will evaluate it by using the proposed QALM benchmark.

Acknowledgements

We thank Amazon, eBay and BestBuy for contributing to this work by sharing with us public data about their commercial products. The work of E. Cabrio was funded by the French Government through the ANR-11-LABX-0031-01 program.

References

1. Cimiano, P., Lopez, V., Unger, C., Cabrio, E., Ngomo, A.C.N., Walter, S.: Multilingual question answering over linked data (qald-3): Lab overview. In: CLEF. pp. 321–332 (2013)
2. Fellbaum, C.: WordNet: An Electronic Lexical Database. Bradford Books (1998)
3. Hallili, A.: Toward an ontology-based chatbot endowed with natural language processing and generation. In: Proc. of ESSLLI 2014 - Student Session, Poster paper (2014)

GeoTriples: a Tool for Publishing Geospatial Data as RDF Graphs Using R2RML Mappings

Kostis Kyzirakos¹, Ioannis Vlachopoulos², Dimitrianos Savva²,
Stefan Manegold¹, and Manolis Koubarakis²

¹ Centrum Wiskunde & Informatica, Amsterdam, The Netherlands
{firstname.lastname}@cwi.nl

² National and Kapodistrian University of Athens, Greece
{johnvl,dimis,koubarak}@di.uoa.gr

Abstract. In this paper we present the tool GeoTriples that allows the transformation of Earth Observation data and geospatial data into RDF graphs, by using and extending the R2RML mapping language to be able to deal with the specificities of geospatial data. GeoTriples is a semi-automated tool that transforms geospatial information into RDF following the state of the art vocabularies like GeoSPARQL and stSPARQL, but at the same time it is not tightly coupled to a specific vocabulary.

Keywords: Linked Geospatial Data, data publishing, GeoSPARQL, stSPARQL

1 Introduction

In the last few years there has been significant effort on publishing EO and geospatial data sources as linked open data. However, the problem of publishing geospatial data sources into RDF graphs using a generic and extensible framework has received little attention as it has only recently emerged. Instead, scripting methods, that were adapted to the subject, were employed mostly for this task, such as custom python scripts developed in project TELEIOS³. However, some work towards developing automated methods for translating geospatial data into RDF has been presented in the latest LGD Workshop⁴. In this paper we present the tool GeoTriples that allows the transformation of geospatial data stored in spatially-enabled relational databases and raw files. It is implemented as an extension to the D2RQ platform⁵ [1] and goes beyond the state of the art by extending the R2RML mapping language⁶ to deal with the specificities of geospatial data. GeoTriples uses GeoSPARQL⁷ as the target vocabulary but the user is free to use any vocabulary she finds appropriate.

³<http://www.earthobservatory.eu>

⁴<http://www.w3.org/2014/03/lgd>

⁵<http://d2rq.org>

⁶<http://www.w3.org/TR/r2rml/>

⁷<http://www.opengeospatial.org/standards/geosparql/>

NAME	TYPE	WIDTH
Zeitlbach	stream	1
Mangfall	river	25
Triftbach	canal	10

```
osm_w:1 rdf:type geo:Feature ;
      osm_ont:hasName "Mangfall"^^xsd:string ;
      geo:hasGeometry osm_g:1 .
osm_g:1 rdf:type geo:Geometry ;
      geo:dimension "2"^^xsd:integer .
```

(a) Example data from an ESRI shapefile (b) Expected RDF triples about Mangfall

```
_:osm
  rr:logicalTable [ rr:tableName "'osm'"; ];
  rr:subjectMap [
    rr:class geo:Feature;
    rr:template "http://data.example.com/osm-waterways/
      Feature/id/{'gid'}"; ];
  rr:predicateObjectMap [
    rr:predicate osm:hasName;
    rr:objectMap [ rr:datatype xsd:string;
      rr:column "'NAME'"; ]; ];
  rr:predicateObjectMap [
    rr:predicate geo:hasGeometry ;
    rr:objectMap [
      rr:parentTriplesMap _:osmGeometry;
      rr:joinCondition [
        rr:child "gid";
        rr:parent "gid"; ]; ]; ].

_:osmGeometry
  rr:logicalTable [ rr:tableName "'osm'"; ];
  rr:subjectMap [
    rr:class geo:Geometry;
    rr:template "http://data.example.com/osm-waterways/
      Geometry/id/{'gid'}"; ];
  rr:predicateObjectMap [
    rr:predicate geo:dimension;
    rr:objectMap [
      rrx:transformation [
        rrx:function geof:dimension;
        rrx:argumentMap (
          [rr:column "'geom'"]
        ); ]; ]; ].
```

(c) Mapping of thematic information (d) Mapping of geometric information

Fig. 1: Examples of extended R2RML mappings for OSM

2 The Tool GeoTriples

GeoTriples⁸ is an open source tool, that takes as input geospatial data that are stored in a spatially enabled database, data that reside in raw files (e.g. ESRI shapefiles) or the results that derive from processing of the aforementioned data (e.g. a SciQL query over raster or array data). At a lower level, GeoTriples uses a connector for each type of input data that transparently accesses and processes the input data. It also consists of two main components: the mapping generator and the R2RML processor. The mapping generator creates automatically an R2RML mapping document from the input data source. The mapping is also enriched with subject and predicate object maps so that the RDF graph that will be produced follows the GeoSPARQL vocabulary. Geospatial information is modeled using a variety of data models (e.g., relational, hierarchical) and is made available using a variety of formats (e.g., ESRI shapefiles, KML documents). In order to deal with these specificities of geospatial information, we extended the R2RML language to allow the representation of a transformation function over the input data via an object map. In [2] we provide more information about our approach. Figure 1 presents an example of such a transformation. The R2RML processor is responsible for producing the desired RDF output by taking into account the mapping document generated, which is also optionally edited by the user. When the R2RML processor of GeoTriples detects an object map with a transformation function, it applies on the fly this function on the serialization of the geometry described in the subject map. However, if the input data source is a spatially enabled database, it generates the appropriate SQL queries that push these transformations to the underlying DBMS.

⁸<https://sourceforge.net/projects/geotriples/>

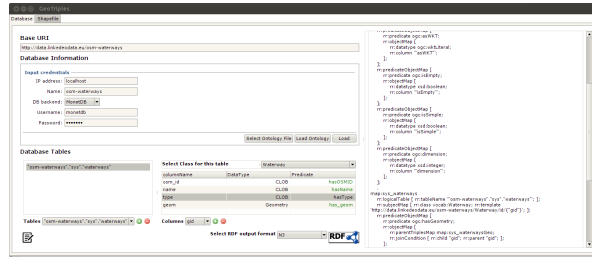


Fig. 2: The graphical user interface of GeoTriples

3 Using GeoTriples in a real-world scenario

In this section we present how we will demonstrate the tool GeoTriples in the context of a precision farming application that is developed by the FP7 EU project LEO⁹. The application combines traditional geospatial data with linked geospatial data for enhancing the quality of precision farming activities. Precision farming aims to solve numerous problems for farmers such as the minimization of the environmental pollution by fertilizers. For dealing with this issue, the farmers have to comply with many legal and technical guidelines that require the combination of information that resides in diverse information sources. In this section we present how linked geospatial data can form the knowledge base for providing solutions for this problem. We will publish the following datasets as RDF graphs using GeoTriples in order to use them in the precision farming application.

OpenStreetMap (OSM) is a collaborative project for publishing free maps of the world. OSM maintains a community-driven global editable map that gathers map data in a crowdsourcing fashion.

Talking Fields aims to increase the efficiency of agricultural production via precision farming by means of geo-information services integrating space and ground-based assets. It produces products for improved soil probing using satellite-based zone maps, and provide services for monitoring crop development through provision of biomass maps and yield estimates.

Natura 2000 is a European ecological network where national authorities submit a standard data form that describes each site and its ecology in order to be characterized as a Natura site.

Corine Land Cover (CLC) is an activity of the European Environment Agency that collects data regarding the land cover of European countries.

In this demo we will use GeoTriples in order to produce the R2RML mappings that dictate the process of generating the desired RDF output from the above data. Then, using the R2RML processor of GeoTriples, we translate the input data into RDF graphs and store the latter into the geospatial RDF store Strabon¹⁰ [3].

⁹<http://www.linkedeodata.eu>

¹⁰<http://www.strabon.di.uoa.gr/>

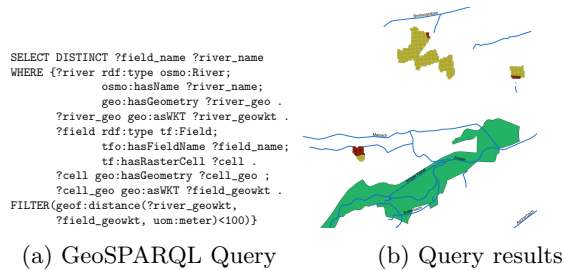


Fig. 3: Discover the parts of the agricultural fields that are close to rivers.

The user can use the graphical interface of GeoTriples that is displayed in Figure 2 for publishing these datasets as RDF graphs. At first the user defines the necessary credentials of the DBMS that stores the above datasets. Then, she selects the tables and the columns that contain the information that she wants to publish as RDF graphs. Optionally, an existing ontology may be loaded, in order to map the columns of the selected table to properties from the loaded ontology and map the instances generated from the rows to a specific class. Afterwards, GeoTriples generates automatically the R2RML mappings and presents them to the user. Finally, the user may either customize the generated mappings or proceed with the generation of the RDF graph.

A series of GeoSPARQL queries will be issued afterwards in Strabon for providing the precision farming application with information like the location of agricultural fields that are close to a river. This information allows the precision farming application to take into account legal restrictions regarding distance requirements when preparing the prescription maps that the farmers will utilize afterwards. In Figure 3a we present a GeoSPARQL query that discovers this information, and in Figure 3b we depict the query results.

4 Conclusions

In this paper we presented the tool GeoTriples that uses an extended form of R2RML mapping language to transform geospatial data into RDF graphs, and the GeoSPARQL ontology to properly express it. We demonstrate how GeoTriples is being used for publishing geospatial information that resides in different data sources for the realization of a precision farming application.

References

1. C. Bizer and A. Seaborne. D2RQ-treating non-RDF databases as virtual RDF graphs. In *Proceedings of the 3rd International Semantic Web Conference*, 2004.
2. K. Kostis, V. Ioannis, S. Dimitrianos, M. Stefan, and K. Manolis. Data models and languages for mapping EO data to RDF. Del. 2.1, FP7 project LEO, 2014.
3. K. Kyzirakos, M. Karpathiotakis, and M. Koubarakis. Strabon: A Semantic Geospatial DBMS. In *International Semantic Web Conference*, 2012.

New Directions in Linked Data Fusion

Jan Michelfeit¹ and Jindřich Mynarz²

¹ Faculty of Mathematics and Physics, Charles University in Prague, Czech Rep.
michelfeit@ksi.mff.cuni.cz,

² University of Economics, Prague, Czech Republic

Abstract. When consuming Linked Data from multiple sources, or from a data source after deduplication of entities, conflicting, missing or outdated values must be dealt with during data fusion in order to increase the usefulness and quality of the data. In this poster, we argue that the nature of Linked Data in RDF requires a more sophisticated approach to data fusion than the current Linked Data fusion tools provide. We demonstrate where they fall short on a real case of public procurement data fusion when dealing with property dependencies and fusion of structured values, and we propose new data fusion extensions to address these problems.

Keywords: Data Fusion, Linked Data, RDF, Data Integration

1 Introduction

The value of Linked Data lies in the ability to link pieces of data. A data integration process applied to the data can provide a unified view on data and simplify the creation of Linked Data consuming applications. Nevertheless, conflicts emerge during the integration. Deduplication reveals different URIs representing the same real-world entities (identity conflicts), and conflicting values appear due to errors, missing, or outdated pieces of information (data conflicts).

Resolution of these conflicts is a task for data fusion. It *combines multiple records representing the same real-world object into a single, consistent and clean representation* [1]. In the context of Linked Data represented as RDF, real-world objects are represented as *resources*. A set of RDF triples describing a resource, a *resource description*, corresponds to a “record”. Conflicts are resolved, and low-quality values purged to get a clean representation of a resource. This is typically realized by *fusion functions* such as LATEST, VOTE, or AVERAGE. Tools implementing Linked Data fusion include, e.g., Sieve [2], or LD-FusionTool³ which we develop as part of the UnifiedViews ETL framework⁴ [3].

In this poster, we demonstrate how errors can be introduced in the fused data when there are dependencies between RDF properties, or when fusing the common pattern of structured values (e.g., address of an entity). We propose how to deal with these cases by extending the data fusion process with the notion of *dependent properties* and *dependent resources*.

³ <https://github.com/mifeet/LD-FusionTool>

⁴ successor of the ODCleanStore framework, where LD-FusionTool originated

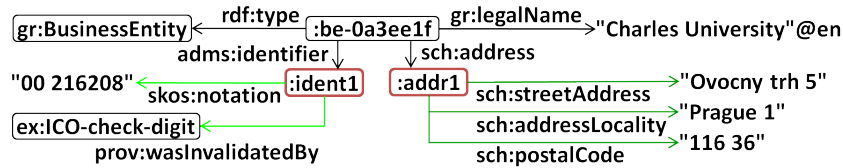


Fig. 1. Sample representation of a business entity. Red boxes denote dependent resources (structured values), green arrows denote groups of dependent properties.

2 Motivating Example

We demonstrate the need for new data fusion capabilities on a real scenario with public procurement data extracted from an XML-based API and RDFized using the UnifiedViews framework. The extracted data needs to be deduplicated and fused in order to obtain high-quality data before further analytical processing.

Fig. 1 shows how a business entity (BE) is represented in RDF. It has a legal name, address, and official identifier, which may be marked as syntactically invalid. The extracted data contains many copies of the same BE because of duplication in the source dataset. Simple merge of matched BEs would result in data conflicts due to misspellings and errors in the dataset or mismatches in the generated `owl:sameAs` links. Our goal is to fuse BEs so that each has a single legal name, address, and identifier, choosing the best possible values.

Property dependencies. We encounter the first problem with the state-of-the-art Linked Data fusion tools when fusing addresses. The tools resolve each property independently which can result in the selection of, e.g., a town from one address in the input and a postal code from another one. Such result could be incorrect, however, because the postal code is related to the town. We need to introduce dependency between properties to obtain a correct fused result.

Fusing structured values. Both address and identifier can be regarded as structured values of a BE. We will refer to the main resource (e.g., BE) as a *parent resource* and to the resource representing the structured value (e.g., address) as a *dependent resource*. Currently, structured values need to be fused separately. One way of achieving this is generating `owl:sameAs` links among structured values based on their properties, e.g., match addresses based on similarity of street, and town. This approach has two drawbacks: it doesn't guarantee that a BE will have only a single address after fusion, and the error of automatically generated `owl:sameAs` links accumulates. Another way is generating `owl:sameAs` links between dependent resources that belong to the same parent resource. This approach may lead to errors when two parent resources point to the same dependent resource, e.g., two different BEs point to the same address. All addresses for the two BEs would incorrectly be merged in such case.

We argue that a smarter approach considering structured values in resource descriptions could (1) overcome the outlined problems with the separate fusion of structured values, (2) reduce the overhead of additional linking, fusion, and validation, (3) gracefully handle blank nodes, where linking may not be practical.

3 Extending Linked Data Fusion

In this section, we propose how to extend data fusion to improve on the issues demonstrated in Section 2. Let there be a set U (RDF URI references), a set B (blank nodes) and a set L (literals). A triple $(s, p, o) \in (U \cup B) \times U \times (U \cup B \cup L)$ is an *RDF triple* and we refer to its components as subject, predicate, and object, respectively. Let $g \in U$ be a graph name. We regard a triple (s, p, o) that belongs to a *named graph* g as a *quad* (s, p, o, g) .

3.1 Property Dependencies

Independent fusion of properties is not always sufficient, as we demonstrated in Section 2. What we want is to keep the values of dependent properties together in the fused result if the values occurred together in the input data. Let us call a set of input quads sharing the same subject s and graph name g before resolution of identity conflicts an input group $IG_{s,g}$. Furthermore, let $d(p_1, p_2)$ denote that there is a dependency between properties p_1 and p_2 .

Definition 1. A fused result R from input quads I satisfies property dependencies if and only if $\forall p_1, p_2 \in U$ such that $d(p_1, p_2)$: all quads $(s, p, o, g) \in R$ such that $p = p_1 \vee p = p_2$ are derived⁵ from the same input group in I .

We chose to define input groups based on subject and graph because it covers two common scenarios: (1) fusing data from multiple sources (input quads can have different graph names), and (2) fusion after deduplication of a single source (quads will have different subjects before resolution of identity conflicts).

Here is how a basic data fusion algorithm can be extended to produce results satisfying property dependencies. The input of the algorithm includes these dependencies – we assume it is given as an equivalence relation d . We also assume the input resource description contains all quads for all mutually dependent properties. The extended algorithm consists of the following high level steps:

1. Find equivalence classes \mathcal{C} of the equivalence relation d .
2. For every class of dependent properties $C \in \mathcal{C}$:
 - (a) Let I_C be all input quads with one of the properties in C .
 - (b) For every nonempty input group $I_{s,g}$ in I_C , let $O_{s,g}$ be the fused result of the basic data fusion algorithm applied on $I_{s,g}$.
 - (c) Select one set O_C from all sets $O_{s,g}$ of fused quads according to some fusion tool specific criterion and add O_C to the result.
3. Fuse input quads with properties that do not have any dependency using the basic data fusion algorithm.

It is straightforward to prove that the extended algorithm indeed produces results satisfying property dependencies. The criterion used in step (2c) can depend on the implementing fusion tool. In LD-FusionTool, which can assess the quality of fused quads, we select $O_{s,g}$ such that the average quality of the fused result is maximal.

⁵ By *derived* we mean “selected from” for the so called *deciding* fusion functions such as LATEST, or “computed from” for *mediating* fusion functions such as AVERAGE.

3.2 Dependent Resources

Current Linked Data tools fuse resource descriptions composed of triples having the respective resource as its subject. Further triples describing structured values are not included (e.g., street is not included for a BE). This leaves a space for improvement as demonstrated in Section 2. We propose the inclusion of dependent resources reachable from the parent resource through specially annotated properties, in analogy to [4]. For resource r with resource description R , we fuse a property p annotated with fusion function `DEPENDENTRESOURCE` as follows:

1. Let $D_{r,p} = \{o \mid (r, p, o, g) \in R, o, g \in U\}$ be dependent resources. Recursively fuse resources in $D_{r,p}$ as if there were `owl:sameAs` links between all pairs of resources in $D_{r,p}$. Denote the fused result $F_{r,p}$.
2. Let $d \in U$ be a new unique URI. Add a new quad (r, p, d, g) , and quads $\{(d, q, o, g) \mid (s, q, o, g) \in F_{r,p}\}$ to the result.

This approach produces a single fused dependent resource (e.g., a single address of a BE), and takes advantage of the locality of `owl:sameAs` links to avoid incorrect merge of dependent resources with multiple parents. A unique URI is generated in step (2) so that other parts of the RDF graph where the dependent resource may occur are not affected by its “local” fusion for one parent resource.

4 Conclusion

Our practical experience with fusion of public procurement data shows that the graph nature of RDF has its specifics that need to be addressed. State-of-the-art Linked Data fusion tools do not cover two common patterns in RDF: fusion of structured values, and dependencies between properties.

We answer this challenge with new data fusion features. We introduce the concepts of dependent properties and dependent resources, and propose how to appropriately extend data fusion. The extensions have been implemented in `LD-FusionTool` and successfully used to fulfill the goals of our motivational scenario.

The new data fusion features show a new direction in Linked Data fusion – taking advantage of the broader context in the RDF graph. This can be further leveraged not only in conflict resolution, but also in quality assessment.

Acknowledgement. This work was supported by a grant from the EU’s 7th Framework Programme number 611358 provided for the project COMSODE.

References

- [1] Bleiholder, J., Naumann, F.: Data Fusion. In: ACM Computing Surveys 41.1 (2008)
- [2] Mendes, P. N., Mhleisen, H., Bizer, C.: Sieve: Linked Data Quality Assessment and Fusion. In: Proceedings of the 2012 Joint EDBT/ICDT Workshops, ACM (2012)
- [3] Knap, T., et al.: UnifiedViews: An ETL Framework for Sustainable RDF Data Processing. In: The Semantic Web: ESWC 2014, Posters and Demos Track (2014)
- [4] Mynarz, J., Svátek, V.: Towards a Benchmark for LOD-enhanced Knowledge Discovery from Structured Data. In: Proceedings of the Second International Workshop on Knowledge Discovery and Data Mining Meets Linked Open Data (2013).

Bio2RDF Release 3: A Larger Connected Network of Linked Data for the Life Sciences

Michel Dumontier¹, Alison Callahan¹, Jose Cruz-Toledo², Peter Ansell³, Vincent Emonet⁴, François Belleau⁴, Arnaud Droit⁴

¹Stanford Center for Biomedical Informatics Research, Stanford University, CA; ²IO Informatics, Berkeley, CA; ³Microsoft QUT eResearch Centre, Queensland University of Technology, Australia; ⁴Department of Molecular Medicine, CHUQ Research Center, Laval University, QC

```
{michel.dumontier, alison.callahan, josemiguelcruztoledo,  
peter.ansell, vincent.emonet, francois.belleau,  
arnaud.droit}@gmail.com
```

Abstract. Bio2RDF is an open source project to generate and provide Linked Data for the Life Sciences. Here, we report on a third coordinated release of ~11 billion triples across 30 biomedical databases and datasets, representing a 10 fold increase in the number of triples since Bio2RDF Release 2 (Jan 2013). New clinically relevant datasets have been added. New features in this release include improved data quality, typing of every URI, extended dataset statistics, tighter integration, and a refactored linked data platform. Bio2RDF data is available via REST services, SPARQL endpoints, and downloadable files.

Keywords: linked open data, semantic web, RDF

1 Introduction

Bio2RDF is an open-source project to transform the vast collections of heterogeneously formatted biomedical data into Linked Data [1], [2]. GitHub-housed PHP scripts convert data (e.g. flat files, tab-delimited files, XML, JSON) into RDF using downloadable files or APIs. Bio2RDF scripts follow a basic convention to specify the syntax of HTTP identifiers for i) source-identified data items, ii) script-generated data items, and iii) vocabulary used to describe the dataset contents [1]. Bio2RDF scripts uses the Life Science Registry (<http://tinyurl.com/lsregistry>), a comprehensive list of over 2200 biomedical databases, datasets and terminologies, to obtain a canonical dataset name (*prefix*), which is used in the formulation of a Bio2RDF URI - <http://bio2rdf.org/{prefix}:{identifier}> and identifiers.org URI. Each data item is annotated with provenance, including the URL of the files from which it was generated. Bio2RDF types and relations have been mapped to the Semanticscience Integrated Ontology (SIO)[3], thereby enabling queries to be formulated using a single terminol-

ogy [4]. Bio2RDF has been used for a wide variety of biomedical research including understanding HIV-based interactions [5] and drug discovery [6].

Here, we report an update to the Bio2RDF network, termed Bio2RDF Release 3, and compare the results to Bio2RDF Release 2.

2 Bio2RDF Release 3

Bio2RDF Release 3 (July 2014) is comprised of ~11B triples across 30 datasets, 10 of which are new since Release 2 (Table 1). The top 3 datasets are Pubmed (scholarly citations; 5B triples), iProClass (database cross-references; 3B triples), and NCBI Gene (sequences and annotations; 1B triples). Compared to Release 2, there are 10x the number of triples, and each dataset has increased by an average of 300%.

Table 1. Bio2RDF Release 3 Datasets.

Dataset	new	triples	% increase	types	out links	in links
Affymetrix		86,943,196	196%	2	30	0
Biomodels		2,380,009	404%	16	50	0
BioPortal		19,920,395	125%	-	-	-
Clinicaltrials	*	8,323,598	100%	55	1	1
CTD		326,720,894	230%	9	9	1
dbSNP	*	8,801,487	100%	6	5	2
DrugBank		3,649,561	325%	69	23	2
GenAge	*	73,048	100%	2	6	0
GenDR	*	11,663	100%	4	4	0
GO Annotations		97,520,151	122%	1	6	1
HGNC		3,628,205	434%	3	11	3
Homologene		7,189,769	561%	1	4	0
InterPro		2,323,345	233%	8	18	3
iProClass		3,306,107,223	1564%	0	16	0
iRefIndex		48,781,511	157%	5	24	0
KEGG	*	50,197,150	100%	17	43	7
LSR	*	55,914	100%	1	2	0
MeSH		7,323,864	176%	7	0	4
MGD		8,206,813	334%	11	8	4
NCBI Gene		1,164,672,432	296%	16	11	11
NDC		6,033,632	34%	12	0	2
OMIM		7,980,581	432%	8	17	8
OrphaNet	*	377,947	100%	3	12	2
PharmGKB		278,049,209	733%	18	41	1
PubMed		5,005,343,905	1350%	9	0	18
SABIO-RK		2,716,421	104%	15	11	1

now return RDF triples or quads based on content negotiation or RESTful URIs of the form `http://bio2rdf.org/[prefix]/[service]/[format]/[searchterm]`. The *describe* service returns statements with the searchterm as an identifier in the subject position. The *links* service returns triples with the searchterm as an identifier in the object position. Finally, the *search* service returns triples containing matched literals. Datasets and available services descriptions are stored and retrieved by the web application using a new SPARQL endpoint (`http://dataset.bio2rdf.org/sparql`).

4 Availability

Bio2RDF is accessible from `http://bio2rdf.org`. Bio2RDF scripts, mappings, and web application are available from GitHub (`https://github.com/bio2rdf`). A list of the datasets, detailed statistics, and downloadable content (RDF files, VoID description, statistics, virtuoso database) are available from `http://download.bio2rdf.org/current/release.html`. Descriptions of Bio2RDF datasets and file locations are also available from `datahub.io`.

5 References

- [1] A. Callahan, J. Cruz-Toledo, P. Ansell, and M. Dumontier, “Bio2RDF Release 2: Improved coverage, interoperability and provenance of life science linked data,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2013, vol. 7882 LNCS, pp. 200–212.
- [2] F. Belleau, M. A. Nolin, N. Tourigny, P. Rigault, and J. Morissette, “Bio2RDF: Towards a mashup to build bioinformatics knowledge systems,” *J. Biomed. Inform.*, vol. 41, no. 5, pp. 706–716, 2008.
- [3] M. Dumontier *et al*, “The SemanticScience Integrated Ontology (SIO) for biomedical research and knowledge discovery.,” *J. Biomed. Semantics*, vol. 5, p. 14, 2014.
- [4] A. Callahan, J. Cruz-Toledo, and M. Dumontier, “Ontology-Based Querying with Bio2RDF’s Linked Open Data.,” *J. Biomed. Semantics*, vol. 4 Suppl 1, p. S1, 2013.
- [5] M. A. Nolin, M. Dumontier, F. Belleau, and J. Corbeil, “Building an HIV data mashup using Bio2RDF,” *Brief. Bioinform.*, vol. 13, pp. 98–106, 2012.
- [6] B. Chen, X. Dong, D. Jiao, H. Wang, Q. Zhu, Y. Ding, and D. J. Wild, “Chem2Bio2RDF: a semantic framework for linking and data mining chemogenomic and systems chemical biology data.,” *BMC Bioinformatics*, vol. 11, p. 255, 2010.

Infoboxer: Using Statistical and Semantic Knowledge to Help Create Wikipedia Infoboxes

Roberto Yus¹, Varish Mulwad², Tim Finin², and Eduardo Mena¹

¹ University of Zaragoza, Zaragoza, Spain
{ryus, emena}@unizar.es,

² University of Maryland, Baltimore County, Baltimore, USA
{varish1, finin}@cs.umbc.edu

Abstract. Infoboxer uses statistical and semantic knowledge from linked data sources to ease the process of creating Wikipedia infoboxes. It creates dynamic and semantic templates by suggesting attributes common for similar articles and controlling the expected values semantically.

Keywords: Infoboxes, Wikipedia, DBpedia, Semantic Web

1 Introduction

Wikipedia is a free and collaborative encyclopedia launched in 2001 which, as of June 2014, has more than four million English articles. Wikipedia is centered around collaboratively creating and editing articles for a variety of topics and subjects. The information in these articles is often split into two parts: 1) unstructured text with details on the article's subject and 2) a semi-structured *infobox* that summarizes the most important facts about the article's subject. Thus, infoboxes are usually preferred by systems using Wikipedia content (such as Google's Knowledge Graph or Microsoft Bing's Satori) as they are easier to process by machines.

Current creation of Wikipedia infoboxes is based on templates that are created and maintained collaboratively. While templates provide a standardized way of representing infobox information across Wikipedia articles, they pose several challenges. Different communities use different infobox templates for the same category articles; attribute names differ (e.g., date of birth vs. birthdate), and attribute values are expressed using a wide variety of measurements and units [2]. Infobox templates are grouped by article categories with typically one template associated with one category (e.g., it is hard to find an infobox template for article whose categories are both Artist and Politician). Given the large number of Wikipedia categories, it is difficult to create templates for every possible category and combination. Finally, templates are free form in nature; when users fill attribute values no integrity check is performed on whether value is of appropriate type for the given attribute, often leading to erroneous infoboxes.

*Infoboxer*³ is a tool grounded in Semantic Web technologies that overcomes challenges in creating and updating infoboxes, along the way making the process easier for users. Using statistical information from Linked Open Data (LOD) datasets, Infoboxer helps people populate infoboxes using the most popular attributes used to describe instances for a given category or any combination of categories, thus generating an infobox “template” automatically. For each attribute or property Infoboxer also identifies the most popular types and provides them as suggestions to be used to represent attribute values. The attribute value types allows Infoboxer to enforce semantic constraints on the values entered by the user. It also provides suggestions for attribute values whenever possible and links them to existing entities in Wikipedia.

2 Using DBpedia to Help Creating Wikipedia Infoboxes

The Infoboxer demonstration presented in this paper, uses DBpedia [1], a semi-structured representation of Wikipedia’s content, to implement and power all of its features and functionalities. While our demonstration system uses DBpedia, it could be replaced with any other LOD knowledge base, such as Yago or Freebase. In the following sections we explain each functionality in detail.

Identifying popular attributes. The most popular attributes for a given category are generated by computing attribute usage statistics based on instance data for the category. Infoboxer first obtains a list of DBpedia instances for the given category. For example, list of instances associated with the category *dbpedia-owl:SoccerPlayer* include *dbpedia:David_Beckham* and *dbpedia:Tim_Howard*. A list of attributes used by these instances is generated and then ordered based on number of instances using each attribute. Duplicate counts are avoided by noting distinct attribute for every instance only once (at this point we want to know how many different instances of the category are using the property to highlight its popularity). For example, the property *dbpedia-owl:team* appears several times with the soccer player *dbpedia:David_Beckham* (as he played for several soccer teams), but it is only counted once.

Sorting the list of attributes based on frequency of usage provides Infoboxer with the most popular attributes for each category. Figure 1 shows the most popular properties for soccer players, e.g., *dbpedia-owl:team*, *foaf:name*, and *dbpedia-owl:position*, along with the percentage of instances using them. This first step could be simplified by only using information about the domains and ranges of each property (e.g., to obtain properties where the domain is a soccer player). However, DBpedia does not impose restrictions over domain and range for most of the properties. In fact, in a previous analysis, we detected that for DBpedia 3.9, 21% of properties have no domain defined, 15% have no range, and 2% have no domain and range. On July 1, Wikidata, a project focused on human

³ <http://sid.cps.unizar.es/Infoboxer>



Fig. 1. Screenshot of Infoboxer creating the Wikipedia infobox of a soccer player.

edited structured Wikipedia, rolled out a similar feature which is restricted to suggesting only popular properties⁴.

Identifying popular range types. Infoboxer finds the most popular types used to represent values for each attribute identified in the previous step. Attribute value types is akin to *rdfs:range* classes associated with an attribute or a property in an ontology. Infoboxer first obtains a list of attribute values for a given category and attribute by identifying list of triples in DBpedia’s ABox whose subject are instances of the given category and property, the given attribute. For example, the category *dbpedia-owl:SoccerPlayer* and attribute *dbpedia-owl:team* generates a list of values such as *dbpedia:Arsenal.F.C.* and *dbpedia:Korea.University*. A list of value types is generated from the values and ordered based on number of instances whose attribute values have the type. Based on the attribute, value types are either semantic classes, such as *dbpedia-owl:SoccerClub* and *dbpedia-owl:University*, or xml datatypes such as *xsd:string*, *xsd:integer*, or *xsd:datetime*. Sorting the list of types provides Infoboxer with the most popular attribute value (or range) types.

Suggesting attribute values and enforcing semantic constraints. The top three value types for an attribute are provided as suggestions to users as they add values for the most popular attributes in the infobox. Infoboxer also uses these types to enforce semantic constraints on the values entered, thus ensuring infobox correctness. In cases where value type is a semantic class, Infoboxer retrieves instances of that class and populates them for auto-completion as user starts filling up the value. In cases where value type is an XML datatype, Infoboxer

⁴ <http://lists.wikimedia.org/pipermail/wikidata-1/2014-July/004148.html>

shows the most popular values used as examples. Once the user enters a value, Infoboxer checks whether value conforms to the expected type.

Fixing existing infoboxes. Infoboxer also uses its functionalities to improve existing Wikipedia infoboxes. Given an article title, Infoboxer fetches its categories and existing attribute values. Then, it highlights popular properties with missing values and also highlights attribute values that have an incorrect semantic type. For example, as of June 2014, *dbpedia:David_Beckham* has the value *dbpedia:England_national_football_team* (whose *rdf:type* is *dbpedia-owl:SoccerClub*) for the attribute *dbpedia-owl:birthPlace* and Infoboxer highlights it as a possible error as only 2% of soccer players have a soccer club as birth place (49% of them have a *dbpedia-owl:Settlement* and 22% a *dbpedia-owl:City*). Also, Infoboxer encourages users to update the attribute value if it is of a less popular type (e.g., suggesting a value of type *dbpedia-owl:SoccerClub* over *dbpedia-owl:Organisation* for the property *dbpedia-owl:team*).

The combination of the four functionalities allows Infoboxer to dynamically generate infobox templates, ensure infobox correctness, and help assist in fixing existing ones. Since Infoboxer relies on KBs such as DBpedia, generated templates will automatically evolve with change in information in KBs over time.

3 Demonstration

The demo will allow users to create new infoboxes and edit existing ones. They begin by entering the name of a new or existing Wikipedia article and select appropriate categories for it (e.g., Soccer Player and Scientist). Users will be provided with the most popular attributes to be completed, along with its popularity, based on the selected categories. For each attribute, users will also be provided information about the top three value types; auto-complete will assist users in selecting the appropriate value. A “Google it” button will help user fire Google search queries to discover a possible value. Also, as users start filling values in the forms, current version of the infobox will be displayed on the side. In summary, users will be able to experience how fast and controlled it is to create semantically correct Wikipedia infoboxes with Infoboxer.

Acknowledgments. This research was supported by the CICYT project TIN-2010-21387-C02-02, DGA FSE, NSF awards 1228198, 1250627 and 0910838 and a gift from Microsoft Research.

References

1. Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., Hellmann, S.: DBpedia - a crystallization point for the web of data. *Web Semantics: Science, Services and Agents on the World Wide Web* 7(3), 154–165 (2009)
2. Morsey, M., Lehmann, J., Auer, S., Stadler, C., Hellmann, S.: DBpedia and the Live Extraction of Structured Data from Wikipedia. *Program: electronic library and information systems* 46, 157–181 (2012)

The Topics they are a-Changing — Characterising Topics with Time-Stamped Semantic Graphs

A. Elizabeth Cano,¹ Yulan He,² and Harith Alani¹

¹ Knowledge Media Institute, Open University, UK
ampaeli@gmail.com, h.alani@open.ac.uk

² School of Engineering and Applied Science, Aston University, UK
y.he@cantab.net

Abstract. DBpedia has become one of the major sources of structured knowledge extracted from Wikipedia. Such structures gradually re-shape the representation of Topics as new events relevant to such topics emerge. Such changes make evident the continuous evolution of topic representations and introduce new challenges to supervised topic classification tasks, since labelled data can rapidly become outdated. Here we analyse topic changes in DBpedia and propose the use of semantic features as a more stable representation of a topic. Our experiments show promising results in understanding how the relevance of features to a topic changes over time.

Keywords: social media, topic detection, DBpedia, concept drift, feature relevance decay

1 Introduction

Supervised topic classifiers which depend on labelled data can rapidly become outdated since new information regarding these topics emerge. This challenge becomes apparent when applying topic classifiers to streaming data like Twitter. The continuous change of vocabulary – in many cases event-dependent – makes the task of retraining such classifiers with fresh topic-label annotations a costly one. In event-dependent topics not only new lexical features re-characterise the topic but also existing features can potentially become irrelevant to the topic (e.g., Jan25 being relevant to violence in the Egyptian revolution is now less relevant to current representations of the topic violence). In dynamic environments the expectation that the progressive feature drifts of topics to be in the same feature space is not normally met.

The incorporation of new event-data to a topic representation leads to a linguistic evolution of a topic, but also to a change on its semantic structure. To the best of our knowledge, none of the existing approaches for topic classification using semantic features [4][2][5][7], has focused on the epoch-based transfer learning task. In this paper we aim to disseminate our work presented in [1] by summarising our proposed transfer learning approach for the epoch-based topic classification of tweets. In [1] we investigate whether the use of semantic features as opposed to lexical features can provide a more stable representation of a topic. Here we extend our work by representing cross-epoch settings gain in F-measure for both lexical and semantic feature with infographics. This enables us to highlight the relevance of the studied semantic features over the lexical ones.

1.1 Evolving Topics

DBpedia is periodically updated to incorporate any additions and modification in Wikipedia. This enables us to track how specific resources evolve over time, by comparing these resources over subsequent DBpedia editions. For example, changes to the semantic graph for the concept Barack_Obama can be derived from snapshots of this resource’s semantic graph from different DBpedia dumps³. E.g., in Figure 1, although some of the triples remain unchanged in consecutive dumps, new triples provide further information on the resource.

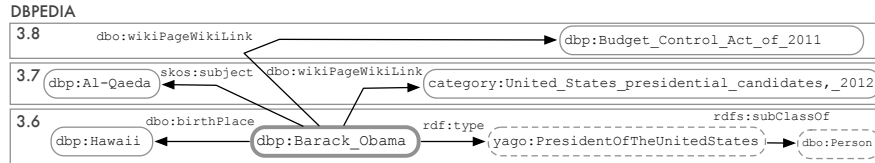


Fig. 1. Triples of the Barack_Obama resource extracted from different DBpedia dumps (3.6 to 3.8). Each DBpedia dump presents a snapshot in time of factual information of a resource.

Changes regarding a resource are exposed both through new semantic features (i.e. triples) and new lexical features –appearing on changes in a resource’s abstract–. In DBpedia a topic can be represented by the collection of resources belonging to both the main topic (e.g. `cat:War`) and resources (e.g. `dbp:Combat_assessment`) belonging to subcategories (e.g. `cat:Military_operations`) of the main Topic. Therefore a topic’s evolution can be easily tracked by tracking changes in existing and new resources belonging to it.

2 Topic Classification with Time-Stamped Semantic Graphs

In [1], we propose a novel transfer learning [6][3] approach to address the classification task of new data when the only available labelled data belongs to a previous epoch. This approach relies on the incorporation of knowledge from DBpedia graphs. This approach is summarised in Figure 2 and consists of the following stages: 1) Extraction of lexical and semantic features from tweets; 2) Time-dependent content modelling; 3) Strategy for weighting topic-relevant features with DBpedia; and 4) Construction of time-dependent topic classifiers based on lexical, semantic and joint features.

Our analysis involves the use of two main feature types: lexical and semantic features. The semantic features consist on Class, Property, Category, and Resource. The semantic feature representation of a document therefore is build upon the collection of such features derived from the document’s entities mapped to a DBpedia resource. The mapping targets the available DBpedia dump when the document was generated. In [1], we proposed different weighting strategies some of which made use of graph properties of a Topic in a DBpedia graph. Such strategies incorporated statistics of the topic graph representation considering a DBpedia graph at time t .

2.1 Construction of Time-Dependent Topic Classifiers

We focus on the binary topic classification in epoch-based scenarios, where the classifier that we train on a corpus from epoch $t - 1$, is tested on a corpus on epoch t . Our

³ The DBpedia dumps correspond to Wikipedia articles at different time periods as follows: DBp3.6 generated on 2010-10-11; DBpedia 3.7 on 2011-07-22, DBp3.8 on 2012-06-01, DBp3.9 on late April. DBpedia have them available to download at DBpedia <http://wiki.dbpedia.org/Downloads39>

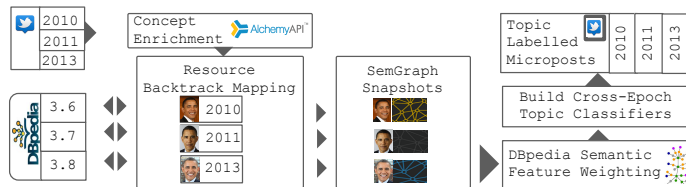


Fig. 2. Architecture for backtrack mapping of resources to DBpedia dumps and deriving topic-relevance based features for epoch-dependent topic classification.

analysis targeted our hypothesis that, as opposed to lexical features which are situation-dependent and can change progressively in time, semantic structures – including ontological classes and properties – can provide a more stable representation of a Topic.

Following the proposed weighting strategies the semantic feature representations of the $t - 1$ corpus and the t corpus, are both generated from the DBpedia graph available at $t - 1$. For example when applying a classifier trained on data from 2010, the feature space of a target test set from 2011 is computed based on the DBpedia version used for training the 2010-based classifier. This is in order to simulate the availability of resources in a DBpedia graph at a given time. The semantic feature f in a document x is weighted based on the frequency of a semantic feature f in a document x with Laplace smoothing and the topic-relevance of the feature in the \mathcal{DB}_t graph:

$$W_x(f)_{\mathcal{DB}_t} = \left[\frac{N_x(f)_{\mathcal{DB}_t} + 1}{|F| + \sum_{f' \in F} N_x(f')_{\mathcal{DB}_t}} \right] * (W_{\mathcal{DB}_t}(f))^{1/2} \quad (1)$$

where $N_x(f)$ is the number of times feature f appears in all the semantic meta-graphs associated with document x derived from the \mathcal{DB}_t graph ; F is the semantic features' vocabulary of the semantic feature type and $W_{\mathcal{DB}_t}(f)$ is the weighting function corresponding to the semantic feature type computed based on the \mathcal{DB}_t graph. This weighting function captures the relative importance of a document's semantic features against the rest of the corpus and incorporates the topic-relative importance of these features in the \mathcal{DB}_t graph.

3 Experiments

We evaluated our approach using two collections: DBpedia and Twitter datasets. The DBpedia collection comprises four DBpedia dumps (3.6 to 3.9)⁴. The Twitter datasets consist of a collection of Violence-related topics: Disaster_Accident, Law_Crime and War_Conflict. Each of these datasets comprises three epoch-based collections of tweets, corresponding to 2010, 2011, and 2013. The Twitter dataset contained 12,000 annotated tweets⁵. To compare the overall benefit of the use of the proposed weighting strategies against the baselines on this three topics, we averaged the P, R and F-measure of these three cross-epoch settings for each topic. Table 1 presents a summarised version of our results in [1], showing only the best performing features. We can see that in average the Class-based semantic features improve upon the bag of words (BoW) features in F measures. This reveals that the use of ontological classes is a more stable option for the representation of a topic. In order to analyse the differences in gain in F measure for each topic in each of the examined features we used the radar plots in Figure 3. In

⁴ General statistics of these dumps are available at <http://wiki.dbpedia.org/Downloads39>

⁵ Further information about this dataset is available at [1]

this figure a positive value indicates an improvement on the classifier. While semantic features improve upon lexical feature in the three topics, the weighted features for resource, class and category exhibit a positive improvement on these scenarios. Moreover the class based features consistently outperform the BoW in all three topics.

	BoW	Cat_{sff}	Cat_{sfg}	Cat_j	Res_{sff}	Res_{sfg}	Res_j	Cls_{sff}	Cls_{sfg}	Cls_j	Sem_{sff}	Sem_{sfg}	Sem_j
P	0.808	0.719	0.784	0.775	0.764	0.775	0.777	0.692	0.691	0.705	0.708	0.751	0.75
R	0.429	0.433	0.434	0.383	0.438	0.426	0.408	0.649	0.638	0.640	0.438	0.373	0.404
F	0.536	0.524	0.550	0.501	0.544	0.529	0.517	0.660	0.658	0.665	0.525	0.490	0.518

Table 1. Average results for the cross-epoch scenarios for all three topics.

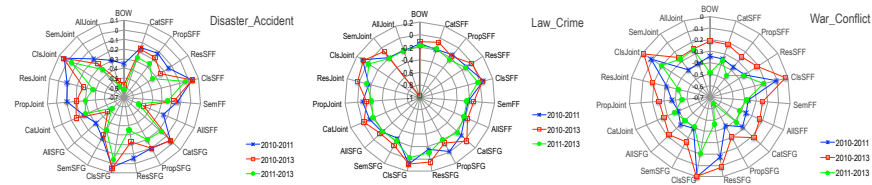


Fig. 3. Summary of performance decays for each feature for each Topic on the three cross-epoch scenarios.

4 Conclusions

Our results showed that Class-based semantic features are much slower to decay than other features, and that they can improve performance upon traditional BoW-based classifiers in cross-epoch scenarios. These results demonstrate the feasibility of the use of semantic features in epoch-based transfer learning tasks. This opens new possibilities for the research of concept drift tracking for transfer learning based on existing Linked Data sources.

References

1. A. E. Cano, Y. He, and H. Alani. Stretching the life of twitter classifiers with time-stamped semantic graphs. In *ISWC 2014, Riva del Garda, Trentino, Italy, Oct 19-23, 2014. Proceedings*, Lecture Notes in Computer Science. Springer, 2014.
2. A. E. Cano, A. Varga, M. Rowe, F. Ciravegna, and Y. He. Harnessing linked knowledge source for topic classification in social media. In *Proc. 24th ACM Conf. on Hypertext and Social Media (Hypertext)*, Paris, France, 2013.
3. R. Caruana. Multitask learning. 28(1):41–75, 1997.
4. Y. Genc, Y. Sakamoto, and J. V. Nickerson. Discovering context: classifying tweets through a semantic transform based on wikipedia. In *Proceedings of the 6th international conference on Foundations of augmented cognition: directing the future of adaptive systems*, FAC’11, pages 484–492, Berlin, Heidelberg, 2011. Springer-Verlag.
5. Y. He. Incorporating sentiment prior knowledge for weakly supervised sentiment analysis. *ACM Transactions on Asian Language Information Processing*, 11(2):4:1–4:19, June 2012.
6. S. Thrun. Is learning the n-th thing any easier than learning the first? In *Advances in Neural Information Processing Systems*, pages 640–646. The MIT Press, 1996.
7. A. Varga, A. Cano, M. Rowe, F. Ciravegna, and Y. He. Linked knowledge sources for topic classification of microposts: A semantic graph-based approach. *Journal of Web Semantics: Science, Services and Agents on the World Wide Web (JWS)*, 2014.

Linked Data and facets to explore text corpora in the Humanities: a case study

Christian Morbidoni

Semedia, Department of Information Engineering (DII), Università Politecnica delle Marche, Ancona, Italy

Abstract. Faceted search and browsing is an intuitive and powerful way of traversing a structured knowledge base and has been applied with success in many contexts. The GramsciSource project is currently investigating how faceted navigation and Linked Data can be combined to help Humanities scholars in working with digital text corpora. In this short paper we focus on the "Quaderni dal carcere" by Antonio Gramsci, one of the most popular Italian philosophers and politicians, we present our ongoing work and we discuss our approach. This consists of first building a RDF graph to encode different "levels" of knowledge about the texts and then extracting relevant graph paths to be used as navigation facets. We then built a first prototype exploration tool with a two-fold objective: a) allow non experts to make sense of the extremely fragmented and multidisciplinary text corpus, and b) allow Gramsci scholars to easily select a subset of the corpus of interest as well as possibly discovering new insights or answer research questions.

Keywords: Faceted browsing, Digital Humanities, Entity Extraction

1 The data

Gramsci's "Quaderni dal carcere" is an extremely fragmented corpus composed of more than 4,000 "notes" organized in 29 books (quaderni). Notes vary in length from single lines to several pages and span different domains, from sociology and politics to literature. They are available in the GramsciSource digital library¹ (created in the frame of the project) with stable URLs. We built a Linked Data graph by merging structured knowledge coming from different sources: the Linked Data Gramsci Dictionary, data coming from DBpedia Italia² and semantic annotations made with Pundit [1]³.

The **Linked Data Gramsci Dictionary**, is a dataset extracted from the Gramsci Dictionary [2], a recognized scholarly contribution within the international Gramsci scholarly community, which includes all the most important topics in the Gramsci's thought. Each topic in the dictionary is documented by a

¹ The DL is currently officially offline due to maintenance, but can be reached at the following address: <http://89.31.77.216>

² <http://it.dbpedia.org>

³ <http://thepund.it>

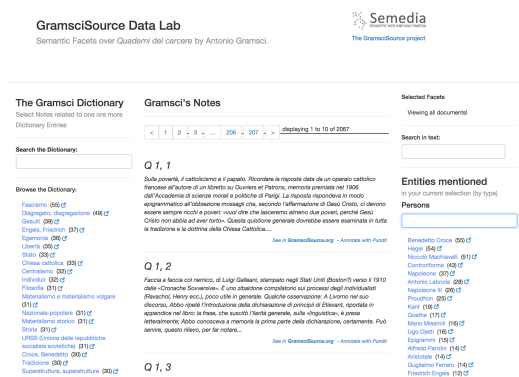


Fig. 1. A screenshot of the prototype

text with references to specific Gramsci's Notes from the Quaderni dal Carcere. We automatically processed such citations using regular expressions and producing RDF triples representing such connections, expressing relations among single notes and a number of dictionary topics they are relevant to.

Entity extraction and linking. Several approaches and tools for extracting and disambiguating relevant entities mentioned in a text appeared in the last years. Among them, DataTXT⁴ is, to our knowledge, one of the best tools supporting Italian language. DataTXT derives from previous academic research [3], makes use of Wikipedia to disambiguate matched entities and to link them to the Italian DBpedia and proved to be highly performant even on very short texts. Running such a entity extraction tool on all the notes resulted in over 2,038 notes (50% of the total number) annotated with at least one entity and a total of 43,000 entities matched. After a manual revision of the results we removed around 30 entities that were clearly wrong matches. We then inspected 80 random notes (2% of the total number of notes) and measured an accuracy of around 85%. Extracted entities span 144 different entity rdf:types and 5,876 distinct dc:types (which can be considered as entities categories). A more accurate evaluation of the results as well as a better tuning of the tool are goals for the next stage of the project.

Scholars annotations. Pundit⁵ is a semantic web tool that enables users to produce machine readable data in the form of RDF by annotating web pages. Annotations from a single scholar are collected in so called "notebooks", which can be private or public. For the purpose of our proof of concept we created a set of sample annotations by manually linking texts to DBpedia and Freebase entities. At data representation level, such annotations are equivalent to those produced by DataTXT, once imported they are naturally captured by the facet queries (discussed in the next section).

⁴ <https://dandelion.eu/products/datatxt/>

⁵ <http://thepund.it>

2 Faceted search prototype

Existing approaches to identify relevant facets to browse a RDF graph based on quantitative measures such as predicate frequency, balance and objects cardinality [4]. This kind of approaches do not account for informative content of a facet and only consider facets derived from a set of triples with the same predicate. In the general case, however, relevant facets could be derived from more complex paths in the graph. Approaches to automatic facets extraction in such a general case have been recently proposed [6] and we plan to investigate their applicability in the near future.

Our simple approach is to derive facets from SPARQL queries of the form:

```
select distinct ?url ?facet ?value where { CUSTOM_QUERY }
```

Where ?url is a resource of interest (notes in our case), ?facet is a facet name and ?value is a possible value of such a facet. Such a simple approach is also quite flexible and allows, for example, to easily turn all the datatype properties of a resource to facets, e.g with the following query:

```
select distinct ?uri ?facet ?value where {
?uri rdf:type gramsci:Note. ?uri ?facet ?facet.}
```

Deriving facets from SPARQL queries is an approach already explored in literature [5]. For the purpose of our proof of concept we chose candidate graph paths by inspecting the data and accordingly to scholars preferences. The facets we implemented in our prototype are:

- Gramsci Dictionary topics. This facet lists all the dictionary topics where a note is referenced;
- DBpedia entities. A set of facets where entities mentioned in a note are grouped according to their rdf:type. Relevant rdf:types individuated are Persons, Books, Languages, Places and Events, but they could be more specific (e.g. Politicians, Artists, Magazines, etc.);
- Categories. A facet listing all the dc:types associated to entities mentioned in a note;
- Scholars Notebooks. This facet lists all the scholars (Pundit users) who manually annotated a note.

To enable navigation of the corpus along the different "dimensions", we implemented a faceted browser based on Apache Solr⁶. Solr, along with its Ajax-Solr⁷ frontend provides a relatively easy way to build a performant faceted browser on top of Lucene. We built the solr index by running the SPARQL queries (described in the previous section) and using results associated to the ?uri variable as document ID, ?facet as index field and ?value as field values. The prototype is available at <http://purl.org/gramscisource/quaderni>.

⁶ <http://lucene.apache.org/solr/>

⁷ <http://github.com/evolvingweb/ajax-solr/wiki>

Some usage patterns have been individuated by scholars involved in the project: a) Using the Dictionary facet to intersect two or more topics from the vocabulary. This is a simple but useful "advanced search" feature; b) Choose one or more Dictionary topics (e.g. Storia), then use the facets on the right (DBpedia entities) to provide additional context (e.g. Hegel, Croce and Plechanov are the main persons related to History, "Teoria e storia della storiografia" and "Misre de la philosophie" are two related books, etc.); c) Start from a full text search or from a DBpedia entity (e.g. "Conte di Montecristo") and discover related topics.

3 Conclusions and Acknowledgements

In this short paper we discussed preliminary results in leveraging Linked Data in the GramsciSource project and we presented a proof of concept prototype. Feedback from Humanities scholars involved in the project (and in related projects, such as DM2E⁸) was positive and encouraged us to move further. End user evaluation will be run in the next months. We are currently evaluating automatic methods to derive entities and facets (e.g. based on language analysis tool such as [7]), with the aim of making the approach easily applicable to different texts corpora.

This work is supported by the GramsciSource project funded by the Italian Ministry of Education under the FIRB action.

References

1. Marco Grassi, Christian Morbidoni, Michele Nucci, Simone Fonda and Francesco Piazza. Pundit: Augmenting Web Contents with Semantics. *Literary & Linguistic Computing*, 2013
2. Dizionario gramsciano 1926-1937, Curated by Guido Liguori, Pasquale Voza, Roma, Carocci Editore, 2009, pp. 918.
3. Paolo Ferragina, Ugo Scaiella, TAGME: on-the-fly annotation of short text fragments (by wikipedia entities), *Proceedings of the 19th ACM international conference on Information and knowledge management*, New York, 2010
4. Eyal Oren, Renaud Delbru, Stefan Decker, Extending Faceted Navigation for RDF Data, *The Semantic Web - ISWC 2006, Lecture Notes in Computer Science Volume 4273*, 2006, pp 559-572.
5. Philipp Heim, Jrgen Ziegler, Faceted Visual Exploration of Semantic Data, *Human Aspects of Visualization, Lecture Notes in Computer Science Volume 6431*, 2011, pp 58-75.
6. Bei Xu, Hai Zhuge, Automatic Faceted Navigation, *Future Generation Computer Systems archive*, Volume 32, March, 2014, Pages 187-197
7. Dell'Orletta F., Venturi G., Cimino A., Montemagni S. (2014) T2K: a System for Automatically Extracting and Organizing Knowledge from Texts. In *Proceedings of 9th Edition of International Conference on Language Resources and Evaluation (LREC 2014)*, 26-31 May, Reykjavik, Iceland.

⁸ <http://dm2e.eu>

Dexter 2.0 - an Open Source Tool for Semantically Enriching Data

Salvatore Trani^{1,4}, Diego Ceccarelli^{1,2}, Claudio Lucchese¹,
Salvatore Orlando^{1,3}, and Raffaele Perego¹

¹ISTI-CNR, Pisa, Italy, ²IMT Lucca, Italy, ³Ca' Foscari - University of Venice,
⁴University of Pisa
{name.surname}@isti.cnr.it

Abstract. Entity Linking (EL) enables to automatically link unstructured data with entities in a Knowledge Base. Linking unstructured data (like news, blog posts, tweets) has several important applications: for example it allows to enrich the text with external useful contents or to improve the categorization and the retrieval of documents. In the latest years many effective approaches for performing EL have been proposed but only a few authors published the code to perform the task. In this work we describe Dexter 2.0, a major revision of our open source framework to experiment with different EL approaches. We designed Dexter in order to make it easy to deploy and to use. The new version provides several important features: the possibility to adopt different EL strategies at run-time and to annotate semi-structured documents, as well as a well-documented REST-API. In this demo we present the current state of the system, the improvements made, its architecture and the APIs provided.

1 Introduction

In the latest years many researchers proposed new techniques for performing *Entity Linking* (or *Wikification*) that consists of enriching a document with the entities that are mentioned within it. For example, consider the document in Figure 1: an EL framework first detects the pieces of text that are referring to an entity e.g., **Maradona**, **Argentina**, or **Belgium**, (usually called *mentions* or *spots*); this step is known as *mention detection* or *spotting*. Then the system performs the *disambiguation* step: each spot is linked to an entity chosen from a list of candidates. The entity is represented by its URI or identifier in a knowledge base, in our case Wikipedia. As an example, in Figure 1 the correct entity for the spot **Argentina** is http://en.wikipedia.org/wiki/Argentina_national_football_team. Please note that linking the mention to the correct entity is not a trivial task since often a mention is *ambiguous*: indeed in the previous example **Argentina** is not referring to the most common sense (the country) but rather to the national football team.

In this demo we present the current status of Dexter, our open source framework for entity linking. We introduced Dexter one year ago [1] in order to provide

```
Maradona, [http://en.wikipedia.org/wiki/Diego_Maradona] played his first World Cup
tournament [http://en.wikipedia.org/wiki/FIFA_World_Cup] in 1982 when Argentina
[http://en.wikipedia.org/wiki/Argentina_national_football_team] played Belgium
[http://en.wikipedia.org/wiki/Belgium_national_football_team] in the opening game of the 1982 Cup
[http://en.wikipedia.org/wiki/1982_FIFA_World_Cup] in Barcelona
[http://en.wikipedia.org/wiki/Barcelona].
```

Fig. 1: Example of annotated document

a tool for implementing new EL methods, and for comparing or simply exploiting the existing EL methods on a common platform.

We designed the framework for researchers and students; Dexter is easy to deploy: it consists of a unique jar file without external dependencies, and some binary files representing the model. The user only has to run the program that will expose a web server providing both a Rest API and a web interface for performing EL. The framework is highly modular and it allows the developers to replace single parts of the EL process. It runs on commodity hardware and it requires only 3 gigabytes of memory.

2 Dexter Framework

2.1 Architecture

Dexter¹ is developed in Java, and is organized in several Maven² modules (as depicted in Figure 2):

Json-wikipedia³ This module converts the Wikipedia XML Dump in a JSON Dump, where each line is a JSON record representing an article. The parser is based on the MediaWiki markup parser UKP⁴. While DBpedia only contains semistructured data extracted from the dump (mainly from the infoboxes) in RDF format, JSON-Wikipedia contains other fields, e.g., the section headers, the text (divided in paragraphs), the templates with their schema, text emphasized and so on. The module is designed to support different languages;

Dexter-Common Contains the domain objects, shared among all the modules of Dexter;

Dexter-Core The core implements the EL pipeline (illustrated on the right of Figure 2): the text is first processed by a *Spotter*, that produces a list of *spot matches*. Each spot match contains the offset of the match in the document, the list of entities that could be represented by the spot (produced by an *Entity Ranker*) and other features useful to perform the linking. The spot matches are then processed by a *Disambiguator* that for each spot tries to select the correct entity in the list of candidates (often relying on a *Relatedness* function, that estimates the semantic distance between two entities);

¹ The project page is <http://dexter.isti.cnr.it>, the website also presents a demo

² <http://maven.apache.org/>

³ json-wikipedia is available at <https://github.com/diegoceccarelli/json-wikipedia>

⁴ <http://www.ukp.tu-darmstadt.de/software/jwpl/>

Dexter-Webapp exposes a REST API for performing the annotations. It also implements a simple web interface for performing a demo. The current version of the REST API is briefly described in Table 1, and it is organized in 4 logical categories: the **Annotate API**, used for annotating a document, the **Spot API** that allows to retrieve the candidate spots in a document and to visualize their features, the **Graph API** and the **Category API** that allow to browse respectively the Wikipedia article’s link graph and the category graph. The current API is available and testable. We provide a well written documentation for each method, in a web page that also allows the user to test the service;

Dexter-Client a simple client to perform EL from a client machine, implicitly calling the REST API.

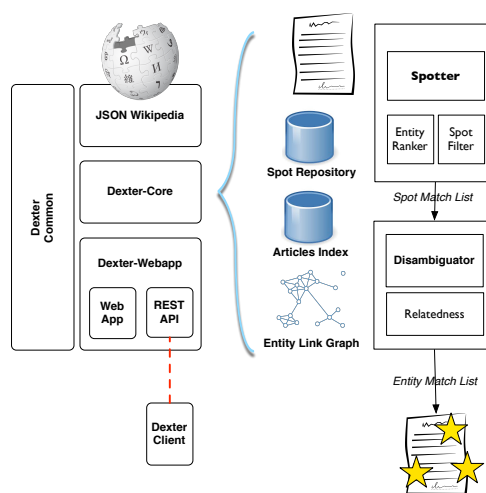


Fig. 2: Dexter Architecture

2.2 Novel Features

Since the first version we added the possibility to replace and combine different versions of the components of the system (the spotter, the disambiguator, the relatedness function *etc.*). An EL annotation can then be performed providing to the linker the symbolic names of the components that the developer wants to use (the spotter x , the disambiguator y ...). More in detail, in the `annotate` REST API the spotter and the disambiguator components are parameters, allowing to make use of different EL techniques at run-time. Another interesting feature is the possibility to annotate *semi-structured documents*; the previous version, as well as other EL frameworks, annotates only flat text, i.e., a plain string. In the new version we added the possibility to annotate documents composed by several fields (*e.g.*, title, headlines, paragraphs); when developing a new spotter/disambiguator, a researcher can exploit this information about the structure

of a document. It is worth to observe that in the new version each candidate spot contains also the field where the match was performed. The system also offers a Category API (extracted from the DBpedia categories).

Annotate API	
api/rest/annotate	Performs the EL on a given text
api/rest/get-desc	Given the Wikipedia Id of an entity, returns an object describing the entity (title, short description, ...)
Spot API	
api/rest/spot	Performs only the spotting step on the document, returning a list of mentions detected in the document, and for each mention some useful features and the list of possible candidate entities
api/rest/get-spots	Given an entity returns all the spots used in the Wikipedia dump for referring to the entity, for example given the entity Mona_lisa , it returns mona lisa,gioconda, la joconde ...
Graph API	
api/rest/get-target-entities	Returns the entities linked by the given entity
api/rest/get-source-entities	Returns the entities that link to the given entity
api/rest/relatedness	Returns the semantic relatedness between two entities (by default using the Milne and Witten formula [2])
Category API	
api/rest/get-parent-categories	Given an category, returns its parent categories
api/rest/get-child-categories	Given an category, returns its child categories
api/rest/get-entity-categories	Given an entity, returns its categories
api/rest/get-belonging-entities	Given a category, returns the entities belonging to the category

Table 1: The current version of the Dexter’s REST-API

Finally, we released a framework for evaluating the quality of the annotations and comparing our framework with the others⁵. We are also planning to integrate our tool with the NERD framework [3].

The demonstration will present the main functionalities provided by our system. We will illustrate how to use the API, how to deploy the system on a server, how to extend the components, and we will show some applications built on top of Dexter.

Future work. We are planning to add several disambiguators and spotters proposed in literature, and produce a performance comparison on different types of datasets.

Acknowledgements This work was partially supported by the EU project E-CLOUD (no. 325091), the Regional (Tuscany) project SECURE! (POR CReO FESR 2007/2011), and the Regional (Tuscany) project MAPaC (POR CReO FESR 2007/2013).

References

1. D. Ceccarelli, C. Lucchese, S. Orlando, R. Perego, and S. Trani. Dexter: an open source framework for entity linking. In *ESAIR*, 2013.
2. D. Milne and I. H. Witten. Learning to link with wikipedia. In *Proceedings of CIKM*, 2008.
3. G. Rizzo and R. Troncy. Nerd: a framework for unifying named entity recognition and disambiguation extraction tools. In *Proceedings of EACL*, 2012.

⁵ <https://github.com/diegoceccarelli/dexter-eval>

A Hybrid Approach to Learn Description Logic based Biomedical Ontology from Texts^{*}

Yue Ma¹ and Alifah Syamsiyah^{1,2}

¹Institute of Theoretical Computer Science, Technische Universität Dresden, Germany,

²Free University of Bozen-Bolzano

mayue@tu-dresden.de, alifah.syamsiyah@stud-inf.unibz.it

Abstract. Augmenting formal medical knowledge is neither manually nor automatically straightforward. However, this process can benefit from rich information in narrative texts, such as scientific publications. *Snomed-supervised relation extraction* has been proposed as an approach for mining knowledge from texts in an unsupervised way. It can catch not only superclass/subclass relations but also existential restrictions; hence produce more precise concept definitions. Based on this approach, the present work aims to develop a system that takes biomedical texts as input and outputs the corresponding $\mathcal{EL}++$ concept definitions. Several extra features are introduced in the system, such as generating general class inclusions (GCIs) and negative concept names. Moreover, the system allows users to trace textual causes for a generated definition, and also give feedback (i.e. correction of the definition) to the system to retrain its inner model, a mechanism for ameliorating the system via interaction with domain experts.

1 Introduction

Biomedicine is a discipline that involves a large number of terminologies, concepts, and complex definitions that need to be modeled in a comprehensive knowledge base to be shared and processed distributively and automatically. The National Library of Medicine (NLM) has maintained the world's largest biomedical library since 1836 [5]. One of the medical terminologies preserved by NLM is Systematized Nomenclature of Medicine Clinical Terms (SNOMED CT). It is a comprehensive clinical vocabulary structured in a well-defined form that has the lightweight Description Logic $\mathcal{EL}++$ [2] as the underlying logic, which can support automatic checking of modeling consistency.

However, creating, maintaining, and extending formal ontology is an expensive process [6]. In contrast, narrative texts, such as medical records, health news, and scientific publications, contain rich information that is useful to augment a medical knowledge base. In this paper, we propose a hybrid system that can generate \mathcal{EL} TBoxes from texts. It extends the formal definition candidates learned by the *Snomed-supervised relation extraction* process [4, 3] with linguistic patterns to give a finer-grained translation of the learned candidates. Besides generating concept name hierarchy that has been widely studied, the system can also generate definitions with existential restrictions to exploit the expressivity of \mathcal{EL} . Moreover, the implemented Graphical User Interface helps a user to visualize the flow of this framework, tracks textual sentences from which a formal definition is generated, and gives feedback to enhance the system interactively. The implementation of the system can be found from the link <https://github.com/alifahsyamsiyah/learningDL>.

^{*} We acknowledge financial support by the DFG Research Unit FOR 1513, project B1. Alifah Syamsiyah was supported by the European Master's Program in Computational Logic (EMCL)

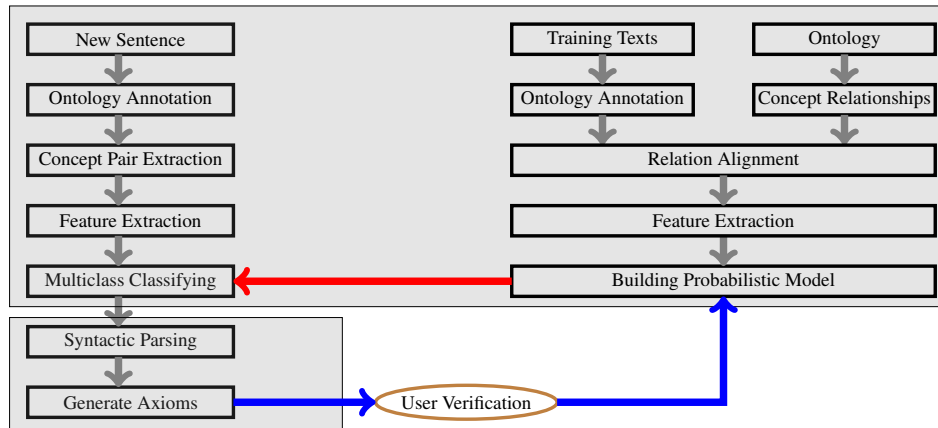


Fig. 1. Hybrid approach overview: the upper darked block is for machine learning phase to extract definition candidates, and the lower left darked block is for the pattern based transformation of definition candidates, and the brown ellipse is for interaction with users.

2 Task and Our Approach

Our task is to generate \mathcal{EL} definitions from textual sentences. For example, from the sentence “Baritosis is a pneumoconiosis caused by barium dust”, it is desired to have an automatic way to generate the formal \mathcal{EL} axiom (together with some confidence value), as shown in the red frame of Figure 2. Moreover, to help users understand the origin of a generated definition and/or give their feedbacks, the system should be able to trace the textual sources from which a definition is generated (implemented with the question mark in our system), and allow users to correct automatically learned definitions (the “V” mark in Figure 2).



Fig. 2. An illustrative example for the functionality of the system (CA is shortened form for the SNOMED CT relation Causative_Agent)

Below we describe our hybrid system that has two components, as shown in Figure 1: one for extracting definition candidates by machine learning techniques, and the other for formulating final definitions from the definition candidates by linguistic patterns.

2.1 Extracting Formal Definition Candidates

The first part of the system is to generate definition candidates via the steps given in the upper block of Figure 1. It again contains two components: learning a model from the training data (training texts and ontology) and generating candidates from new texts. Each steps in the two components are described below.

Common steps in processing training and test texts. One is to recognize SNOMED CT concept names from a given textual sentence, called **ontology annotation** in Figure 1. In our implementation, this is done by invoking the tool Metamap [1]. Since we are only interested in the most specific and precise concepts, we filter the Metamap annotations by keeping merely those that refer to head of a phrase but not a verb. The other common processing step for training and test sentence is to extract textual features for a pair of concepts occurring in a sentence, called **feature extraction**. Currently, the system uses classical lexical features of n-grams over both characters and words as in [4].

A special processing on training texts is concerned to generate labelled training data and to learn multi-class classification model for each predefined relation [4]:

- Automatic generation of training data is realized by the step named **Relationship Alignment** that matches an annotated sentence by Metamap with relationships between concept names from ontology: If one sentence containing a pair of concepts that has a relation R according to the ontology, this sentence is considered as a textual representation of R , thus being labelled with R . Furthermore, we also consider the inverse roles that often appear in texts via active and passive sentences. Hence, if there are n predefined relations, there will be $2n$ possible labels for a sentence.
- **Building probabilistic model** is to learn a probabilistic multi-class classification model based on the textual features of labelled sentences from the previous step. For this, the current system uses the maximum entropy Stanford Classifier¹.

A special processing on test texts is to extract definition candidates from a new test sentence. A definition candidate is a triple (A, R, B) where A, B are concept names and R is a relation, meaning that A and B have a relation R according to a test sentence.

- **Concept pair extraction** is to get pairs of concepts from an annotated test sentence.
- **Multiclass classification** is to answer whether a pair of two concept names has a relation, and if yes, which relation it is. This part can be achieved by the model learned from training data by Stanford Classifier. A positive answer returned by the classifier gives a definition candidate (A, R, B) . Slightly abusing of the notation, we also call $\exists r.B$ a definition candidate for A .

2.2 Pattern based Transformation of Definition Candidates

Once we get definition candidates, we first change the order of inverse role so it always appears as an active role. Next, different from [4], we distinguish two ways to formalize it: (1) into a subsumption $(A \sqsubseteq R.B)$ or (2) into a conjunction $(A \sqcap R.B)$. For example, the sentence “Baritosis is caused by barium dust” stands for the subsumption $Baritosis(disorder) \sqsubseteq Causative_agent.Barium_dust$; whilst “Chest pain from anxiety ...” corresponds to a conjunction $Chestpain(disorder) \sqcap \exists.Causative_agent.Anxiety(disorder)$. To decide which transformation of a definition candidate, we follow the intuition observable from the above examples:

- A subsumption $A \sqsubseteq \exists R.B$ should be generated from a candidate (A, R, B) if A and B are connected in the sentence in a subject-object relation, called *S-form*.
- A conjunction $A \sqcap \exists R.B$ should be formed if A and B appearing in a noun phrase structure, called *NP-form*.

¹ <http://nlp.stanford.edu/software/classifier.shtml>

To implement this linguistic pattern based strategy, we use the Stanford Parser² to get syntactical parsing tree of a test sentence. The S-form and NP-form are detected in the following way: First, the phrases corresponding to *A* and *B* are recognized from the sentences, and then the least common node of these two phrases is searched from the syntactic parsing tree of the whole sentence. If the least common node has type S (resp. NP)³, then *A* and *B* is in S-form (resp. NP-form). Otherwise, a parsing error is returned.

Negation Concept Names In natural language, sometimes we use negative way to define the opposite meaning. For example, the sentence “The disease from foot is not relative to heart attack” will be translated to $Disease(disorder) \sqcap FS. Foot(body\ structure) \sqsubseteq \neg Heart_disease(disorder)$. This is achieved in the system based on negated atomic concept names detectable by Metamap version 2013.

2.3 Tracing Source Sentence and Classifier Model Retraining

There are two extra functions provided by the system, namely tracing to sentence and classifier model retraining. As given in Figure 2, if a user clicks the “?” mark, system will provide sentences from which the formal definition extracted. Note that the system uses machine learning approach to acquire definition candidates which may get wrong. Therefore, we provide a mechanism for user to validate the answer by clicking “V” symbol and then give the correct relation to link two concept names. As shown in Figure 3, the user changes the role relation from inverse of Finding Site (FS-1) to Causative Agent (CA).

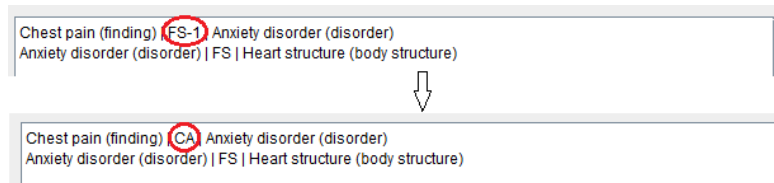


Fig. 3. Interaction with users: change the predicated relation in a definition candidate (FS is the shortened form for the SNOMED CT relation Finding_site, and FS-1 is the inverse role of FS)

References

1. Aronson, A.R., Lang, F.M.: An overview of metamap: historical perspective and recent advances. *JAMIA* **17**(3) (2010) 229–236
2. Baader, F., Brandt, S., Lutz, C.: Pushing the \mathcal{EL} envelope. In: Proceedings of IJCAI’05. (2005)
3. Ma, Y., Distel, F.: Concept adjustment for description logics. In: Proceedings of K-Cap’13. (2013) 65–72.
4. Ma, Y., Distel, F.: Learning formal definitions for Snomed CT from text. In: Proceedings of AIME’13. (2013) 73–77
5. National Library of Medicine: NLM overview. <http://www.nlm.nih.gov/about/index.html> (2014)
6. Simperl, E., Bürger, T., Hangl, S., Wögrl, S., Popov, I.: Ontocom: A reliable cost estimation method for ontology development projects. *Web Semantics: Science, Services and Agents on the World Wide Web* **16**(5) (2012)

² <http://nlp.stanford.edu/software/lex-parser.shtml>

³ “S” is for sentence, and “NP” for noun phase.

Identifying First Responder Communities Using Social Network Analysis

John S. Erickson, Katie Chastain, Evan W. Patton, Zachary Fry, Rui Yan,
James P. McCusker, Deborah L. McGuinness

Rensselaer Polytechnic Institute, Tetherless World Constellation
110 8th Street, Troy NY 12180 USA
{erickj4}@cs.rpi.edu

Abstract. First responder communities must identify technologies that are effective in performing duties ranging from law enforcement to emergency medical to fire fighting. We aimed to create tools that gather and assist in quickly understanding responders' requirements using semantic technologies and social network analysis. We describe the design and prototyping of a set of semantically-enabled interactive tools that provide a "dashboard" for visualizing and interacting with aggregated data to perform focused social network analysis and community identification.¹

Keywords: first responders, emergency response, network analysis, topic modeling

1 Introduction

In response to a request from NIST to develop approaches to using social networks and associated technology to improve first responder effectiveness and safety, we used semantic technologies and social network analysis to locate Twitter-based first responder sub-communities and to identify current topics and active stakeholders within those communities. Our objective is to create a repeatable set of Twitter-compatible methods that constitute an initial requirements gathering process. We report on using social media analysis techniques for the tasks of identifying first responder communities and on examining tools and techniques for identifying potential requirements stakeholders within those networks.

Our First Responders Social Network Analysis Workflow (Figure 1²) has helped researchers make sense of the vast quantity of information moving through Twitter. Identified stakeholders might be engaged by researchers in (for example) *participatory design*³ tasks that are elements of a requirements gathering

¹ A technical report discussing this work in greater detail may be found at [3]. All tool screenshots mentioned in this paper appear in the tech report in greater detail.

² See also <http://tw.rpi.edu/media/latest/workflow2>

³ Participatory design studies end user participation in the design and introduction of computer-based systems in the workplace, with the goal of creating a more balanced relationship between the technologies being developed and the human activities they are meant to facilitate. See e.g. [4], citing [5]

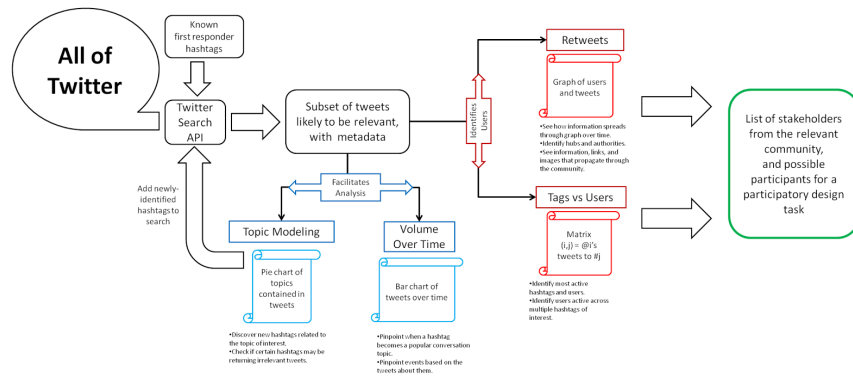


Fig. 1. Overview of a First Responders Social Network Analysis Workflow

methodology. We present first responder-related Twitter data and metadata through interfaces that reduce the overall information, to keep up with the quickly-changing environment of social media.

2 Identifying First Responder Communities During Disasters

We employed the Twitter Search API⁴ to collect tweets containing one or more hashtags from a list of 17 hashtags identified as relevant by the first responder community.⁵ We report on two events: the anticipated February 2013 Nemo storm and the unanticipated Boston Marathon bombing. A visualization tool allows browsing over time showing (for example) total tweets for a hashtag over time while enabling a user to zoom in and explore with finer temporal granularity.

3 Identifying Themes through Topic Modelling

We created a tool to visualize and enable interaction with topic modeling⁶ results, applying MALLET (<http://mallet.cs.umass.edu/>) across Twitter sample data. The tool presents topics as a pie chart; each "pie slice" represents an emergent topic, with assigned names indicating the most prevalent hashtags occurring in that topic. A popup list of hashtags enables the researcher to view other hashtags that are more loosely related to the topic.

⁴ See e.g. "Using the Twitter Search API" <http://bit.ly/1sY70>

⁵ For the complete list see <http://www.sm4em.org/active-hashtags/>

⁶ See especially [2]

4 Identifying Hashtags of Interest

Machine learning can help researchers identify hashtags that are topically related to the users area of interest but might not be immediately obvious. One of our tools uses co-occurrence to help identify evolving hashtags of interest, relating Twitter frequent posters with hashtags. The intensity of each cell in a matrix indicates the relative frequency with which a given user has tweeted using a particular hashtag. Users are filtered by *weighted entropy* and a subset is selected to provide the most coverage over hashtags of interest. Researchers may use this tool to develop a fine-grained understanding of topics and to pinpoint users of interest for further requirements gathering.

5 Multi-modal Visualization Tools

Situations may arise where close examination of network dynamics and conversation evolution is necessary. "Multi-modal" data visualizations enable researchers to move seamlessly from macro-scale visualizations to the micro-scale of individual tweets. Fig 2 shows one level where the propagation and retweeting of themes can be dynamically observed, while another level supports examination of individual tweets and associated media content.⁷

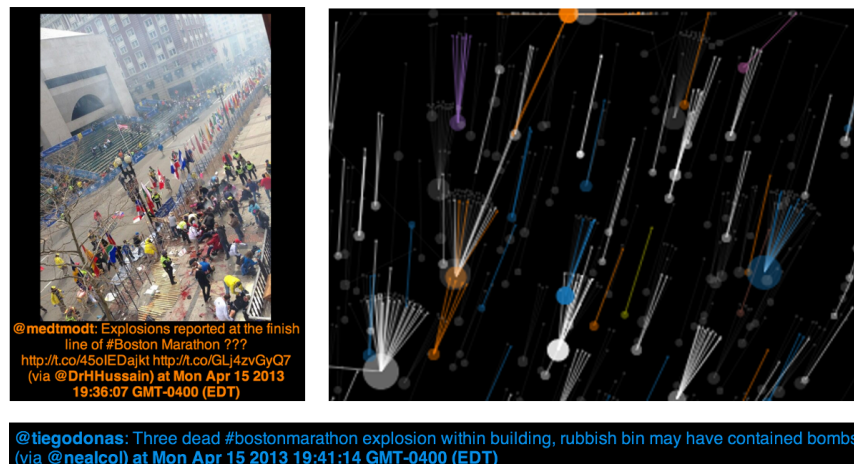


Fig. 2. Multi-modal visualization of Boston Marathon Twitter activity

⁷ Further details of the Twitter dataset used for this visualization may be found in [3]

6 Discussion and Conclusions

Our social network analysis and visualization tools demonstrate methods of passive social network monitoring⁸ intended to help researchers discover topical social network conversations among first responders. These tools have limited ability to connect and engage researchers with individual persons of interest. Current and future work includes extending the tools to expose and make actionable more user information, including identifying which individuals are most active on pertinent hashtags and are stakeholders of interest from a requirements gathering perspective. The time-sensitive nature of any Twitter sample dataset requires that visualization tools be adept at filtering over time periods of interest. Current and future work includes improved support for browsing over time with emphasis on finding and understanding topic shifts.

Recent events have demonstrated [1] that passive studies using social network data without full user knowledge and consent may backfire. Further studies should carefully examine the social implications of this work and in particular seek to understand at what point, if any, researchers should seek informed consent from potential stakeholder candidates.

7 Acknowledgements

We are grateful to the Law Enforcement Standards Office (OLES) of the U.S. National Institute of Standards and Technology (NIST) for sponsoring this work, and members of the DHS First Responders Communities of Practice Virtual Social Media Working Group (VSMWG) for numerous helpful discussions.

References

1. Arthur, C.: Facebook emotion study breached ethical guidelines, researchers say. *The Guardian* (June 2014), <http://bit.ly/1kuebVW>
2. Blei, D.M.: Probabilistic topic models. *Communications of the ACM* 55, 77–84 (April 2012), <http://bit.ly/1rd0ccT>
3. Erickson, J.S., Chastain, K., Patton, E., Fry, Z., Yan, R., McCusker, J., McGuinness, D.L.: Technical report: Identifying first responder communities using social network analysis. Tech. rep., RPI (July 2014), tw.rpi.edu/web/doc/tr_firstresponder_communityanalysis
4. Kensing, F., Blomberg, J.: Participatory design: Issues and concerns. *Computer Supported Cooperative Work* 7, 167185 (1998), <http://bit.ly/1zESj44>
5. Suchman, L.: Forward. In: Schuler, D., Namioka, A. (eds.) *Participatory Design: Principles and Practices*. p. viiix (1993)

⁸ *Passive monitoring* supports constant monitoring of a "default" set of known first responder hashtags meaning that when unanticipated events such as natural disasters happen, it is likely we'll have a useful if not perfect sample dataset. *Active monitoring*, conducted after the fact supports a deeper examination of user activity, including a focused examination of retweets and an investigation of "spontaneous" hashtags that emerge throughout the event.

Exploiting Semantic Annotations for Entity-based Information Retrieval

Lei Zhang¹, Michael Färber¹, Thanh Tran², and Achim Rettinger¹

¹ Institute AIFB, Karlsruhe Institute of Technology, Germany

² San Jose State University, USA

{l.zhang,michael.farber,rettinger}@kit.edu,
{ducthanh.tran}@sjsu.edu

Abstract. In this paper, we propose a new approach to entity-based information retrieval by exploiting semantic annotations of documents. With the increased availability of structured knowledge bases and semantic annotation techniques, we can capture documents and queries at their semantic level to avoid the high semantic ambiguity of terms and to bridge the language barrier between queries and documents. Based on various semantic interpretations, users can refine the queries to match their intents. By exploiting the semantics of entities and their relations in knowledge bases, we propose a novel ranking scheme to address the information needs of users.

1 Introduction

The ever-increasing amount of semantic data on the Web pose new challenges but at the same time open up new opportunities for information access. With the advancement of semantic annotation technologies, the semantic data can be employed to significantly enhance information access by increasing the depth of analysis of current systems, while traditional document search excels at the shallow information needs expressed by keyword queries and the meaningful semantic annotations contribute very little. There is an impending need to exploit the currently emerging knowledge bases (KBs), such as DBpedia and Freebase, as underlying semantic model and make use of semantic annotations that contain vital cues for matching the specific information needs of users.

There is a large body of work that automatically analyzes documents and the analysis results, such as part-of-speech tags, syntactic parses, word senses, named entity and relation information, are leveraged to improve the search performance. A study [1] investigates the impact of named entity and relation recognition on search performance. However, this kind of work is based on natural language processing (NLP) techniques to extract linguistic information from documents, where the rich semantic data on the Web has not been utilized. In [2], an ontology-based scheme for semi-automatic annotation of documents and a retrieval system is presented, where the ranking is based on an adaptation of the traditional vector space model taking into account adapted TF-IDF weights.

This work can be dedicated to research in this area. Nevertheless, it provides a significantly new search paradigm. The main contributions include: (1) The rich semantics in KBs are used to yield the semantic representations of documents and queries. Based on the various semantic interpretations of queries, users can refine them to match their intents. (2) Given our emphasize on semantics of entities and relations, we introduce a novel scoring mechanism to influence document ranking through manual selection of entities and weighting of relations by users. (3) Another important feature is the support of cross-linguality, which is crucial when queries and documents are in different languages.

2 Document Retrieval Process

In this section, we present our document retrieval process, which consists of five steps. While *lexica extraction* and *text annotation* are performed offline, *entity matching*, *query refinement* and *document ranking* are handled online based on the index generated by offline processing.

Lexica Extraction. In this step, we constructed the cross-lingual lexica by exploiting the multilingual Wikipedia to extract the cross-lingual groundings of entities in KBs, also called *surface forms*, i.e., words and phrases in different languages that can be used to refer to entities [3]. Besides the extracted surface forms, we also exploit statistics of the cross-lingual groundings to measure the association strength between the surface forms and the referent entities.

Text Annotation. The next step is performed to enrich documents with entities in KBs to help to bridge the ambiguity of natural language text and precise formal semantics captured by KBs as well as to transform documents in different languages into a language independent representation. For this purpose, we employ our cross-lingual semantic annotation system [4] and the resulting annotated documents are indexed to make them searchable with KB entities.

Entity Matching. Our online search process starts with the keyword query in a specific language. Instead of retrieving documents, our approach first finds entities from KBs matching the query based on the index constructed in the lexica extraction step. These entities represent different semantic interpretations of the query and thus are employed in the following steps to help users to refine the search and influence document ranking according to their intents.

Query Refinement. Different interpretations of the query are presented for users to select the intended ones. Since interpretations correspond to entities in this step, users can choose the intended entity for refinement of their information needs. We also enable users to adjust the weights of entity relations to influence the document ranking for a personalized document retrieval. For this, the chosen entity is shown and extended with relations to other entities retrieved from KBs.

Document Ranking. After query refinement by users, the documents in different languages containing the chosen entity are retrieved from the index constructed by text annotation. Then, we exploit the semantics of entities and relations for ranking. We observe that annotated documents generally share the following *structure pattern*: every document is linked to a set of entities, where

a subset (several subsets) of these entities are connected via relations in the KB, forming a graph (graphs). In this regard, a document can be conceived as a graph containing several connected components. Leveraging this pattern, we propose a novel ranking scheme based on the focus on the chosen entity and the relevance to the weighted relations.

Focus-Based Ranking: Intuitively, given two documents d_1 and d_2 retrieved for the chosen entity e , d_1 is more relevant than d_2 if it focuses more on e than d_2 does, i.e., when the largest connected component of d_1 containing e is larger than that of d_2 . Based on this rationale, we propose $\text{SCORE}_{Focus}(d, e)$ between document d and entity e to capture the focus of d on e as follows:

$$\text{SCORE}_{Focus}(d, e) = |LCC_d^e| \quad (1)$$

where LCC_d^e is the largest connected component of d containing e and $|LCC_d^e|$ represents the number of entities in LCC_d^e .

Relation-Based Ranking: Given the chosen entity e , the users can weight both the existence and the occurrence frequency of its relations to influence the document ranking. This differentiation separates the one scenario where users are interested in obtaining more detailed information about the relationship (qualitative information) from the other, where users are interested in the quantity. Let R_e be the set of relations of chosen entity e . We define $x_r = 1$ if $r \in R_e$, otherwise 0, and $y_r = \frac{|r_d|}{\log(avg_r)}$, where $|r_d|$ denotes the occurrence frequency of r in d and avg_r is the average occurrence frequency of r . Then, we propose $\text{SCORE}_{Relation}(d, e)$ between document d and entity e to capture the relevance of d to the weighted relations in R_e as follows:

$$\text{SCORE}_{Relation}(d, e) = \sum_{r \in R_e} x_r \cdot w_r^{existence} + y_r \cdot w_r^{frequency} \quad (2)$$

where $w_r^{existence}$ and $w_r^{frequency}$ are weights given by users for the existence and the occurrence frequency of relation r , respectively.

By taking into account both focus-based and relation-based ranking, we present the final function for scoring the documents as given in Eq 3.

$$\text{SCORE}(d, e) = \frac{\text{SCORE}_{Focus}(d, e) \cdot \text{SCORE}_{Relation}(d, e)}{ndl_d^e} \quad (3)$$

where ndl_d^e is the normalized document length of d w.r.t. annotations, i.e. the number of entities contained in d , which is used to penalize documents in accordance with their lengths because a document containing more entities has a higher likelihood to be retrieved. The effect of this component is similar to that of normalized document length w.r.t. terms in IR. We can compute it as

$$ndl_d^e = (1 - s) + s \cdot \frac{ef_d}{avg_{ef}} \quad (4)$$

where ef_d denotes the total number of entities in d , avg_{ef} is the average number of entities in the document collection, and s is a parameter taken from IR literature, which has been typically set to 0.2.

3 Evaluation

We now discuss our preliminary evaluation results. In the experiment, we use DBpedia [5] as the KB and Reuters Corpus Volume 1 (RCV1) as the document corpus containing about 810,000 English news articles. To assess the effectiveness of our approach, we investigate the normalized discounted cumulative gain (nDCG) measure of the top- k results instead of the common measures like precision and recall, which are not suitable to our scenario because the results can be different in relevance for each query and differ for each facet or weight used. We asked volunteers to provide keyword queries in Chinese (17 in total) along with descriptions of the intents used to set the weight for the relations, which yield the average nDCG of 0.87 and the average number of results of 612.

4 Conclusions and Future Work

In this paper, we show that the semantics captured in KBs can be exploited to allow the information needs to be specified and addressed on the semantic level, resulting in the semantic representations of documents and queries, which are language independent. The user feedback on our demo system [6] suggests that the proposed approach enables *more precise refinement* of the queries and is also valuable in terms of the *cross-linguality*. In the future, we plan to advance the query capability to support keyword queries involving several entities and conduct more comprehensive experiments to evaluate our system.

Acknowledgments. This work is supported by the European Community’s Seventh Framework Programme FP7-ICT-2011-7 (XLike, Grant 288342) and FP7-ICT-2013-10 (XLiMe, Grant 611346). It is also partially supported by the German Federal Ministry of Education and Research (BMBF) within the SyncTech project (Grant 02PJ1002) and the Software-Campus project “SUITE” (Grant 01IS12051).

References

1. Chu-Carroll, J., Prager, J.M.: An experimental study of the impact of information extraction accuracy on semantic search performance. In: CIKM. (2007) 505–514
2. Castells, P., Fernández, M., Vallet, D.: An adaptation of the vector-space model for ontology-based information retrieval. IEEE Trans. Knowl. Data Eng. **19**(2) (2007) 261–272
3. Zhang, L., Färber, M., Rettinger, A.: xlid-lexica: Cross-lingual linked data lexica. In: LREC. (2014) 2101–2105
4. Zhang, L., Rettinger, A.: X-lisa: Cross-lingual semantic annotation. PVLDB **7**(13) (2014) 1693–1696
5. Bizer, C., Lehmann, J., Kobilarov, G., Auer, S., Becker, C., Cyganiak, R., Hellmann, S.: DBpedia - A crystallization point for the Web of Data. J. Web Sem. **7**(3) (2009) 154–165
6. Färber, M., Zhang, L., Rettinger, A.: Kuphi - an investigation tool for searching for and via semantic relations. In: ESWC. (2014)

Crawl Me Maybe: Iterative Linked Dataset Preservation

Besnik Fetahu, Ujwal Gadiraju, and Stefan Dietze

L3S Research Center, Leibniz Universität Hannover, Germany
{fetahu, gadiraju,dietze}@L3S.de

Abstract. The abundance of Linked Data being published, updated, and interlinked calls for strategies to preserve datasets in a scalable way. In this paper, we propose a system that iteratively crawls and captures the evolution of linked datasets based on flexible crawl definitions. The captured deltas of datasets are decomposed into two conceptual sets: evolution of (i) *metadata* and (ii) the actual *data* covering schema and instance-level statements. The changes are represented as logs which determine three main operations: *insertions*, *updates* and *deletions*. Crawled data is stored in a relational database, for efficiency purposes, while exposing the *diffs* of a dataset and its live version in RDF format.

Keywords: Linked Data; Dataset; Crawling; Evolution; Analysis

1 Introduction

Over the last decade there has been a large drive towards publishing structured data on the Web. A prominent case being data published in accordance with Linked Data principles [1]. Next to the advantages concomitant with the distributed and linked nature of such datasets, challenges emerge with respect to managing the evolution of datasets through adequate preservation strategies. Due to the inherent nature of linkage in the LOD cloud, changes with respect to one part of the LOD graph, influence and propagate changes throughout the graph. Hence, capturing the evolution of entire datasets or specific subgraphs is a fundamental prerequisite, to reflect the temporal nature of data and links. However, given the scale of existing LOD, scalable and efficient means to compute and archive *diffs* of datasets are required.

A significant effort towards this problem has been presented by Käfer et al.[2], with the Dynamic Linked Data Observatory: a long-term experiment to monitor a two-hop neighbourhood of a core set of diverse linked data documents.

The authors investigate the lifespan of the core set of documents, measuring their on and off-line time, and the frequency of changes. Furthermore, they delve into how the evolution of links between dereferenceable documents over time. An understanding of how links evolve over time is essential for traversing linked data documents, in terms of reachability and discoverability. In contrast to the previous initiatives, in this work we provide an iterative linked dataset crawler.

computed at different levels. Each crawl explicitly logs the various changes at schema and resource-levels in a dataset as either *inserted*, *updated* or *deleted*. The changes themselves are first captured at triple-level, and then attributed to either schema-level or resource instance-level. The following log operators with respect to dataset evolution are handled by the dataset crawler.

- **Insertions.** New triples may be added to a dataset. Such additions introduced in the dataset correspond to insertions.
- **Deletions.** Over time, triples may be deleted from a dataset due to various reasons ranging from persisting correctness to detection of errors. These correspond to deletions.
- **Updates.** Updates correspond to the update of one element of a triple $\langle s, p, \rangle$.

Figure 2 presents an example depicting the computation of Δ between a previously crawled dataset at crawl-point t_0 and a fresh crawl at crawl-point t_1 .

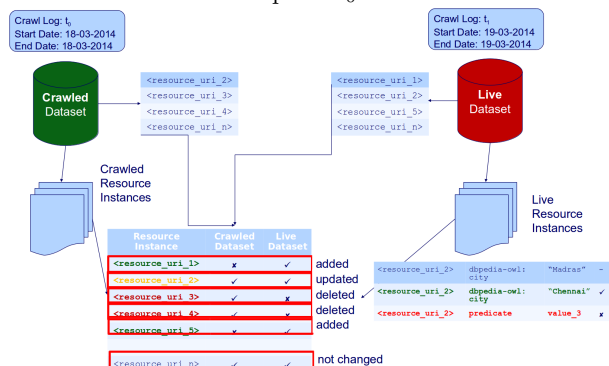


Fig. 2. Computation of *diffs* on-the-fly.

First, assume a change in the ‘live dataset’ in the form of an insertion of the triple corresponding to the URI `resource_uri_2`. Thus, the triple describing the city `Madras` is added. Consequently, if the value of the property `dbpedia-owl:city` is updated, then a subsequent crawl would capture this difference in the literal value of the property as an update to `Chennai`. Similarly, deletions made are also detected during the computation of *diffs*. Thus, computing and storing *diffs* on-the-fly in accordance with the log operators is beneficial; we avoid the overheads emerging from storing dumps of entire datasets.

2.2 Web Interface for the Iterative Dataset Crawler

We present a Web interface (accessible at http://data-observatory.org/dataset_crawler) that provides means to access the crawled resources, given specific crawl-points of interest from the periodical crawls. The interface allows us to filter for specific datasets and resource types. The Web application has three main components (see Figure 3): (i) displaying metadata of the dataset, (ii) dataset evolution, showing summaries of added/updated/deleted resources for

the different types, and (iii) dataset type-specific evolution, showing a summary of the added/updated/deleted resource instances for a specific resource type and corresponding to specific crawl time-points. In addition, the crawler tool is made available along with instructions for installation and configuration².

The screenshot displays the Dataset Crawler Web Interface with three main sections:

- View Metadata:** Contains a 'Select Dataset' dropdown menu set to 'All Datasets', a 'Dataset-id' dropdown menu set to 'data-incubator-our-airports', and a blue 'View Metadata' button.
- Dataset Evolution:** Contains a 'Dataset-id' dropdown menu set to 'clean-energy-data-reegle', 'Start Time' and 'End Time' dropdown menus both set to '2014-06-10 20:05:06', and a blue 'View' button.
- View Resource Type Evolution:** Contains a 'Dataset-id' dropdown menu set to 'geonames-semantic-web', a 'Resource-Type' dropdown menu set to '- SELECT -', 'Start Time' and 'End Time' dropdown menus both set to '2014-06-10 20:05:06', and a blue 'View' button.

Fig. 3. Functionalities of the Dataset Crawler Web Interface.

3 Conclusion

In this paper, we presented a linked dataset crawler for capturing dataset evolution. Data is preserved in the form of three logging operators (insertions/updates/deletions) by performing an online Δ computation for any given dataset with respect to the live state of the dataset and its previously crawled state (if available). Furthermore, the crawled and computed Δ of a dataset can be used to assess its state at any given crawl-point. Finally, we provided a web interface which allows the setup of the crawler, and facilitates simple query functionalities over the crawled data.

References

1. C. Bizer, T. Heath, and T. Berners-Lee. Linked data - the story so far. *Int. J. Semantic Web Inf. Syst.*, 5(3):1–22, 2009.
2. T. Käfer, A. Abdelrahman, J. Umbrich, P. OByrne, and A. Hogan. Observing linked data dynamics. In *The Semantic Web: Semantics and Big Data*, pages 213–227. Springer, 2013.

² https://github.com/bfetahu/dataset_crawler

A Semantics-Oriented Storage Model for Big Heterogeneous RDF Data

HyeongSik Kim, Padmashree Ravindra, and Kemafor Anyanwu

Department of Computer Science, North Carolina State University, Raleigh, NC
{hkim22, pravind2, kogan}@ncsu.edu

Abstract. Increasing availability of RDF data covering different domains is enabling ad-hoc integration of different kinds of data to suit varying needs. This usually results in large collections of data such as the Billion Triple Challenge datasets or SNOMED CT, that are not just “big” in the sense of volume but also “big” in variety of property and class types. However, techniques used by most RDF data processing systems fail to scale adequately in these scenarios. One major reason is that the storage models adopted by most of these systems, e.g., vertical partitioning, do not align well with the semantic units in the data and queries. While Big Data distributed processing platforms such as the Hadoop-based platforms offer the promise of “unlimited scale-out processing”, there are still open questions as to how best to physically partition and distribute RDF data for optimized distributed processing. In this poster, we present the idea of a *semantics-oriented RDF storage model* that partitions data into logical units that map to subqueries in graph patterns. These logical units can be seen as *equivalence classes of star subgraphs* in an RDF graph. This logical partitioning strategy enables more aggressive pruning of irrelevant query results by pruning irrelevant partitions. It also enables the possibility of semantic-query optimization for some queries such as eliminating joins under appropriate conditions. These benefits in addition to appropriate techniques for physically partitioning the logical partitions, translate to improved performance as shown by some preliminary results.

Keywords: RDF Storage Model, Partitioning Scheme, Hadoop, MapReduce

1 Introduction and Motivation

The Resource Description Framework (RDF) has been widely adopted and used to represent various datasets in many different communities such as government, life science, and finance, etc. One challenge that arises from this phenomenon is that most RDF datasets now contain a significant number of various properties and classes, e.g., 10^5 distinct properties and classes in DBPedia [3] and 400k concepts in SNOMED CT [7]. This is in contrast to popular benchmark datasets that are often used for evaluating RDF data processing systems like LUBM¹, which contain only a few hundreds of distinct properties and classes. To efficiently process such collections, data needs to be organized suitably into a storage model. A very common storage model is called *Vertical Partitioning* [1](VP) and its variants [4] which partition data in terms of the

¹ The Lehigh University Benchmark: <http://swat.cse.lehigh.edu/projects/lubm>

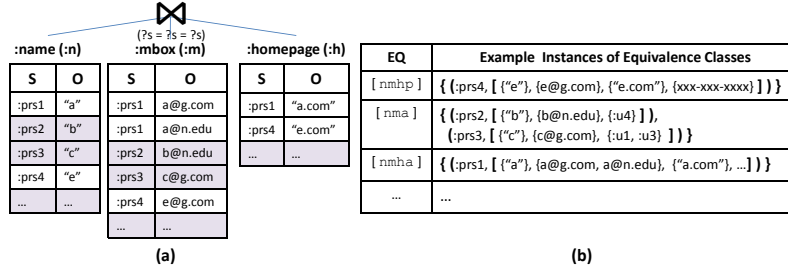


Fig. 1: A comparison of partitioning schemes: (a) vertical partitioning and its execution plan and (b) equivalence-class-based partitioning.

types of properties and classes in a dataset. Given a query, matching vertical partitions are selected based on the properties in the graph pattern and then join operations are performed using the partitions. In a sense, this approach allows all vertical partitions corresponding to properties that are not in the query to be pruned out. However, despite this degree of prunability, the joins between “relevant” vertical partitions still incurs some overhead of processing irrelevant data since not all entries the vertical partitions form joined results. For example, consider the star pattern query with properties `:name`, `:mbox`, and `:homepage`. Fig. 1(a) shows the example of the execution plan and partitioned data using the VP-based approach, which results in two join operations. The violet-colored cells denote triples that are not relevant to query, but are processed and discarded during expensive join operations. Furthermore, the vertical partitioning process itself can be challenging for large heterogeneous datasets for multiple reasons. First, it may require the management of a large number file descriptors/buffers ($> 10^5$ for DBPedia) in memory during the partitioning process which can be impractical depending on hardware architecture being used. Second, a scalability is a key design objective on Hadoop-based frameworks, but the distributed file system used in Hadoop (or HDFS) does not scale well when there are numerous small files². Given these challenges, there is a clear need for investigating novel storage and distribution schemes for RDF on scale-out platforms such as Hadoop.

2 Semantics-Oriented Storage Model : SemStorm

In this poster, we build on our previous works (e.g., [5, 6]) which introduced the notion of a *triple group* as a first class object in our data and query model. A *triple group* is a group of triples related to the same resource (i.e. with the same subject), i.e. a star subgraph. Fig. 1(b) shows the example triple group representation of our previous example, e.g., under the equivalence class `[nmha]`, a single triple group instance exists, which contains a subject (`:prs1`) and objects corresponding to properties `:n`, `:m`, `:h`, and `:a`. The benefits of both the triple group data model and algebra have been articulated in our previous works, including shortening of query execution workflows, reducing the footprint of intermediate results which impacts I/Os in distributed processing. Here, we present an overview of an RDF storage model called *SemStorm* that is based on logically and

² <http://blog.cloudera.com/blog/2009/02/the-small-files-problem>

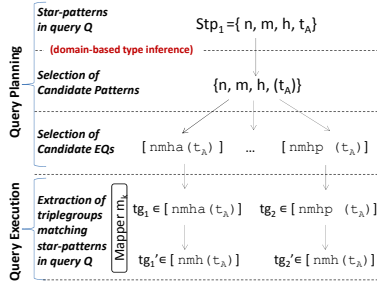
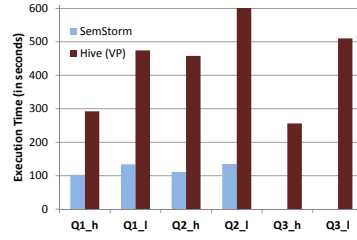


Fig. 2: Query processing in SemStorm.

Fig. 3: An execution time of queries with type triples ($Q1, Q2$) and a negative query ($Q3$).

physically partitioning triplegroups in a semantics-oriented way. By semantics-oriented we mean, partitioning triplegroups into equivalence classes such that all members in an equivalence class are equivalent with respect to queries. This approach enables more aggressive pruning of logical partitions, i.e. equivalence classes than other approaches like the vertical partitioning. For example, Fig. 2 shows that we select matching equivalence class sets (*mecs*): $[nmha]$ and $[nmhp]$, which contain all the properties in the example query, i.e. $:n, :m$, and $:h$ (ignore a type property for now such as t_A). All other remaining equivalence classes are pruned out, e.g., $[nma]$ is not selected due to the absence of $:h$. The *mecs* sometimes contain extra values, e.g., objects for property $:a$ and $:p$ in $[nmha]$ and $[nmhp]$. We later filter such values for exact matching results.

Another unique advantage of SemStorm is that it enables additional optimizations that are not possible with other approaches, e.g., it may be possible to avoid explicitly materializing *rdf:type* triples if such triples can be inferred by the label of an equivalence class (the label of an equivalence class can be considered to be the set of properties in that equivalence class). For example, triplegroups under an equivalence class $[pubAuthor, rdf:type \text{ with } Publication]$ can skip materializations of *Publication* type triples if a schema file contains a triple “*pubAuthor rdfs:domain Publication*”. Fig. 2 shows that type triples are not materialized in triplegroup instances such as tg_1 and tg_2 , which are denoted as (t_A) for the class A . This optimization can add significant advantages because *rdf:type* triples tend to be disproportionately larger than other properties for many datasets, e.g., approx. 20% in LUBM datasets. Thus, avoiding their explicit representation reduces the amount of I/Os needed when *rdf:type* triples need to be processed and may in some cases eliminate the need to perform a join with such properties since it is implicitly captured in the equivalence class representation.

Implementation Issues. Triplegroups can be generated easily using a group-by operation on a subject field of triples using a single MR job; they are then categorized based equivalence class and stored in HDFS. Each equivalence class could be mapped into a physical file, but it is likely that such 1:1 mappings could cause the many file issue in case that many distinct equivalence classes are generated. To relieve the issue, we need a heuristic that clusters equivalence classes into a smaller number of files, e.g., group equivalence classes that share a specific set of properties and store them together. We also need to consider building indexes to locate matching equivalence classes from physical files, e.g., mappings between equivalence class and their offsets in files.

Preliminary Evaluation. We evaluated three types of queries using the LUBM datasets (450GB, Univ. 20k) on a 80-node Hadoop cluster in VCL³, where each node was equipped with 2.33 GHz dual core CPUs, 4GB RAM, and 40GB HDD. Hive 0.12⁴ was selected for the VP approach. Query *Q1* retrieves a list of publication authors with *Publication* type triples, and *Q2* additionally retrieves name (or title) of the publications. We evaluated two variations of queries, with object field of some non-type triple pattern bounded (high selectivity, denoted with postfix *h*), and same query with unbounded object (low selectivity marked with *l*). Fig. 3 shows that SemStorm was 3 times faster than Hive for *Q1* because SemStorm can process queries using a Map-only job and save the disk I/O for all the type triples. The execution time of Hive increased from *Q1* to *Q2* due to reading additional property relation *:name* but the execution time of SemStorm was almost constant because both queries read the same equivalence classes. Finally, *Q3* was a negative query, which produces no answers. While SemStorm determined that there are no answers (due to no matching equivalence classes) even before launching the job, Hive executed join operations, producing 0 answer. The details are available in the project website.⁵

Related Work. Our approach might be similar with Property Table [2], which groups triples that tend to be together. However, the main difference is that the property table is query-driven, which is built gradually based on query logs. However, SemStorm is data-driven one, which directly can be constructed from the datasets without any query logs. In addition, while the property table approach mainly suffers from its storage inefficiencies, e.g., a lot of NULLs and left-over tables, our approach does not, i.e. all triples can be transformed into triplegroups without any leftovers.

Acknowledgment The work presented in this paper is partially funded by NSF grant IIS-1218277.

References

1. Abadi, D.J., Marcus, A., Madden, S.R., Hollenbach, K.: Scalable Semantic Web Data Management Using Vertical Partitioning. In: Proc. VLDB. pp. 411–422 (2007)
2. Carroll, J.J., Dickinson, I., Dollin, C., Reynolds, D., Seaborne, A., Wilkinson, K.: Jena: Implementing the Semantic Web Recommendations. In: Proc. WWW Alt. pp. 74–83 (2004)
3. Duan, S., Kementsietsidis, A., Srinivas, K., Udre, O.: Apples and Oranges: A Comparison of RDF Benchmarks and Real RDF Datasets. In: Proc. SIGMOD. pp. 145–156 (2011)
4. Husain, M., McGlothlin, J., Masud, M., Khan, L., Thuraisingham, B.: Heuristics-Based Query Processing for Large RDF Graphs Using Cloud Computing 23(9), 1312–1327 (2011)
5. Kim, H., Ravindra, P., Anyanwu, K.: Scan-Sharing for Optimizing RDF Graph Pattern Matching on MapReduce. In: Proc. CLOUD. pp. 139–146 (2012)
6. Ravindra, P., Kim, H., Anyanwu, K.: An Intermediate Algebra for Optimizing RDF Graph Pattern Matching on MapReduce. In: Proc. ESWC. pp. 46–61 (2011)
7. Salvadores, M., Horridge, M., Alexander, P.R., Ferguson, R.W., Musen, M.A., Noy, N.F.: Using SPARQL to Query Biportal Ontologies and Metadata. In: Proc. ISWC. pp. 180–195 (2012)

³ Virtual Computing Lab: <http://vcl.ncsu.edu>

⁴ <https://hive.apache.org>

⁵ <http://research.csc.ncsu.edu/coul/RAPID+/ISWC2014.html>

Approximating Inference-enabled Federated SPARQL Queries on Multiple Endpoints

Yuji Yamagata and Naoki Fukuta

Graduate School of Informatics, Shizuoka University
Shizuoka, Japan
{gs14044@s, fukuta@cs}.inf.shizuoka.ac.jp

Abstract. Running inference-enabled SPARQL queries may sometimes require unexpectedly long execution time. Therefore, demand has increased to make them more usable by slightly changing their queries, which could produce an acceptable level of similar results. In this demonstration, we present our query-approximation system that can transform an inference-enabled federated SPARQL query into another one that can produce acceptably similar results without unexpectedly long runtimes to avoid timeout on executing inference-enabled federated SPARQL queries.

Keywords: SPARQL, inference, federated query, ontology mapping

1 Introduction

Reasoning on LODs allows queries to obtain unstated knowledge from a distinct one [1]. Techniques to utilize reasoning capability based on ontology have been developed to overcome several issues, such as higher complexity in the worst case [5][7][8][9].

When a query prepared by the client might require a long execution time, a standard SPARQL endpoint implementation will try to execute the query with lots of cost to return answers. If the endpoint receives lots of heavy queries, it might spend much time on their execution or, more severely, it might cause a server-down. This is especially important for endpoints that have inference engines to support OWL reasoning capability.

In this paper, we present an idea and its prototype implementation of a query-approximation system that can transform an inference-enabled federated SPARQL query into another one that can produce acceptably similar results without unexpectedly long runtimes to avoid timeout on executing inference-enabled federated SPARQL queries.¹

2 Background

In [7], Kang et al. introduced a number of metrics that can be used to predict reasoning performance and evaluated various classifiers to know how accurately

¹ A demonstration is available at <http://whitebear.cs.inf.shizuoka.ac.jp/Yaseki/>

they predict classification time for an ontology based on its metric values. According to their evaluation results, they have prepared prediction models with accuracy of more than 80%, but there are still major difficulties in improving them.

In [5], it was introduced that reasoning tasks on ontologies constructed from an expressive description logic have a high worst-case complexity. It has been done by analyzing experimental results that divided each of several ontologies into four and eight random subsets of equal size and measuring classification times of these subsets as increments. They reported that some ontologies exhibit non-linear sensitivity on their inference performance. They also argued that there is no straightforward relationship between the performance of a subset of each isolated ontology and the contribution of each subset to the whole inference performance on the whole ontology, while they provided an algorithm that identifies an ontology's hot spots.

There are two possible approaches to managing long-running queries. One is to utilize parallel and distributed computing techniques to make those executions faster [10]. Another possible approach is rewriting a query that requires long execution time to a light-weight one. There are some query rewriting approaches to improve the quality of queries [2][3][4]. Also, there are some heuristic techniques to approximate inference-enabled queries by modifying some hotspots in the query that prevent faster execution [12]. However, since those hotspots are also dependent on their individual ontologies, such query modification should take into account both query-structure and characteristics of the ontologies used.

3 Outline and System Architecture

If a query seems not to be a time-consuming one, the endpoint executes the query. If the query execution is classified as time-consuming, the endpoint may have an option to reject the execution of the query or transform that query into an optimized one. To implement such behaviors in an endpoint, some extensions should be provided to allow a notification to the client that the received query has been transformed into another one, or the query has been rejected due to a heavy-load condition.

To realize the idea, we are implementing a preliminary system to classify whether a query execution is time-consuming or not, rewriting the query to a more light-weight one, and extending the protocol to notify the rejection of the query, the applied query-transformation for the query, and so on. We applied a pattern-based heuristic query rewriting technique that, for example, substitutes some named classes to subsets of their potential classes that are derived by the inference. Our prototype system has a unique proxy module called "Front-end EP" between the client and the endpoint (called "Back-end EP" in this paper). Figure 1 shows a brief overview of the query execution process mediated by a Front-end EP. Figure 2 shows the basic procedure of query processing on our system. Table 1 shows our preliminary evaluation on the heavy-query detection

on a single endpoint configuration shown in [12]. Here, we used Linklings ontology from the OAEI dataset in the preliminary experiment.

To prepare datasets to evaluate the performance sensitivity of ontology-level simplification techniques, we reduced Linklings ontology by cutting several relational descriptions and added 10 instances for each named class. As an experimental environment, we set up a SPARQL endpoint using Joseki (v3.4.4) in conjunction with a server-side reasoner using Pellet [11] to enable OWL-level inference capability on the endpoint. In this experiment, we used 100 ms as the threshold time. The evaluation data set was generated by queries to get the instances of a named class in the Linklings ontology. Here, we conducted an experiment for all 1,369 queries on N-fold cross validation. We used two classifiers: Bagged C4.5 and Boosted C4.5, implemented in Weka [6] with default parameters. Further evaluation of the performance on multiple-endpoint configurations remains as future work.

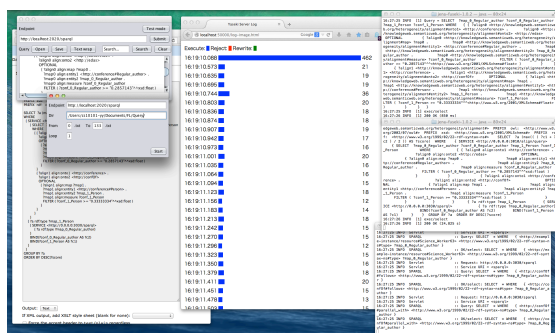


Fig. 1. Overview of Our System

Table 1. Classification Performance on Our Approach (N-fold Cross Validation)[12]

Classifier	Recall	Precision	F-Measure
Bagged C4.5	0.959	0.977	0.964
Boosted C4.5	0.999	0.999	0.999

References

1. Baader, F., Suntisrivaraporn, B.: Debugging SNOMED CT Using Axiom Pinpointing in the Description Logic \mathcal{EL}^+ . In: Cornet, R., Spackman, K. (eds.) Representing and sharing knowledge using SNOMED. Proceedings of the 3rd International Conference on Knowledge Representation in Medicine KR-MED 2008, vol. 410, pp. 1–7. CEUR-WS (2008)
2. Bischof, S., Polleres, A.: RDFS with Attribute Equations via SPARQL Rewriting. In: Cimiano, P., Corcho, O., Presutti, V., Hollink, L., Rudolph, S. (eds.) The Semantic Web: Semantics and Big Data. LNCS, vol. 7882, pp. 335–350. Springer-Verlag (2013)
3. Fujino, T., Fukuta, N.: SPARQLoid - a Querying System using Own Ontology and Ontology Mappings with Reliability. In: Proc. of the 11th International Semantic Web Conference (Poster & Demos) (ISWC 2012) (2012)

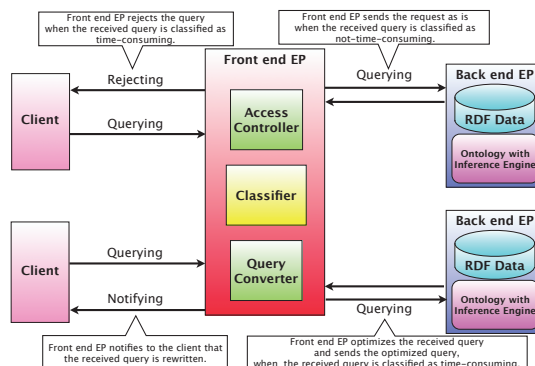


Fig. 2. Basic Query Processing Procedure

4. Fujino, T., Fukuta, N.: Utilizing Weighted Ontology Mappings on Federated SPARQL Querying. In: Kim, W., Ding, Y., Kim, H.G. (eds.) The 3rd Joint International Semantic Technology Conference (JIST2013). LNCS, vol. 8388, pp. 331–347. Springer International Publishing (2013)
5. Gonçalves, R.S., Parsia, B., Sattler, U.: Performance Heterogeneity and Approximate Reasoning in Description Logic Ontologies. In: Cudré-Mauroux, P., Heflin, J., Sirin, E., Tudorache, T., Euzenat, J., Hauswirth, M., Parreira, J.X., Hendler, J., Schreiber, G., Bernstein, A., Blomqvist, E. (eds.) The Semantic Web–ISWC 2012 Part I. LNCS, vol. 7649, pp. 82–98. Springer-Verlag (2012)
6. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The WEKA Data Mining Software: An Update. ACM SIGKDD explorations newsletter 11(1), 10–18 (2009)
7. Kang, Y.B., Li, Y.F., Krishnaswamy, S.: Predicting Reasoning Performance Using Ontology Metrics. In: Cudré-Mauroux, P., Heflin, J., Sirin, E., Tudorache, T., Euzenat, J., Hauswirth, M., Parreira, J.X., Hendler, J., Schreiber, G., Bernstein, A., Blomqvist, E. (eds.) The Semantic Web–ISWC 2012 Part I. LNCS, vol. 7649, pp. 198–214. Springer-Verlag (2012)
8. Motik, B., Shearer, R., Horrocks, I.: Hypertableau Reasoning for Description Logics. *Journal of Artificial Intelligence Research* 36, 165–228 (2009)
9. Romero, A.A., Grau, B.C., Horrocks, I.: MORE: Modular Combination of OWL Reasoners for Ontology Classification. In: Cudré-Mauroux, P., Heflin, J., Sirin, E., Tudorache, T., Euzenat, J., Hauswirth, M., Parreira, J.X., Hendler, J., Schreiber, G., Bernstein, A., Blomqvist, E. (eds.) The Semantic Web–ISWC 2012 Part I. LNCS, vol. 7649, pp. 1–16. Springer (2012)
10. Schätzle, A., Przyjaciół-Zablocki, M., Hornung, T., Lausen, G.: PigSPARQL: A SPARQL Query Processing Baseline for Big Data. In: Proc. of the 12th International Semantic Web Conference (Poster & Demos) (ISWC 2013) (2013)
11. Sirin, E., Parsia, B., Grau, B.C., Kalyanpur, A., Katz, Y.: Pellet: A practical owl-dl reasoner. *Journal of Web Semantics* 5(2), 51 – 53 (2007)
12. Yamagata, Y., Fukuta, N.: A Dynamic Query Optimization on a SPARQL Endpoint by Approximate Inference Processing . In: Proc. of 5th International Conference on E-Service and Knowledge Management (ESKM 2014). pp. 161–166 (2014)

VKGBuilder – A Tool of Building and Exploring Vertical Knowledge Graphs

Tong Ruan, Haofen Wang, and Fanghui Hu

East China University of Science & Technology, Shanghai, 200237, China
{ruantong,whfcarter}@ecust.edu.cn, xiaohuqi@126.com

Abstract. Recently, search engine companies like Google and Baidu are building their own knowledge graphs to empower the next generation of Web search. Due to the success of knowledge graphs in search, customers from vertical sectors are eager to embrace KG related technologies to develop domain specific semantic platforms or applications. However, they lack skills or tools to achieve the goal. In this paper, we present an integrated tool VKGBuilder to help users manage the life cycle of knowledge graphs. We will describe three modules of VKGBuilder in detail which construct, store, search and explore knowledge graphs in vertical domains. In addition, we will demonstrate the capability and usability of VKGBuilder via a real-world use case in the library industry.

1 Introduction

Recently, an increasing amount of semantic data sources are published on the Web. These sources are further interlinked to form Linking Open Data (LOD). Search engine companies like Google and Baidu leverage LOD to build their own semantic knowledge bases (called *knowledge graphs*¹) to empower semantic search. The success of KGs in search attracts much attention from users in vertical sectors. They are eager to embrace related technologies to build semantic platforms in their domains. However, they either lack skills to implement such platforms from scratch or fail to find sufficient tools to accomplish the goal.

Compared with general-purpose KGs, knowledge graphs in vertical industries (denoted as VKG) have the following characteristics: a) *More accurate and richer data* of certain domains to be used for business analysis and decision making; b) *Top-down construction* to ensure the data quality and stricter schema while general KGs are built in a bottom-up manner with more emphasis on the wide coverage of data from different domains; c) *Internal data stored in RDBs* are further considered to be integrated into VKGs; and d) Besides search, VKGs should provide *user interfaces* especially for KG construction and maintenance.

While there exist tool suites (e.g., LOD2 Stack²) which help to build and explore LOD, these tools are mainly developed for researchers and developers of the Semantic Web community. Vertical markets, on the other hand, need

¹ http://en.wikipedia.org/wiki/Knowledge_Graph

² <http://lod2.eu/>

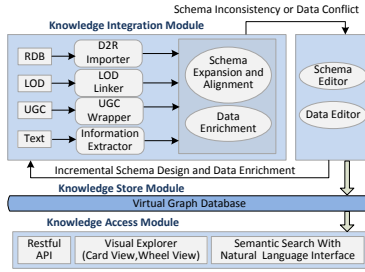


Fig. 1. Architecture of VKGBuilder



Fig. 2. Semantic Search Interface

end-to-end solutions to manage the life cycle of knowledge graphs and hide the technical details as much as possible. To the best of our knowledge, we present the first suitable tool for vertical industry users called VKGBuilder. It allows *rapid and continuous VKG construction* which imports and extracts data from diverse data sources, provides a mechanism to detect intra- and inter-data source conflicts, and consolidates these data into a consistent KG. It also provides *intuitive and user-friendly interfaces* for novice users with little knowledge of semantic technologies to understand and exploit the underlying VKG.

2 Description of VKGBuilder

VKGBuilder is composed of three modules namely the *Knowledge Integration* module, the *Knowledge Store* module, and the *Knowledge Access* module. The whole architecture is shown in Figure 1. Knowledge Integration is the core module for VKG construction with three main components. Knowledge Store is a virtual graph database which combines RDBs, in-memory stores and inverted indexes to support fast access of VKG in different scenarios, and the Knowledge Access module provides different interfaces for end users and applications.

2.1 Knowledge Integration Module

- *Data Importers and Information Extractors.* Structured data from internal relational database are imported and converted into RDF triples by D2R importers³. A *LOD Linker* is developed to enrich VKG with domain ontologies from the public linked open data. For the user generated contents (UGCs), we mainly consider encyclopaedic sites like Wikipedia, Baidu Baike, and Hudong Baike. Due to the semi-structured nature of these sites, *wrappers* automatically extract properties and values of certain entities. As for unstructured text, distant-supervised learning methods are adapted to discover missing relations between entities or fill property values of a given entity where the above extracted semantic data serve as seeds.

³ <http://d2rq.org/>

- *Schema Inconsistency and Data Conflict Detection.* After semantic data are extracted or imported from various sources, data integration is performed to build an integrated knowledge graph. During integration, schema-level inconsistency and data-level conflicts might occur. Schema editing is used to define axioms of properties such as (e.g., functional, inverse, transitive), concept subsumptions, and concepts of entities. Then a rule-based validator is triggered to check whether the newly added data or imported ontologies will cause any conflicts with existing ones. The possible conflicts are resolved by user defined rules or delivered to domain experts for human intervention.
- *Schema and Data Editor.* Knowledge workers can extend or refine a VKG in both schema-level and data-level with a collaborative editing interface.

2.2 Knowledge Access Module

- *Visual Explorer.* It includes three views namely the *Wheel View*, the *Card View*, and the *Detail View*. The Wheel View organizes concepts and entities in two wheels. In the left wheel, the node of interest is displayed in the center. If it is a concept, its child concepts are neighbors in the same wheel. If it is an entity, its related entities are connected via properties as outgoing (or incoming) edges. When a related concept (or entity) is clicked, the right wheel is expanded with the clicked node in the center surrounded with its related information on the VKG. Thus, we allow users to navigate through the concept hierarchy and traverse between different entities. The Card View visualizes entities in a force-directed graph layout, which is similar to the galaxy visualization in a 3D space. The Card View also allows to change the focus through drag and drop as well as zoom-in and zoom-out. The Detailed View shows all properties and property values of a particular entity. The three views can be switched from one to another in a flexible way.
- *Semantic Search with Natural Language Interface.* Users can submit any keyword query or natural language question. The query is interpreted into possible SPARQL queries with natural language descriptions. Once a SPARQL query is selected, the corresponding answers are returned, along with relevant documents which contain semantic annotations on these answers. Besides, a summary (a.k.a, knowledge card) of the main entity mentioned in the query or the top-ranked answer is shown. Related entities defined in the VKG as well as correlated entities in the query log are recommended.
- *Restful APIs.* They are designed for developers with little knowledge of semantic technologies to access the VKG using any programming language from any platform at ease. These APIs are actually manipulations of SPARQL queries to support graph traversal or sub-graph matching on the VKG.

3 Demonstration

VKGBuilder is first used in the ZhouShan Library. The current VKG (marine-oriented KG) contains more than 32,000 fishes and each fish has more than 20

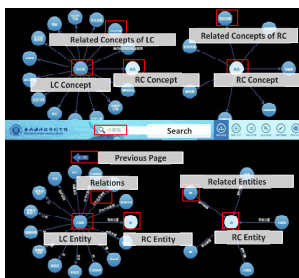


Fig. 3. Wheel View

Source	Value	Conflict Values
Source 1	Value 1	Value 1
Source 2	Value 2	Value 2
Source 3	Value 3	Value 3
Source 4	Value 4	Value 4
Source 5	Value 5	Value 5
Source 6	Value 6	Value 6
Source 7	Value 7	Value 7
Source 8	Value 8	Value 8
Source 9	Value 9	Value 9
Source 10	Value 10	Value 10
Source 11	Value 11	Value 11
Source 12	Value 12	Value 12
Source 13	Value 13	Value 13
Source 14	Value 14	Value 14
Source 15	Value 15	Value 15
Source 16	Value 16	Value 16
Source 17	Value 17	Value 17
Source 18	Value 18	Value 18
Source 19	Value 19	Value 19
Source 20	Value 20	Value 20
Source 21	Value 21	Value 21
Source 22	Value 22	Value 22
Source 23	Value 23	Value 23
Source 24	Value 24	Value 24
Source 25	Value 25	Value 25
Source 26	Value 26	Value 26
Source 27	Value 27	Value 27
Source 28	Value 28	Value 28
Source 29	Value 29	Value 29
Source 30	Value 30	Value 30
Source 31	Value 31	Value 31
Source 32	Value 32	Value 32
Source 33	Value 33	Value 33
Source 34	Value 34	Value 34
Source 35	Value 35	Value 35
Source 36	Value 36	Value 36
Source 37	Value 37	Value 37
Source 38	Value 38	Value 38
Source 39	Value 39	Value 39
Source 40	Value 40	Value 40
Source 41	Value 41	Value 41
Source 42	Value 42	Value 42
Source 43	Value 43	Value 43
Source 44	Value 44	Value 44
Source 45	Value 45	Value 45
Source 46	Value 46	Value 46
Source 47	Value 47	Value 47
Source 48	Value 48	Value 48
Source 49	Value 49	Value 49
Source 50	Value 50	Value 50

Fig. 4. Conflict Resolution

properties. Besides fishes, VKGBuilder also captures knowledge about fishing grounds, fish processing methods, related researchers and local enterprises. An online demo video of VKGBuilder can be downloaded at <http://202.120.1.49:19155/SSE/video/VKGBuilder.wmv>.

Figure 2 shows a snapshot of the semantic search interface. When a user enters a query “Distribution of Little Yellow Croaker”, VKGBuilder first segments the query into “Little Yellow Croaker” and “Distribution”. Here, “Little Yellow Croaker” is recognized as a fish, and properties about “distribution” are returned. Then all sub-graphs connecting the fish with each property are found as possible SPARQL query interpretations of the input query. Top interpretations whose scores are above a threshold are returned with natural language descriptions for further selection. Once a user selects a query, the answers (e.g., China East Sea) are returned. Also, related books with these answers as semantic annotations are returned. The related library classification of these books are displayed in the left, and the knowledge card as well as related concepts and entities of Little Yellow Croaker are listed in the right panel.

In Figure 3, the Wheel View initially shows the root concept (owl:Thing) in the center of the left wheel (denoted as LC). When a sub-concept **Fish** is clicked, it becomes the center of the right wheel (denoted as RC) with its child concepts (e.g., **Chondrichthyes**). We can also navigate between entities. For instance, **selenium** is one of the nutrients of Little Yellow Croaker. When clicking **selenium**, all fishes containing this nutrient are shown in the right wheel.

The user experience heavily depends on the quality of the underlying VKG. The extraction and importing are executed automatically in the back-end while we provide a user interface for conflict resolution. For “Little Yellow Croaker”, we extract **Ray-finned Fishes** and **Actinopterygii** from different sources as values of the property **Class** in the scientific classification. Since **Class** is defined as a functional property and the two values do not refer to the same thing, a conflict occurs. As shown in Figure 4, VKGBuilder accepts **Actinopterygii** as the final value because this value is extracted from more trusted sources.

Acknowledgements This work is funded by the National Key Technology R&D Program through project No. 2013BAH11F03.

Using the semantic web for author disambiguation - are we there yet?

Cornelia Hedeler¹, Bijan Parsia¹, and Brigitte Mathiak²

¹ School of Computer Science, The University of Manchester, Oxford Road,
M13 9PL Manchester, UK,

{chedeler,bijan.parsia}@manchester.ac.uk

² GESIS - Leibniz Institute for the Social Sciences, Unter Sachsenhausen 6-8,
50667 Cologne, Germany

brigitte.mathiak@gesis.org

Abstract. The quality, and therefore, the usability and reliability of data in digital libraries depends on author disambiguation, i.e., the correct assignment of publications to a particular person. Author disambiguation aims to resolve name ambiguity, i.e., synonyms (the same author publishing under different names), and polysemes (different authors with the same name), and assign publications to the correct person. However, author disambiguation is difficult given that the information available in digital libraries is sparse and, when integrated from multiple data sources, contain inconsistencies in representation, e.g., of person names, or venue titles. Here we analyse and evaluate the usability of person-centred reference data available as linked data to complement the information present in digital libraries and aid author disambiguation.

1 Introduction

Users of digital libraries are not only interested in literature related to a particular topic or research field of interest, but more frequently also in literature written by a particular author [2]. However, as digital libraries tend to integrate information from various sources, they suffer from inconsistencies in representation of, e.g., author names or venue titles, despite best efforts to maintain a high data quality. For the actual disambiguation process, a wide variety of additional metadata are used, e.g., journal or conference names, author affiliations, co-author networks, and keywords or topics [1, 6]. However, in some digital libraries the available metadata can be quite sparse, providing insufficient amount and detail of information to disambiguate authors efficiently.

To complement the sometimes sparse bibliographic information a number of approaches surveyed in [1] utilise information available elsewhere, e.g., using web searches, and most of the approaches proposed are evaluated utilising gold standard datasets of high quality, such as Google Scholar author profiles. However, to the best of the authors' knowledge, these high quality data sets have so far not been used as part of the disambiguation process itself. Here we analyse person-centred reference data available on the semantic web and evaluate whether it contains sufficient detail and content to provide additional information and aid author disambiguation.

2 Data sets

2.1 Digital library data sets

In contrast to the wealth of metadata available in some digital libraries, the records in the two digital library data sets used here only offer limited metadata. **DBLP** Most publication records in the DBLP Computer Science Bibliography [4] consist only of author names, publication titles, and venue information, such as names of conferences and journals. In addition to the publication records, DBLP also contains person records, which are created as result of ongoing efforts for author disambiguation [5].

Sowiport The portal Sowiport(<http://sowiport.gesis.org>) is provided by GESIS and contains publication records relevant to the social sciences. Here we only focus on a subset of just over 500,000 literature entries in Sowiport from three data sources (SOFIS, SOLIS, SSOAR) within GESIS, that have been annotated with keywords from TheSoz, a German thesaurus for the Social Sciences.

So far, no author disambiguation has taken place in these records, and inconsistencies in particular in author names make it hard for users to find all publications by a particular author. An analysis of the search logs has shown that the authors most frequently searched for are those with large numbers of publications, who tend to have entries in DBpedia and GND, motivating the use of the reference data sources introduced below.

2.2 Person-centred reference data

GND authority file and GND publication information. As the literature in Sowiport, in particular the subset used here, is heavily biased towards German literature, we use the Integrated Authority File (GND) of the German-speaking countries and the bibliographic data offered as part of the linked data service by the German National Library (<http://www.dnb.de/EN/lds>). Amongst other information, which also includes keywords, the GND file contains differentiated person records, which refer to a real person, and are used here.

DBpedia [3] is available for download(<http://wiki.dbpedia.org/Downloads39>) and comes in various data sets containing different kinds of data, amongst them ‘Persondata’, with information about people, such as their date and place of birth and death. As the persondata subset itself does not contain much additional detail, other data sets are required to obtain information useful for author disambiguation. The data is available either as raw infobox data or cleaned mapping-based data, which we use here.

3 Approach for author disambiguation

Our approach for author disambiguation can be seen as preliminary, as the main focus of this work was to evaluate whether there is sufficient information available in such reference data sets to make this a viable approach. It uses a domain specific heuristic as similarity function, and the reference data sets introduced above as additional (web information) evidence. To limit the number of records that need to be compared in detail, we use an index on the author/person names

Table 1. left: Number of person records in GND with selected professions; right: Number of instances in persondata in DBpedia for selected classes (y = yago).

Profession	# person	Class	English	German
author / female author	8,319 / 6,301	#foaf:person	1,055,682	479,201
lecturer / female lecturer	7,931 / 1,204	#dbpedia-owl:person	652,031	215,585
research associated / female	1,117 / 759	#y:person	844,562	0
physicist / female physicist	11,595 / 1,280	#dbpedia-owl:scientist	15,399	0
mathematician / female	7,561 / 908	#y:Scientist110560637	44,033	0
computer scientist / female	5,443 / 589	#y:ComputerScientist109951070	1,667	0
sociologists / female sociologist	3,298 / 1,590	#y:Mathematician110301261	4,994	0
social scientist / female	998 / 546	#y:Physicist110428004	6,020	0
		#y:SocialScientist110619642	9,083	0
		#y:Philosopher110423589	6,116	0
		#dbpedia-owl:Philosopher	1,276	0

Table 2. Number of person instances in DBpedia with selected properties of relevance.

Property	English	German	Property	English	German
Author names			Co-authors		
foaf:Name	1,055,682	479,201	dbpedia-owl:academicAdvisor	508	0
rdfs:label	1,055,682	479,201	dbpedia-owl:doctoralAdvisor	3,698	0
dbpedia-owl:birthName	44,977	285	dbpedia-owl:doctoralStudent	1,791	0
dbpedia-owl:pseudonym	1,865	0	dbpedia-owl:notableStudent	372	0
Author affiliation			dbpedia-owl:influenced	2,830	0
dbpedia-owl:almaMater	42,318	0	dbpedia-owl:influencedBy	5,928	0
dbpedia-owl:employer	3,232	0	Keywords or topics / research area		
dbpedia-owl:school	1,974	0	dbpedia-owl:knownFor	17,702	0
dbpedia-owl:university	1,073	0	dbpedia-owl:notableIdea	392	0
dbpedia-owl:institution	923	0	dbpedia-owl:field	17,831	0
dbpedia-owl:college	13,510	1,829	dbpedia-owl:significantProject	614	0

for the records in each of the data sources. We preprocess the author names to make the representation of names consistent across all data sources. The search over the index allows for slight spelling variations, the presence of only the initial of the forename, missing middle names, and a swap in the order of the fore- and surnames. The decision of whether a person record is considered to be sufficiently similar to the author of a publication record is currently based on a domain specific heuristic, and can be improved. However, the algorithm only serves as a test-bed to assess whether GND and DBpedia provide sufficiently detailed information to be used for author disambiguation.

4 Analysis and Evaluation

Analysis of GND and DBpedia. In addition to the (incomplete) list of publications of a person, GND contains additional information that could characterise a person sufficiently, including subject categories, their profession, and keywords for their publications. Unlike the GND authority file and the additional publication records, which are maintained by a library, and therefore, are structured and contain data more akin to digital libraries, DBpedia was not developed for that purpose, resulting in the required information being less readily available. In addition, the German part of DBpedia contains significantly less of the information useful for author disambiguation (see Table 1 right and Table 2).

Evaluation To determine whether the lack of more detailed information has a negative effect on the performance of author disambiguation using these reference data, we have taken the following data sets: (i) A manually created small test data set consisting of 30 of the top social scientists with a differentiated entry in GND, and DBpedia. (ii) A random subset of 250 computer scientists from

person records in DBLP with a link to the corresponding wikipedia page, and run the part of the author disambiguation algorithm that identifies the GND and DBpedia entries of an author of a publication record in Sowiport and DBLP. The precision ranging between 0.7 and 1 is encouraging (In detail: social scientists with entry in German DBpedia: 0.97; - with entry in English DBpedia: 1; - with entry in GND: 0.92; computer scientists with entry in DBpedia: 0.89 taking into account the language of the false positives; - with entry in GND: 0.7). However, the data set used here is fairly small and does not contain too many people with common names, which contribute the majority of the false positives.

5 Discussion

The analysis and evaluation of DBpedia and GND has shown that the semantic markup of the information in DBpedia is still lacking in various aspects. How much of an issue this lack of appropriately detailed information and lack of completeness really causes for tasks does not only depend on the corresponding subset of the reference data and its properties, but also on the remainder of the reference data set, and the digital library data set. This would suggest that a quality measure that assesses the suitability of the reference data set for author disambiguation should take into account the following: (i) tuple completeness, (ii) specificity of the annotation with ontologies, (iii) how much of the information is provided in form of ontologies or thesaurus or even worse literal strings, which provides an indication of the expected heterogeneity of the information across different data sets, and (iv) the number of people in the reference data set who share their names.

To bring this into context with the digital library data set, one could also determine whether and how many of the author names are shared with several person records in the reference data set. In particular in these cases, sufficiently detailed information is vital in order to be able to identify the correct person record or determine that there is no person record available for that particular person, even though there are plenty of records for people with the same name.

References

1. Ferreira, A.A., Gonçalves, M.A., Laender, A.H.F.: A brief survey of automatic methods for author name disambiguation. *SIGMOD Record* 41(2) (2012)
2. Herskovic, J.R.J., Tanaka, L.Y.L., Hersh, W.W., Bernstam, E.V.E.: A day in the life of PubMed: analysis of a typical day's query log. *Journal of the American Medical Informatics Association : JAMIA* 14(2), 212–220 (2007)
3. Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P.N., Hellmann, S., Morse, M., van Kleef, P., Auer, S.: Dbpedia-a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web Journal* (2014)
4. Ley, M.: DBLP: some lessons learned. In: *VLDB'09*. pp. 1493–1500 (2009)
5. Reuther, P., Walter, B., Ley, M., Weber, A., Klink, S.: Managing the Quality of Person Names in DBLP. In: *ECDL'06*. pp. 508–511 (2006)
6. Smalheiser, N.R., Torvik, V.I.: Author name disambiguation. *Annual review of information science and technology* 43(1) (2009)

SHEPHERD: A Shipping-Based Query Processor to Enhance SPARQL Endpoint Performance

Maribel Acosta¹, Maria-Esther Vidal², Fabian Flöck¹,
Simón Castillo², Carlos Buil-Aranda³, and Andreas Harth¹

¹ Institute AIFB, Karlsruhe Institute of Technology, Germany
{maribel.acosta, fabian.floeck, harth}@kit.edu

² Universidad Simón Bolívar, Venezuela
{mvidal, scastillo}@ldc.usb.ve

³ Department of Computer Science, Pontificia Universidad Católica, Chile
cbuil@ing.puc.cl

Abstract. Recent studies reveal that publicly available SPARQL endpoints exhibit significant limitations in supporting real-world applications. In order for this querying infrastructure to reach its full potential, more flexible client-server architectures capable of deciding appropriate shipping plans are needed. Shipping plans indicate how the execution of query operators is distributed between the client and the server. We propose SHEPHERD, a SPARQL client-server query processor tailored to reduce SPARQL endpoint workload and generate shipping plans where costly operators are placed at the client site. We evaluated SHEPHERD on a variety of public SPARQL endpoints and SPARQL queries. Experimental results suggest that SHEPHERD can enhance endpoint performance while shifting workload from the endpoint to the client.

1 Introduction

Nowadays, public SPARQL endpoints are widely deployed as one of the main mechanisms to consume Linked Data sets. Although endpoints are acknowledged as a promising technology for RDF data access, a recent analysis by Buil-Aranda et al. [1] indicates that performance and availability vary notably between different public endpoints. One of the main reasons for the at times undesirable performance of public SPARQL endpoints is the unpredictable workload, since a large number of clients may be concurrently accessing the endpoint and some of the queries handled by endpoints may incur prohibitively high computational costs. To relieve endpoints of some of the workload they face, many operators of the query can potentially be executed at the client side. Shipping policies [2] allow for deciding which parts of the query will be executed at the client or the server according to the abilities of SPARQL endpoints.

The goal of this work is to provide a system to access SPARQL endpoints that shifts workload from the server to the client taking into account the capabilities of the addressed endpoint for executing a certain query – while still offering a competitive performance in terms of execution time and the number of answers produced. We propose SHEPHERD, a SPARQL query processor that mitigates the workload posed to public SPARQL endpoints by tailoring hybrid shipping plans to every specific endpoint. In particular, SHEPHERD performs the following tasks: (i) decomposing SPARQL

queries into lightweight sub-queries that will be posed against the endpoint, (ii) traversing the plan space in terms of formal properties of SPARQL queries, and (iii) generating shipping-based query plans based on the public SPARQL endpoint performance statistics collected by SPARQLES [1]. We designed a set of 20 different SPARQL queries over four public SPARQL endpoints. We empirically analyzed the performance of the hybrid shipping policies devised by SHEPHERD and the query shipping policy when submitting a query directly to a SPARQL endpoint.

2 The SHEPHERD Architecture

SHEPHERD is a SPARQL query processor based on the wrapper architecture [4]. SHEPHERD implements different shipping policies to reduce the workload posed over public SPARQL endpoints. Figure 1 depicts the SHEPHERD architecture which consists of three core components: the SHEPHERD optimizer, the engine broker, and the SPARQL query engine that is considered a *black box* component.

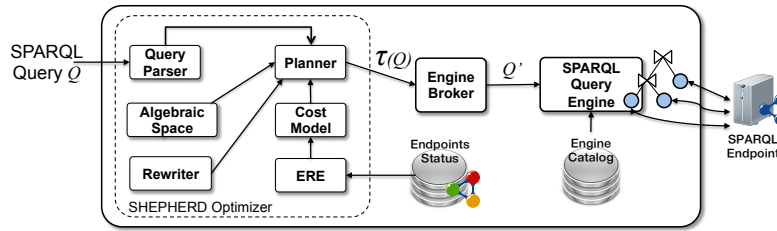


Fig. 1. The SHEPHERD architecture

The SHEPHERD optimizer is designed for enhancing SPARQL query plans since it relies on formal properties of SPARQL and statistics of SPARQL endpoints to estimate the plan cost. In the following, we elaborate on each of the sub-components that comprise the proposed optimizer.

- **Query parser:** Translates the input query Q into internal structures that will be processed by the planner.
- **Planner:** Implements a Dynamic Programming algorithm to traverse the space of plans. During the optimization process, SHEPHERD decides whether to place the operators at the server (i.e., endpoint), or client (i.e., SHEPHERD) according to statistics of the endpoint. In this way, SHEPHERD explores shipping policies tailored for each public endpoint. The planner generates bushy-tree plans, where the leaves correspond to light-weight sub-queries and the nodes correspond to operators (annotated with the shipping policy to follow).
- **Algebraic space and rewriter:** The algebraic space defines a set of algebraic rules to restrict plan transformations. The algebraic rules correspond to the formal properties for well-designed SPARQL patterns [3]. The rewriter transforms a query in terms of the algebraic space to produce a more efficient equivalent plan.
- **Cost model:** The cost of executing sub-queries and SPARQL operators at the endpoint is obtained from the ERE component. Based on these values, SHEPHERD

estimates the cost of combining sub-plans with a given operator. The cost model is designed to gradually increase the cost of operators when more complex expressions are evaluated. This behavior is modeled with the Boltzmann distribution.¹

- **Endpoint Reliability Estimator (ERE)**: Endpoint statistics collected by the SPARQLES tool [1] are used to provide reliable estimators for the SHEPHERD cost model. The endpoint statistics are aggregated and stored in the SHEPHERD catalog, and used to characterize endpoint in terms of operator performance.

The engine broker translates the optimized plan $\tau(Q)$ into the corresponding input for the SPARQL query engine that will execute the plan. The engine broker can specify the plan in two different ways: i) as a query Q' with the SPARQL Federation Extension, to execute the query with a SPARQL 1.1 engine; ii) translating the plans directly into the internal structures of a given query engine.

3 Experimental Study

We empirically compared the performance of the hybrid shipping policies implemented by SHEPHERD with the query shipping policy when executing queries directly against the endpoint. We selected the following four public SPARQL endpoints monitored by SPARQLES [1]: DBpedia, IproBio2RDF, data.oceandrilling.org and TIP.² We designed a query benchmark comprising five different queries for each endpoint.³ Each query contains modifiers as well as graph patterns that include UNIONS, OPTIONALS, and filters. SHEPHERD was implemented using Python 2.7.6. Queries were executed directly against the endpoints using the command `curl`. All experiments were performed from the Amazon EC2 Elastic Compute Cloud infrastructure.⁴

Figure 2 depicts the result of the queries in terms of the execution time (sec.) as reported by the Python `time.time()` function. We can observe that in the majority of the cases SHEPHERD retrieves the results notably faster, except in three queries. Concerning the cardinality of the result set retrieved, a similar picture emerges. For 18 of the overall 20 queries tested, both methods produced the same amount of results, while in one instance each SHEPHERD and the query shipping approach did not retrieve any answers. Both methods therefore seem to be on par in this regard.

Even though SHEPHERD is able to reduce the execution time to a certain extent, the most important finding is that it shifted in average 26% of the operators in the queries to the client, thereby relieving the servers of notable workload. The ratio of data shipping varies significantly from case to case depending on the individual shipping strategy chosen, but does not show a direct correlation with the achieved runtime decrease.⁵

Hence, we can affirm that SHEPHERD is able to reduce computational load on the endpoints in an efficient way; this could not be achieved neither by simply moving all

¹ The Boltzmann distribution is also used in Simulated Annealing to model the gradual decreasing of a certain function (temperature).

² Available at <http://dbpedia.org/sparql>, <http://iproclass.bio2rdf.org/sparql>, <http://data.oceandrilling.org/sparql> and <http://lod.apc.gov.tw/sparql>, respectively.

³ <http://people.aifb.kit.edu/mac/shepherd/>

⁴ <https://aws.amazon.com/ec2/instance-types/>

⁵ Std.Dev. is 0.09. The poster will discuss further details about this ratio and its impact.

operator execution to the client – since this increments the bandwidth consumption and the evaluation of non-selective queries may starve the resources of the client – nor by submitting the whole query to the endpoint as shown in our experiments.

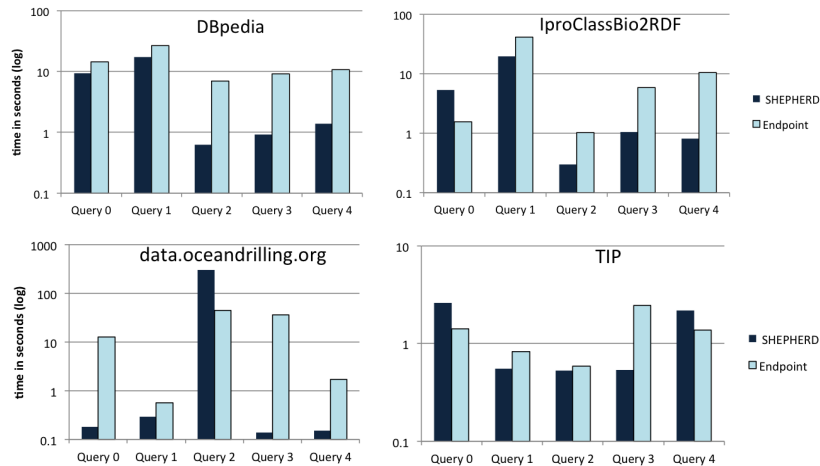


Fig. 2. Runtime results for the four different public endpoints and studied queries

4 Conclusions

We presented SHEPHERD, a SPARQL query processor that implements hybrid shipping policies to reduce public SPARQL endpoint workload. We crafted 20 different queries against four SPARQL endpoints and empirically demonstrated that SHEPHERD is (i) able to adapt to endpoints with different characteristics by varying the rate of operators executed at the client, and (ii) in doing so not only retrieves the same number of results as query shipping but even decreases runtime in the majority of the cases. While these results provide a first insight into SHEPHERD’s capabilities, they showcase the potential of adaptive hybrid shipping approaches, which we will explore in future work.

Acknowledgements

The authors acknowledge the support of the European Community’s Seventh Framework Programme FP7-ICT-2011-7 (XLike, Grant 288342).

References

1. C. B. Aranda, A. Hogan, J. Umbrich, and P.-Y. Vandenbussche. SPARQL web-querying infrastructure: Ready for action? In *International Semantic Web Conf. (2)*, pp. 277–293, 2013.
2. M. J. Franklin, B. T. Jónsson, and D. Kossmann. Performance tradeoffs for client-server query processing. In *SIGMOD Conference*, pp. 149–160, 1996.
3. J. Pérez, M. Arenas, and C. Gutierrez. Semantics and complexity of SPARQL. *ACM Trans. Database Syst.*, 34(3):16:1–16:45, Sept. 2009.
4. G. Wiederhold. Mediators in the architecture of future information systems. *IEEE Computer*, 25(3):38–49, 1992.

AgreementMakerLight 2.0: Towards Efficient Large-Scale Ontology Matching

Daniel Faria¹, Catia Pesquita^{1,2}, Emanuel Santos², Isabel F. Cruz³, and Francisco M. Couto^{1,2}

¹ LASIGE, Faculdade de Ciências, Universidade de Lisboa, Portugal

² Dept. Informática, Faculdade de Ciências, Universidade de Lisboa, Portugal

³ ADVIS Lab, Dept. of Computer Science, University of Illinois at Chicago, USA

Abstract. Ontology matching is a critical task to realize the Semantic Web vision, by enabling interoperability between ontologies. However, handling large ontologies efficiently is a challenge, given that ontology matching is a problem of quadratic complexity.

AgreementMakerLight (AML) is a scalable automated ontology matching system developed to tackle large ontology matching problems, particularly for the life sciences domain. Its new 2.0 release includes several novel features, including an innovative algorithm for automatic selection of background knowledge sources, and an updated repair algorithm that is both more complete and more efficient.

AML is an open source system, and is available through GitHub ¹ both for developers (as an Eclipse project) and end-users (as a runnable Jar with a graphical user interface).

1 Background

Ontology matching is the task of finding correspondences (or mappings) between semantically related concepts of two ontologies, so as to generate an alignment that enables integration and interoperability between those ontologies [2]. It is a critical task to realize the vision of the Semantic Web, and is particularly relevant in the life sciences, given the abundance of biomedical ontologies with partially overlapping domains.

At its base, ontology matching is a problem of quadratic complexity as it entails comparing all concepts of one ontology with all concepts of the other. Early ontology matching systems were not overly concerned with scalability, as the matching problems they tackled were relatively small. But with the increasing interest in matching large (biomedical) ontologies, scalability became a critical aspect, and as a result, traditional all-versus-all ontology matching strategies are giving way to more efficient anchor-based strategies (which have linear time complexity).

¹<https://github.com/AgreementMakerLight>

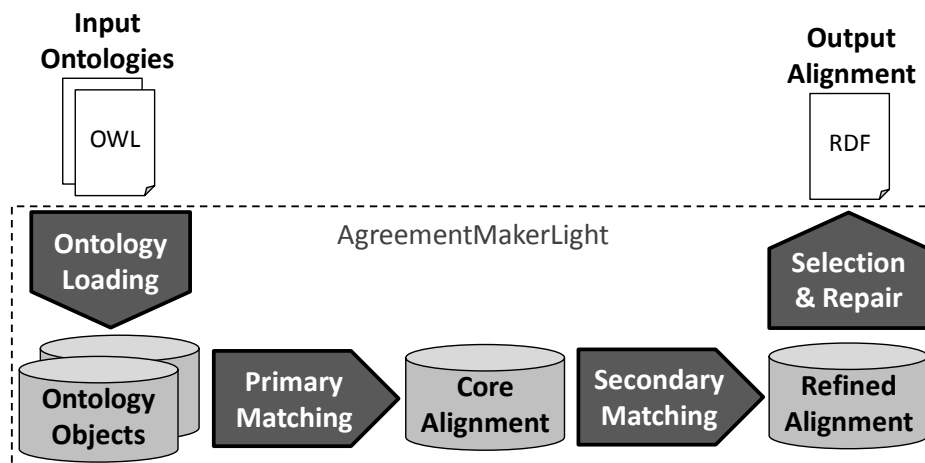


Fig. 1. AgreementMakerLight ontology matching framework.

2 The AgreementMakerLight System

AgreementMakerLight (AML) is a scalable automated ontology matching system developed to tackle large ontology matching problems, and focused in particular on the biomedical domain. It is derived from AgreementMaker, one of the leading first generation ontology matching systems [1], and adds scalability and efficiency to the design principles of flexibility and extensibility which characterized its namesake.

2.1 Ontology Matching Framework

The AML ontology matching framework is represented in Figure 1. It is divided into four main modules: ontology loading, primary matching, secondary matching, and alignment selection and repair

The ontology loading module is responsible for reading ontologies and parsing their information into the AML ontology data structures, which were conceived to enable anchor-based matching [5]. AML 2.0 marks the switch from the Jena2 ontology API to the more efficient and flexible OWL API, and includes several upgrades to the ontology data structures. The most important data structure AML uses for matching is the *Lexicon*, a table of class names and synonyms in an ontology, which uses a ranking system to weight them and score their matches [7].

The primary and secondary matching modules contain AML's ontology matching algorithms, or matchers, with the difference between them being their time complexity. Primary matchers have $O(n)$ time complexity and therefore can be

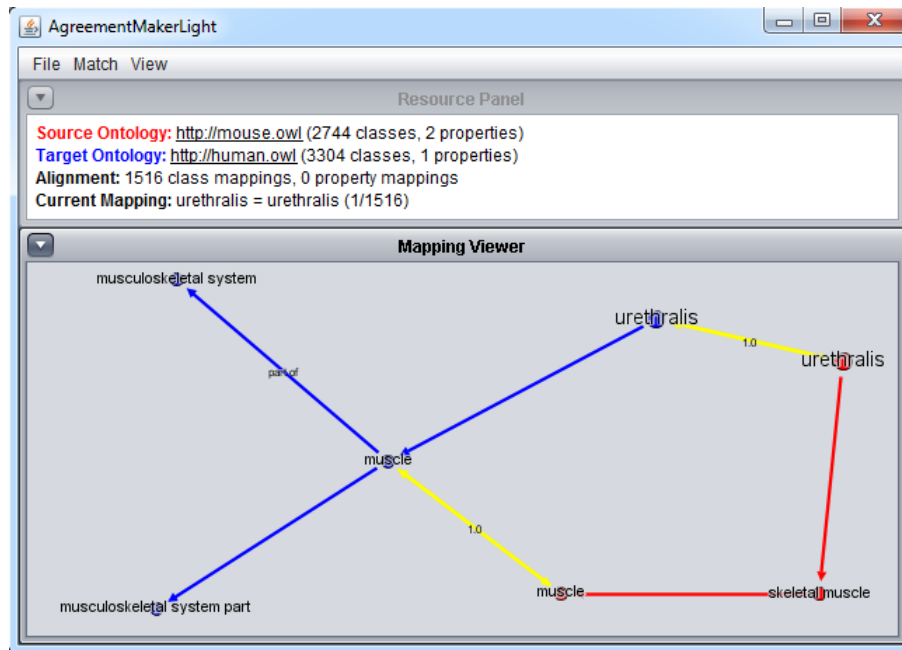


Fig. 2. AgreementMakerLight graphical user interface.

employed globally in all matching problems, whereas secondary matchers have $O(n^2)$ time complexity and thus can only be applied locally in large problems. The use of background knowledge in primary matchers is a key feature in AML, and it includes an innovative automated background knowledge selection algorithm [4].

The alignment selection and repair module ensures that the final alignment has the desired cardinality and that it is coherent (i.e., does not lead to the violation of restrictions of the ontologies) which is important for several applications. AML's approximate alignment repair algorithm features a modularization step which identifies the minimal set of classes that need to be analyzed for coherence, thus greatly reducing the scale of the repair problem [8].

2.2 User Interface

The GUI was a recent addition to AML, as we sought to make our system available to a wider range of users. The main challenge in designing the GUI was finding a way to visualize an alignment between ontologies that was both scalable and useful for the user. Our solution was to visualize only the neighborhood of one mapping at a time, while providing several options for navigating through the alignment [6]. The result is a simple and easy to use GUI which is shown in Figure 2.

3 Performance

AML 1.0 achieved top results in the 2013 edition of the Ontology Alignment Evaluation Initiative (OAEI) [3]. Namely, it ranked first in F-measure in the anatomy track, and second in the large biomedical ontologies, conference and interactive matching tracks. In addition to its effectiveness in matching life sciences ontologies, AML was amongst the fastest systems in all tracks, and more importantly, had consistently a high F-measure/run time ratio.

AML 2.0 is more effective than its predecessor (thanks to the improved handling of background knowledge, the richer data structures and the addition of new matching algorithms) without sacrificing efficiency, so we expect it to perform even better in this year's edition of the OAEI.

Acknowledgments

DF, CP, ES and FMC were funded by the Portuguese FCT through the SOMER project (PTDC/EIA-EIA/119119/2010) and the LASIGE Strategic Project (PEst-OE/EEI/UI0408/2014). The research of IFC was partially supported by NSF Awards CCF-1331800, IIS-1213013, IIS-1143926, and IIS-0812258 and by a UIC-IPCE Civic Engagement Research Fund Award.

References

1. I. F. Cruz, F. Palandri Antonelli, and C. Stroe. AgreementMaker: Efficient Matching for Large Real-World Schemas and Ontologies. *PVLDB*, 2(2):1586–1589, 2009.
2. J. Euzenat and P. Shvaiko. *Ontology Matching*. Springer-Verlag New York Inc, 2007.
3. D. Faria, C. Pesquita, E. Santos, I. F. Cruz, and F. M. Couto. AgreementMakerLight Results for OAEI 2013. In *ISWC International Workshop on Ontology Matching (OM)*, 2013.
4. D. Faria, C. Pesquita, E. Santos, I. F. Cruz, and F. M. Couto. Automatic Background Knowledge Selection for Matching Biomedical Ontologies. *PLoS One*, In Press, 2014.
5. D. Faria, C. Pesquita, E. Santos, M. Palmonari, I. F. Cruz, and F. M. Couto. The AgreementMakerLight Ontology Matching System. In *OTM Conferences*, volume 8185 of *LNCS*, pages 527–541, 2013.
6. C. Pesquita, D. Faria, E. Santos, and F. M. Couto. Towards visualizing the alignment of large biomedical ontologies. In *10th International Conference on Data Integration in the Life Sciences*, 2014.
7. C. Pesquita, D. Faria, C. Stroe, E. Santos, I. F. Cruz, and F. M. Couto. What's in a 'nym'? Synonyms in Biomedical Ontology Matching. In *The Semantic Web - ISWC 2013*, volume 8218 of *Lecture Notes in Computer Science*, pages 526–541. Springer Berlin Heidelberg, 2013.
8. E. Santos, D. Faria, C. Pesquita, and F. M. Couto. Ontology alignment repair through modularization and confidence-based heuristics. *CoRR*, arXiv:1307.5322, 2013.

Extracting Architectural Patterns from Web Data

Ujwal Gadiraju, Ricardo Kawase, and Stefan Dietze

L3S Research Center, Leibniz University Hannover, Germany
{gadiraju, kawase, dietze}@L3S.de

Abstract. Knowledge about the reception of architectural structures is crucial for architects or urban planners. Yet obtaining such information has been a challenging and costly activity. With the advent of the Web, a vast amount of structured and unstructured data describing architectural structures has become available publicly. This includes information about the perception and use of buildings (for instance, through social media), and structured information about the building's features and characteristics (for instance, through public Linked Data). In this paper, we present the first step towards the exploitation of structured data available in the Linked Open Data cloud, in order to determine well-perceived architectural patterns.

1 Introduction and Motivation

Urban planning and architecture encompass the requirement to assess the popularity or perception of built structures (and their evolution) over time. This aids in understanding the impact of a structure, identify needs for restructuring, or to draw conclusions useful for the entire field, for instance, about successful architectural patterns and features. Thus, information about how people think about a building that they use or see, or how they feel about it, could prove to be invaluable information for architects, urban planners, designers, building operators, and policy makers alike. For example, keeping track of the evolving feelings of people towards a building and its surroundings can help to ensure adequate maintenance and trigger retrofit scenarios where required. On the other hand, armed with prior knowledge of specific features that are well-perceived by the public, builders and designers can make better-informed design choices and predict the impact of building projects.

The Web contains structured information about particular building features, for example, size, architectural style, built date, etc. of certain buildings through public Linked Data. Here in particular, reference datasets such as Freebase¹ or DBpedia² offer useful structured data describing a wide range of architectural structures.

The perception of an architectural structure itself has historically been studied to be a combination of the aesthetic as well as functional aspects of the structure [3, 4]. The impact of such buildings of varying types on the built environment, as well as how these buildings are perceived, thus varies. For example, intuitively we can say that in

¹ <http://www.freebase.com/>

² <http://dbpedia.org/>

case of churches, the appearance plays a vital role in the emotions induced amongst people. However, in case of airports or railway stations, the functionality aspects such as the efficiency or the accessibility may play a more significant role. This suggests that the impact of particular *influence factors* differs significantly between different *building types*.

In this paper, we present our work regarding the alignment of *Influence Factors* with structured data. Firstly, we identified the influence factors for a predefined set of architectural structures. Secondly, we align these factors with structured data from DBpedia. This work serves as a first step towards semantic enhancement of the architectural domain, which can support semantic classification of architectural structures, semantic analysis, and ranking, amongst others.

2 Crowdsourcing Influential Factors and Ranking Buildings

Recent research works in the field of Neuroscience [1, 2], reliably suggest that neurophysiological correlates of building perception successfully reflect aspects of an architectural rule system that adjust the appropriateness of style and content. They show that people subconsciously rank buildings that they see, between the categories of either high-ranking ('sublime') or low-ranking ('low') buildings. However, what exactly makes a building likeable or prominent remains unanswered. *Size* could be an influential factor. At the same time, it is not sound to suggest that architects or builders should design and build only big structures. For instance, a small hall may invoke more sublime feelings while a huge kennel may not. This indicates that there are additional factors that influence building perception. In order to determine such factors, we employ Crowdsourcing.

An initial survey was conducted using LimeService³ with a primary focus on the expert community of architects, builders and designers in order to determine influential factors. The survey administered 32 questions spanning over the background of the participants and their feelings about certain buildings, of different types (*bridges, churches, skyscrapers, halls* and *airports*). We received 42 responses from the expert community. The important influential factors that surfaced from the responses of the survey are presented below.

For *bridges, churches, skyscrapers* and *halls*: history, surroundings, materials, size, personal experiences, and level of detail. For *airports*: Ease of access, efficiency, appearance, choice/availability, facilities, miscellaneous facilities and size.

Based on these influential factors we acquired perception scores of buildings on a Likert-scale, through crowdsourcing. By aggregating and normalizing these scores between **0** and **1**, we arrived at a ranked list of buildings of each type within our dataset.

3 Correlating Influential Factors with Relevant Structured Data

In order to determine patterns in the perception of well-received structures (as per the building rankings), we correlate the influential factors of buildings with concrete properties and values from DBpedia.

³ <http://www.limeservice.com/>

Table 1: DBpedia properties that are used to materialize corresponding Influence Factors.

Airports	Bridges	Churches	Halls	Skyscrapers
dbpedia-owl:runwaySurface,	dbprop:architect,	dbprop:architectureStyle,	dbpedia-owl:yearOfConstruction,	dbprop:startDate,
dbpedia-owl:runwayLength,	dbpedia-owl:constructionMaterial,	dbprop:consecrationYear,	dbprop:built,	dbprop:completionDate,
dbprop:cityServed,	dbprop:material,	dbprop:materials,	dbprop:area,	dbpedia-owl:architect,
dbpedia-owl:locatedInArea,	dbpedia-owl:length,	dbprop:domeHeightOuter,	dbprop:seatingCapacity,	dbpedia-owl:floorCount
dbprop:direction	dbpedia-owl:width,	dbprop:length,	dbpedia-owl:location	
	dbpedia-owl:mainspan	width,		
		dbprop:area,		
		dbpedia-owl:location,		
		dbprop:district		

Table 1 depicts some of the properties that are extracted from the DBpedia knowledge graph in order to correlate the influence factors corresponding to each structure with specific values.

By doing so, we can analyze the well-received patterns for architectural structures at a finer level of granularity, i.e., in terms of tangible properties. In order to extract relevant data from DBpedia for each structure in our dataset, we first collect a pool of properties that correspond to each of the influence factors as per the building type (see Table 1). In the next step, by traversing the DBpedia knowledge graph leading to each structure in our dataset, we try to extract corresponding values for each of the properties identified. The properties thus extracted semi-automatically, are limited to those available on DBpedia. In addition, it is important to note that not all structures of a particular type have the same properties available on DBpedia. Therefore, although all the identified values accurately correspond to the structure, the coverage itself is restricted to the data available on DBpedia (see Table 2).

Table 2: Coverage of properties related to ‘size’, extracted from DBpedia for different architectural structures in our dataset.

Airports	Bridges	Churches	Halls	Skyscrapers
runwayLength: 95%	length: 67.79%	architectureStyle: 36.69%	seatingCapacity: 65.67%	floorCount: 91%

4 Application and Conclusions

By correlating the influence factors to specific DBpedia properties, we can identify patterns for well-perceived architectural structures. In order to demonstrate how such observed patterns for architectural structures can be used, we choose the influence factor ‘size’ of the structure. Although, this approach can be directly extended to other influence factors and across different kinds of architectural structures, due to the limited space we restrict ourselves to showcasing this influence factor.

We observe that for each airport, we can extract indicators of size using the DBpedia property `dbpedia-owl:runwayLength`. Similarly, in case of bridges the influence factor ‘size’ can be represented using the DBpedia properties `dbpedia-owl:length`,

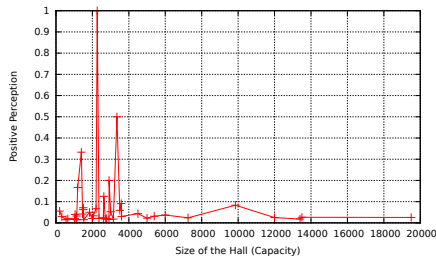


Fig. 1: Influence of Size in the perception of Halls.

dbpedia-owl:width and dbpedia-owl:mainspan, for halls we can use the DBpedia properties dbprop:area and dbprop:seatingCapacity, while we can use dbpedia-owl:floorCount, and dbprop:height to consolidate the well-perceived patterns for Skyscrapers. We thereby extract corresponding property values for each structure in our dataset⁴ using the DBpedia knowledge graph.

Figure 1 depicts the influence of *size* in the perception of halls. We observe that halls with a seating capacity between 1000-4000 people are well-perceived with the positive perception, varying between **0.1** and **1**. The perception scores are obtained through the aggregation of results from the crowdsourcing process. Similarly, as a result of the quantitative analysis of churches, by leveraging the rankings and correlating with the property dbpedia-owl:architecturalStyle, we find that the most well-received styles of churches in Germany are (i) *Gothic*, (ii) *Gothic Revival*, and (iii) *Romanesque*.

With this, we demonstrated that by correlating building characteristics with extracted data from DBpedia, one is able to compute and analyze architectural structures quantitatively. Thus, our main contribution includes semantic analysis and quantitative measurement of public perception of architectural structures based on structured data. As future work, we plan to develop algorithms that exploit properties from the structured data on the web in order to provide multi-dimensional architectural patterns like ‘skyscrapers with x size, y uniqueness, and z materials used are best perceived’, which architects and urban planners can benefit from.

References

1. I. Oppenheim, H. Mühlmann, G. Blechinger, I. W. Mothersill, P. Hilfiker, H. Jokeit, M. Kurthen, G. Krämer, and T. Grunwald. Brain electrical responses to high-and low-ranking buildings. *Clinical EEG and Neuroscience*, 40(3):157–161, 2009.
2. I. Oppenheim, M. Vannucci, H. Mühlmann, R. Gabriel, H. Jokeit, M. Kurthen, G. Krämer, and T. Grunwald. Hippocampal contributions to the processing of architectural ranking. *NeuroImage*, 50(2):742–752, 2010.
3. C. Sitte. *City planning according to artistic principles*. Rizzoli, 1986.
4. L. H. Sullivan. *The autobiography of an idea*, volume 281. Courier Dover Publications, 1956.

⁴ Our dataset and building rankings:
<http://data-observatory.org/building-perception/>

Xodx

A node for the Distributed Semantic Social Network

Natanael Arndt and Sebastian Tramp

Universität Leipzig, Institut für Informatik, AKSW,
Postfach 100920, D-04009 Leipzig, Germany
{arndt|tramp}@informatik.uni-leipzig.de

1 Introduction

The world wide web (WWW) is not anymore just an information retrieval system [1] but rather an interactive communication medium. Within the last decade online social networks have evolved and constantly increased in popularity. The currently most used online social network services, according to their estimated monthly active users¹, are Facebook (1.27 billion, facebook.com), Google Plus (541 million, plus.google.com) and Twitter (283 million, twitter.com). Compared to the estimated total users of the WWW of 2.93 billion, over 40% of the users of the WWW are actively using Facebook. This concentration on some single services contradicts the actual organisation of the WWW and the whole Internet as a network of decentrally organised and interconnected computer nodes. This situation bears risks regarding the privacy, data security, data ownership, reliability of the services and freedom of communication. By building up a distributed online social network with multiple interconnected services this risks can be minimised and a much more flexibly expendable network is created.

We present *Xodx* (<http://aksw.org/Projects/Xodx>, includes a live demo) an implementation of a node for the *Distributed Semantic Social Network* (DSSN). The DSSN is a general architecture for building an online social network using Semantic Web standards and additional protocols for real-time communication. Xodx provides functionality for publishing and editing personal profiles, adding friends to the friend list, sending and receiving friendship requests, publishing posts and following other users activities across distributed nodes.

2 Node Intercommunication and Integration in the Web

The complete architecture for the DSSN is proposed in „An Architecture of a Distributed Semantic Social Network“ [2]. It combines established Web 2.0 technologies i.e. (Semantic) Pingback [3] (Ping) and Activity-Streams published through PubSubHubbub (PuSH)² with an RDF data model and the Linked Data protocol [4]. The consequent use of RDF facilitates the integration of heterogeneous data in the data model. By using the Linked Data protocol the DSSN is

¹ as estimated on Internet Live Stats: internetlivestats.com, July 13th 2014

² PubSubHubbub: <https://code.google.com/p/pubsubhubbub/>

easily integrated with any other Semantic Web application and thus facilitates an extendable infrastructure. This enables us to build up a completely distributed network of data and services, which is highly integrated and embedded in the Web of Data and WWW.

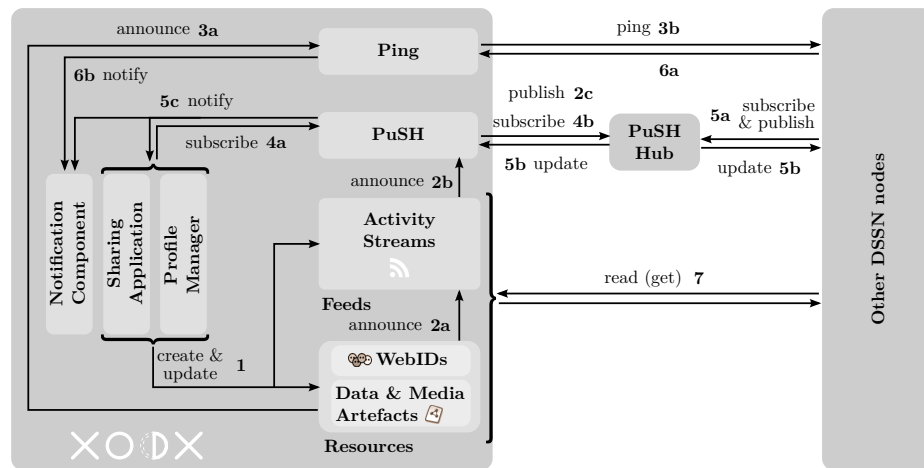


Figure 1. Architecture of an Xodx node and intercommunication between nodes on the DSSN. [5]

The architecture of the Xodx implementation and the intercommunication with other nodes is depicted in fig. 1. The profile manager is used for editing the WebID and adding new friendship relations, it creates and updates the according resources and activity streams (1). Similarly, the sharing application can be used for sharing media artefacts and web resource. Updates of the WebID or any other data or media artefact are announced (2a) by updating the activity stream, which is then announced to the PuSH service (2b), which in turn publishes the updates to the PuSH hub (2c). In parallel this change is announced to the Ping component (3a), which sends a ping to each resource which is mentioned in the update (3b). With the PuSH service a user can be subscribed to any activity stream on the DSSN (4a, 4b), which implements a follow functionality. If the PuSH hub receives a new update announcement from any DSSN node (publish, 5a) it will update all nodes, which are subscribed to the according resource stream (5b). The PuSH service of the Xodx node will then notify the according components (5c). When the ping service receives a new ping (6a) it will call the notification component to generate a new user notification (6b). If other nodes (or any Linked Data application) browses the DSSN the activity streams, data & media artefacts and any RDF resource can be retrieved via standard HTTP-GET-requests (7) according to the Linked Data principle.

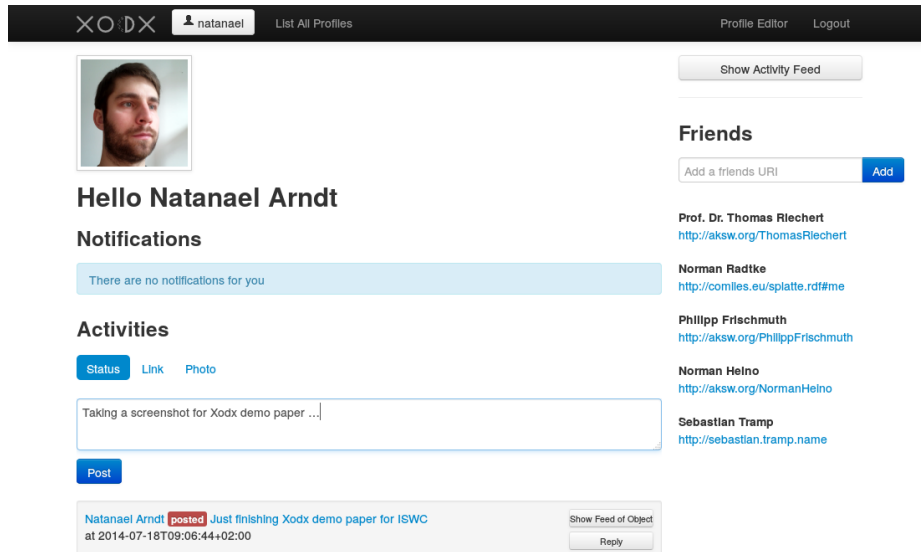


Figure 2. Screenshot and feature demonstration of Xodx

Figure 2 shows the home-screen of a logged-in user, it is organised in three main parts, the top navigation bar, the button and friend-list bar to the right and the main part with activity stream in the middle. All other resource-views are organised very similar to the user's home-screen. The top navigation bar from left to right has a home button for the user, which also indicates if new notifications are available. A button to navigate to the list of profiles, available at the current node. The profile editor enables the user to edit any triple of her WebID. The right most button is for log-out.

On the right hand side one can view the Activity Feed of the current displayed resource by clicking on “Show Activity Feed”. If the profile of a person different from the current user is displayed, an additional button “Add {name} as Friend” is displayed, to add a friendship relation to the own profile, send a ping to the respective person for notification and subscribe to his activity stream for updates. Below the buttons one can find the friend list. It contains a field for adding a new friend using its WebID-URI and a list of existing friendship relations. By clicking on the individual entries the user can browse the profiles and further navigate to the friends of their friends. If the browser reaches a resource, which is not yet available in the local triple store, it is retrieved according to the Liked Data principles.

The central part shows the picture and the name of the current user, current notifications and the sharing application with activity stream. If a new friend request is send to the node or if the user was mentioned in a post, a new notification is generated and displayed in the notification section to inform the user. Using the sharing application a user can create and share new posts, web re-

sources or photos with his subscribers via PuSH. The activity stream displayed at the very bottom is a combined stream of the personal stream of the user and all activity streams she is subscribed to. With the two buttons to the right of the activity entry a user can generate a reply resource answering to this activity resource and view the activity feed of this resource, which includes all its replies.

3 Prospect and Future Work

The Xodx implementation demonstrates the feasibility of a semantic social network build by distributed nodes. It already supports the main features of building friendship relations and sharing resources across the network. Currently some further features for supporting the work in groups are planned. The practical tests with multiple nodes already pointed out the usage of the PubSubHubbub protocol as a limiting factor. Currently only a few hub implementations are available and effectively the google reference implementation and instance is the only usable hub. So we are trying to substitute the federation protocol with a Linked Data and Semantic Pingback protocol. But due to the Linked Data nature of the DSSN we have seen, that it is easy to integrate this social network architecture with any application or data-set on the Web of Data. Possible applications, which can benefit from the semantic integration with the social network are e.g. collaborative-wikis, web-logs or personal information management systems. This concept will give us new opportunities for integrating social functionality in any web application and thus extend the social web to the whole Internet rather than some big closed nodes.

References

1. Berners-Lee, T., Cailliau, R., Groff, J.F., Pollermann, B.: World-Wide Web: The Information Universe. *Electronic Networking: Research, Applications and Policy* **2**(1) (1992) 52–58
2. Tramp, S., Frischmuth, P., Ermilov, T., Shekarpour, S., Auer, S.: An Architecture of a Distributed Semantic Social Network. *Semantic Web Journal **Special Issue on The Personal and Social Semantic Web*** (2012)
3. Tramp, S., Frischmuth, P., Ermilov, T., Auer, S.: Weaving a Social Data Web with Semantic Pingback. In Cimiano, P., Pinto, H., eds.: *Proceedings of the EKAW 2010 - Knowledge Engineering and Knowledge Management by the Masses; 11th October-15th October 2010 - Lisbon, Portugal*. Volume 6317 of *Lecture Notes in Artificial Intelligence (LNAI)*., Berlin / Heidelberg, Springer (October 2010) 135–149
4. Berners-Lee, T.: Linked Data. Design issues, W3C (June 2009) <http://www.w3.org/DesignIssues/LinkedData.html>.
5. Arndt, N.: Xodx – Konzeption und Implementierung eines Distributed Semantic Social Network Knotens. Master's thesis, Universität Leipzig, Fakultät für Mathematik und Informatik, Institut für Informatik (June 2013)

An Ontology Explorer for Biomimetics Database

Kouji KOZAKI¹ and Riichiro MIZOGUCHI²

¹The Institute of Scientific and Industrial Research, Osaka University
8-1 Mihogaoka, Ibaraki, Osaka, 567-0047 Japan
kozaki@ei.sanken.osaka-u.ac.jp

²Japan Advanced Institute of Science and Technology
1-1 Asahidai, Nomi, Ishikawa 923-1292, Japan
mizo@jaist.ac.jp

Abstract. Biomimetics contributes to innovative engineering by imitating the models, systems, and elements of nature. For biomimetics research, it is important to develop biomimetics database including widely varied knowledge across different domains such as biology and engineering. Interoperability of knowledge among those domains is necessary to create such a database. For this purpose, the authors are developing a biomimetics ontology which bridge gaps between biology and engineering. In this demo, the authors shows an ontology exploration tool for biomimetics database. It is based on linked data techniques and allows the users to find important keywords so that they can search meaningful knowledge from various databases.

Keywords: ontology, linked data, biomimetics, database, semantic search

1 Introduction

Learning from nature aids development of technologies. Awareness of this fact has been increasing, and biomimetics¹ [1], innovative engineering through imitation of the models, systems, and elements of nature, has caught the attention of many people. Well-known examples of biomimetics include, paint and cleaning technologies that imitate the water repellency of the lotus, adhesive tapes that imitate the adhesiveness of gecko feet, and high-speed swimsuits that imitate the low resistance of a shark's skin. These results integrate studies on the biological mechanisms of organisms with engineering technologies to develop new materials. Facilitating such biomimetics-based innovations requires integrating knowledge, data, requirements, and viewpoints across different domains. Researchers and engineers need to develop a biomimetics database to assist them in achieving this goal.

Because ontologies clarify concepts that appear in target domains [2], we assume that it is important to develop a biomimetics ontology that contributes to improvement of knowledge interoperability between the biology and engineering domains. Furthermore, linked data technologies are very effective for integrating a database with existing biological diversity databases. On the basis of these observations, we developed a

¹ <http://www.cbid.gatech.edu/>

biomimetics ontology and ontology exploration system based on linked data techniques. The tool allows users to find important keywords for retrieving meaningful knowledge from viewpoints of biomimetics through various databases. This demo shows how the ontology explorer for biomimetics database works on the web.

2 A Biomimetics Ontology

Before we began developing a biomimetics ontology, we conducted interviews with engineers working with biomimetics regarding their requirements for biomimetics database search. When we asked, “What do you want to search for in a biomimetic database?” they said they wanted to search for organisms or organs that perform functions that they were trying to develop in their new products. In fact, most successful examples are imitations of capabilities that organisms possess, such as the water repellency of a lotus and the adhesiveness of a gecko’s feet. Therefore, we proposed that it is important to search the biomimetic database for functions or goals that they want to achieve.

On the other hand, someone engaged in cooperative research with engineers and biologists reported that engineers do not have knowledge that is very familiar to biologists. For instance, when an engineer had a question about functions of projections shown in an electron microscopy image of an insect, a biologist (entomologist) suggested that it could have an anti-slip capability, because the insect often clings to slippery surfaces. This suggests that a biomimetic ontology must bridge knowledge gaps between engineers and biologists.

Considering the requirements discussed in the above, we set the first requirement for biomimetics ontology as to be able to search for related organisms by the function the user wants to perform. At the same time, we propose that it should support various viewpoints to bridge gaps among domains. As a result, we built a biomimetics ontology that includes 379 concepts (classes) and 314 relationships (properties), except for the *is-a* (*sub-class-of*) relation. For example, *Organism* may have relationships such as *Ecological environment*, *Characteristic behavior*, *Characteristic structure*, *Characteristic function*, *Region Part*, and *Goal* may have relationships such as *Structure on which to base* and *Related function*. Other top level concepts includes *Behavior*, *Act*, *Function*, *Process*, *Structure*, *Living environment*, and so on.

3 An Ontology Explorer for Biomimetics Database

We developed the ontology explorer for biomimetics database based on an ontology exploration techniques proposed in our previous work [3]. The framework enables users to freely explore a sea of concepts in the ontology from a variety of perspectives according to their own motives. Exploration stimulates their way of thinking and contributes to deeper understanding of the ontology and hence its target world. As a result, users can discover what interests them. This could include new findings that are new to them, because they might find unexpected conceptual chains from the ontology exploration that they would otherwise never have thought of.

Exploration of an ontology can be performed by choosing arbitrary concepts from which *multi-perspective conceptual chains* can be traced, according to the explorer’s

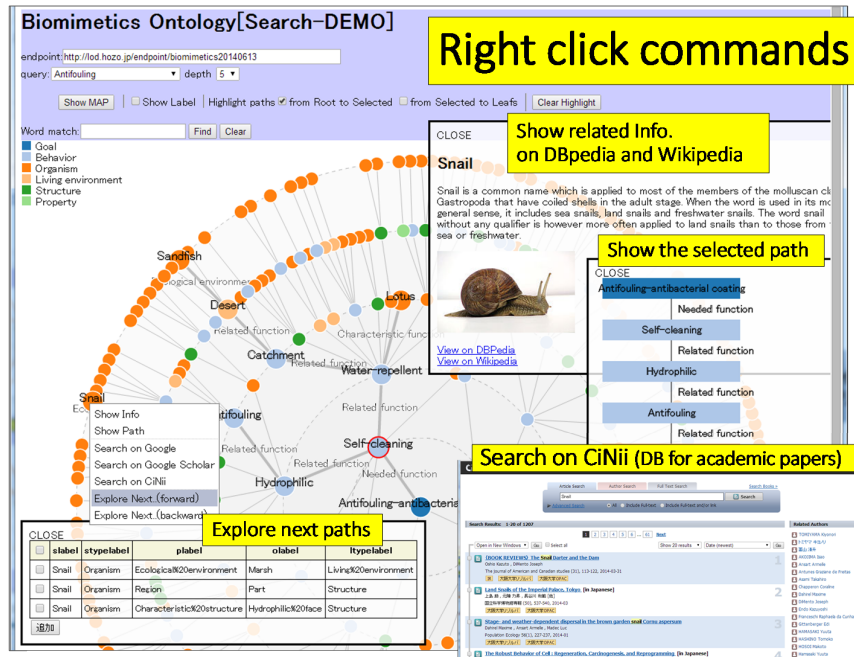


Fig.1 A snapshot of the Ontology Explorer for Biomimetics Database.

intention. We define the viewpoint for exploring an ontology and obtaining multi-perspective conceptual chains as the combination of a *focal point* and *aspects*. A focal point indicates a concept to which the user pays attention as a starting point of the exploration. The aspect is the manner in which the user explores the ontology. Because an ontology consists of concepts and the relationships among them, the aspect can be represented by a set of methods for extracting concepts according to its relationships. The multi-perspective conceptual chains are visualized in a user-friendly form, i.e., in a conceptual map. Based on these techniques, we developed the ontology explorer for retrieving information from biomimetics database as a web application to assist the user in using the results easily for searching other databases, while the previously described system was developed as a Java client application. We implemented the ontology exploration tool using HTML5 and Java Script to enable it to work on web browsers on many platforms, including not only PCs but also tablets and smartphones. We implemented the exploration methods based on Simple Protocol and RDF Query Language (SPARQL) queries.

Fig.1 shows one result of ontology exploration using the system. In this example, the user selected *Antifouling* as the focal point (starting point) and obtained conceptual chains to some *Organism* as the end point. In this case, the system searches all combination of aspects (relationships) to generate conceptual chains from a concept selected as starting point to those specified by the user. As a result, the system shows all conceptual chains between the selected concepts as a conceptual map. By clicking the nodes on the map, the user can detailed information about each paths. Furthermore, the user can use the selected information to search other Linked Data such as DBpedia and

databases. Though the current version supports only a few LODs and databases, it can be easily extended to others.

4 Conclusion and Future work

This article outlined an ontology explorer for biomimetics database. Since the current version of the system is a prototype, it uses only a small ontology and has limits on the conditions of exploration. However, it was well received by researchers on biomimetics. In fact, one of them said that the resulting path from *Antifouling* to *Sandfish* shown in Fig.1 was unexpected one for him. This suggests that the proposed system could contribute innovations in biomimetics. The researchers also plan to use the biomimetics ontology and system as an interactive index for a biomimetics textbook.

Future work includes extensions of the biomimetics ontology and the exploration system. For the former, we plan to use documents on biomimetics and existing linked data related to biology and considering some methods for semi-automatic ontology building using them. For later, we are exploring potentially useful patterns through discussion with biomimetics researchers and ontology engineers.

There are many approaches to Semantic Search using SPARQL. For example, Ferré proposes QFS (Query-based Faceted Search) for support in navigating faceted search using LISQL (Logical Information System Query Language) [4] and implement it based on SPARQL endpoints to scale to large datasets [5]. Popov proposes an exploratory search called Multi-Pivot [6] which extracts concepts and relationships from ontologies according to a user's interest. These are visualized and used for semantic searches among instances (data). The authors took the same approach as Popov. Considering how to use these techniques in our system is an important future work.

The current version of the proposed system is available at the URL;
http://biomimetics.hozo.jp/ontology_db.html.

Acknowledgements

This work was supported by JSPS KAKENHI Grant Number 25280081 and 24120002.

References

1. Shimomura, M.: Engineering Biomimetics: Integration of Biology and Nanotechnology, Design for Innovative Value Towards a Sustainable Society, 905-907 (2012)
2. Gruber, T.: A translation approach to portable ontologies specifications, Proc. of JKAW'92, pp.89-108 (1992)
3. Kozaki K, Hirota T and Mizoguchi R, Understanding an Ontology through Divergent Exploration, Proc. of ESWC 2011, Part I, LNCS 6643, pp.305-320 (2011).
4. Ferré, S., Hermann, A.: Reconciling faceted search and query languages for the semantic web. IJMSO 7(1), 37-54 (2012)
5. Guyonvarch, J., Ferré S.: Scalewelis: a scalable query-based faceted search elenawork. Multilingual Question Answering over Linked Data (QALD-3), Valencia, Spain
6. Popov, I. O., m.c. schraefel, Hall, W., Shadbolt, N.: Connecting the dots: A multi-pivot approach to data exploration. International Semantic Web Conference (ISWC2011), LNCS7031, 553-568 (2011)

Semi-Automated Semantic Annotation of the Biomedical Literature

Fabio Rinaldi

Institute of Computational Linguistics, University of Zurich
fabio.rinaldi@uzh.ch

Abstract. Semantic annotations are a core enabler for efficient retrieval of relevant information in the life sciences as well in other disciplines. The biomedical literature is a major source of knowledge, which however is underutilized due to the lack of rich annotations that would allow automated knowledge discovery.

We briefly describe the results of the SASEBio project (Semi Automated Semantic Enrichment of the Biomedical Literature) which aims at adding semantic annotations to PubMed abstracts, in order to present a richer view of the existing literature.

1 Introduction

The scientific literature contains a wealth of knowledge which however cannot be easily used automatically due to its unstructured nature. In the life sciences, the problem is so acutely felt that large budgets are invested into the process of literature curation, which aims at the construction of structured databases using information mostly manually extracted from the literature. There are several dozens of life science databases, each specializing on a particular subdomain of biology. Examples of well-known biomedical databases are UniProt (proteins), EntrezGene (genes), NCBI Taxonomy (species), IntAct (protein interactions), BioGrid (protein and genetic interactions), PharmGKB (drug-gene-disease relations), CTD (chemical-gene-disease relations), and RegulonDB (regulatory interactions in *E. coli*).

The OntoGene group¹ aims at developing text mining technologies to support the process of literature curation, and promote a move towards *assisted curation*. By assisted curation we mean a combination of text mining approaches and the work of an expert curator, aimed at leveraging the power of text mining systems, while retaining the high quality associated with human expertise. We believe that it is possible to gradually automate much of the most repetitive activities of the curation process, and therefore free up the creative resources of the curators for more challenging tasks, in order to enable a much more efficient and comprehensive curation process. Our text mining system specializes in the detection of entities and relationships from selected categories, such as proteins, genes, drugs, diseases, chemicals. OntoGene derives some of its resources from life sciences databases, thus allowing a deeper connection between the unstructured information contained in the literature and the structured information contained in databases. The quality of the system has been tested several times through participation in some of the community-organized evaluation campaigns, where it often obtained top-ranked results. We have also implemented a platform for assisted curation called ODIN (OntoGene Document INSpector) which aims at serving the needs of the curation community. The usage of ODIN as a tool for assisted curation has been tested within the scope of collaborations with curation groups, including PharmGKB [7], CTD [8], RegulonDB [5].

Assisted curation is also of utility in the process of pharmaceutical drug discovery. Many text mining tasks in drug discovery require both high precision and high recall, due to the importance of comprehensiveness and quality of the output. Text mining algorithms, however, cannot often achieve both high precision and high recall, sacrificing one for the other. Assisted curation can be paired with text mining algorithms which have high recall and moderate precision to produce results that are amenable to answer pharmaceutical problems with only a reasonable effort being allocated to curation.

¹ <http://www.ontogene.org/>

Methods

The Ontogene system is based on a pipeline architecture (see figure 1), which includes, among others, modules for entity recognition and relation extraction. Some of the modules are rule-based (e.g. lexical lookup with variants) while others use machine-learning approaches (e.g. maximum entropy techniques). The initial step consists in the annotation of names of relevant domain entities in biomedical literature (currently the system considers proteins, genes, species, experimental methods, cell lines, chemicals, drugs and diseases). These names are sourced from reference databases and are associated with their unique identifiers in those databases, thus allowing resolution of synonyms and cross-linking among different resources.

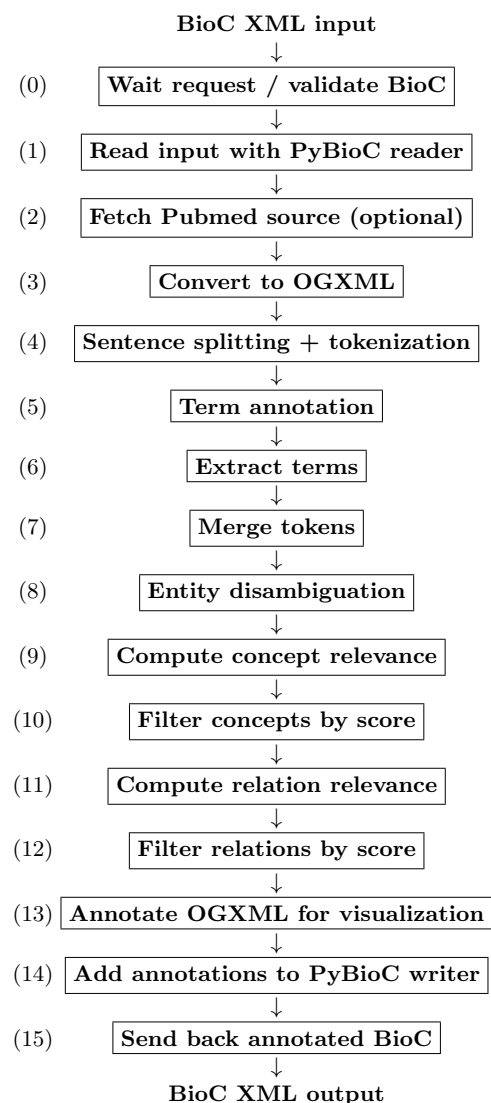


Fig. 1. Schema of the OntoGene pipeline

One of the problems with sourcing resources from several databases is the possible inconsistencies among them. The fact that domain knowledge is scattered across dozens of data sources, occasionally also with some incompatibilities among them, is a severe problem in the life sciences. Ideally these resources should be integrated in a single repository, as some projects are attempting to do (e.g. OpenPhacts [16]), allowing querying within an unified platform. However, a deep integration of the information provided by the scientific literature and the content of the databases is still missing.

We train our system using the knowledge provided by life sciences databases as our gold standard, instead of hand-labeled corpora, since we believe that the scope and size of manually annotated corpora, however much effort has been invested in creating them, is not sufficient to capture the wide variety of linguistic phenomena that can be encountered in the full corpus of biomedical literature, let alone other types of documents, such as internal scientific reports in the pharma industry, which are not represented at all in annotated corpora. For example, PubMed currently contains more than 23 million records, while the entire set of all annotated publications probably barely reaches a few thousands, most of them sparsely annotated for very specific purposes.

We generate interaction candidates using co-occurrence of entities within selected syntactic units (typically sentences). An additional step of syntactic parsing using a state-of-the-art dependency parser allows us to derive specialized features in order to increase precision. The details of the algorithm are presented in [14]. The information delivered by the syntactic analysis is used as a factor in order to score and filter candidate interactions based on the syntactic fragment which connects the two participating entities. All available lexical and syntactic information is used in order to provide an optimized ranking for candidate interactions. The ranking of relation candidates is further optimized by a supervised machine learning method described in detail in [2].

Results

The OntoGene annotator offers an open architecture allowing for a considerable level of customization so that it is possible to plug in in-house terminologies. We additionally provide access to some of our text mining services through a RESTful interface.² Users can submit arbitrary documents to the OntoGene mining service by embedding the text to be mined within a simple XML wrapper. Both input and output of the system are defined according to the BioC standard [4]. However, typical usage will involve processing of PubMed abstracts or PubMed Central full papers. In this case, the user can provide as input simply the PubMed identifier of the article. Optionally the user can specify which type of output they would like to obtain: if entities, which entity types, and if relationships, which combination of types.

The OntoGene pipeline identifies all relevant entities mentioned in the paper, and their interactions, and reports them back to the user as a ranked list, where the ranking criteria is the system's own confidence for the specific result. The confidence value is computed taking into account several factors, including the relative frequency of the term in the article, its general frequency in PubMed, the context in which the term is mentioned, and the syntactic configuration among two interacting entities (for relationships). A detailed description of the factors that contribute to the computation of the confidence score can be found in [14].

The user can choose to either inspect the results, using the ODIN web interface, or to have them delivered back via the RESTful web service in BioC XML format, for further local processing. ODIN (OntoGene Document Inspector) is a flexible browser-based client application which interfaces with the OntoGene server. The curator can use the features provided by ODIN to visualize selected annotations, together with the statements from which they were derived, and, if necessary, add, remove or modify them. Once the curator has validated a set of candidate annotations, they can be exported, using a standard format (e.g. CSV, RDF), for further processing by other tools, or for inclusion in a reference database, after a suitable format conversion. In case of ambiguity, the curator is offered the opportunity to correct the choices made by the system, at any of the different levels of processing: entity identification and disambiguation, organism selection, interaction candidates. The curator can access all the possible readings given by the system and select the most accurate.

As a way to verify the quality of the core text mining functionalities of the OntoGene system, we have participated in a number of text mining evaluation campaigns [9, 3, 12, 13]. Some of the most interesting results include best results in the detection of protein-protein interactions in BioCreative 2009 [14], top-ranked results in several tasks of BioCreative 2010 [15], best results in the triage task of BioCreative 2012 [9]. The usage of ODIN as a curation tool has been tested in a few collaborations with curation groups, including PharmGKB [10], CTD [7], RegulonDB [11]. Assisted curation is also one of the topics being evaluated at the BioCreative competitions [1], where OntoGene/ODIN participated with favorable results. The effectiveness of the web service has been recently evaluated within the scope of one of the BioCreative 2013 shared tasks [6]. Different implementations can rapidly be produced upon request.

Since internally the original database identifiers are used to represent the entities and interactions detected by the system, the annotations can be easily converted into a semantic web format, by using a reference URI for each domain entity, and using RDF statements to express interactions. While it is possible to access the automatically generated annotations for further processing by a reasoner or integrator tool, we strongly believe that at present a process of semi-automated validation is preferable and would lead to better data consistency.

Acknowledgments. The OntoGene group is partially supported by the Swiss National Science Foundation (grant 105315 – 130558/1 to Fabio Rinaldi) and by the Data Science Group at Hoffmann-La Roche, Basel, Switzerland.

² <http://www.ontogene.org/webservices/>

References

1. Arighi, C., Roberts, P., Agarwal, S., Bhattacharya, S., Cesareni, G., Chatr-aryamontri, A., Clematide, S., Gaudet, P., Giglio, M., Harrow, I., Huala, E., Krallinger, M., Leser, U., Li, D., Liu, F., Lu, Z., Maltais, L., Okazaki, N., Perfetto, L., Rinaldi, F., Saetre, R., Salgado, D., Srinivasan, P., Thomas, P., Toldo, L., Hirschman, L., Wu, C.: Biocreative iii interactive task: an overview. *BMC Bioinformatics* 12(Suppl 8), S4 (2011), <http://www.biomedcentral.com/1471-2105/12/S8/S4>
2. Clematide, S., Rinaldi, F.: Ranking relations between diseases, drugs and genes for a curation task. *Journal of Biomedical Semantics* 3(Suppl 3), S5 (2012), <http://www.jbiomedsem.com/content/3/S3/S5>
3. Clematide, S., Rinaldi, F., Schneider, G.: Ontogene at calbc ii and some thoughts on the need of document-wide harmonization. In: *Proceedings of the CALBC II workshop*, EBI, Cambridge, UK, 16-18 March (2011)
4. Comeau, D.C., Doan, R.I., Ciccarese, P., Cohen, K.B., Krallinger, M., Leitner, F., Lu, Z., Peng, Y., Rinaldi, F., Torii, M., Valencia, A., Verspoor, K., Wieggers, T.C., Wu, C.H., Wilbur, W.J.: BIOC: a minimalist approach to interoperability for biomedical text processing. *The Journal of Biological Databases and Curation* bat064 (2013), published online
5. Gama-Castro, S., Rinaldi, F., Lpez-Fuentes, A., Balderas-Martnez, Y.I., Clematide, S., Ellendorff, T.R., Collado-Vides, J.: Assisted curation of growth conditions that affect gene expression in *e. coli* k-12. In: *Proceedings of the Fourth BioCreative Challenge Evaluation Workshop*. vol. 1, pp. 214-218 (2013)
6. Rinaldi, F., Clematide, S., Ellendorff, T.R., Marques, H.: OntoGene: CTD entity and action term recognition. In: *Proceedings of the Fourth BioCreative Challenge Evaluation Workshop*. vol. 1, pp. 90-94 (2013)
7. Rinaldi, F., Clematide, S., Garten, Y., Whirl-Carrillo, M., Gong, L., Hebert, J.M., Sangkuhl, K., Thorn, C.F., Klein, T.E., Altman, R.B.: Using ODIN for a PharmGKB re-validation experiment. *Database: The Journal of Biological Databases and Curation* (2012)
8. Rinaldi, F., Clematide, S., Hafner, S.: Ranking of ctd articles and interactions using the ontogene pipeline. In: *Proceedings of the 2012 BioCreative workshop*. Washington D.C. (April 2012)
9. Rinaldi, F., Clematide, S., Hafner, S., Schneider, G., Grigonyte, G., Romacker, M., Vachon, T.: Using the OntoGene pipeline for the triage task of BioCreative 2012. *The Journal of Biological Databases and Curation*, Oxford Journals (2013)
10. Rinaldi, F., Clematide, S., Schneider, G., Romacker, M., Vachon, T.: ODIN: An advanced interface for the curation of biomedical literature. In: *Biocuration 2010, the Conference of the International Society for Biocuration and the 4th International Biocuration Conference*. p. 61 (2010), available from Nature Precedings <http://dx.doi.org/10.1038/npre.2010.5169.1>
11. Rinaldi, F., Gama-Castro, S., Lpez-Fuentes, A., Balderas-Martnez, Y., Collado-Vides, J.: Digital curation experiments for regulondb. In: *BioCuration 2013*, April 10th, Cambridge, UK (2013)
12. Rinaldi, F., Kappeler, T., Kaljurand, K., Schneider, G., Klenner, M., Clematide, S., Hess, M., von Allmen, J.M., Parisot, P., Romacker, M., Vachon, T.: OntoGene in BioCreative II. *Genome Biology* 9(Suppl 2), S13 (2008), <http://genomebiology.com/2008/9/S2/S13>
13. Rinaldi, F., Kappeler, T., Kaljurand, K., Schneider, G., Klenner, M., Hess, M., von Allmen, J.M., Romacker, M., Vachon, T.: OntoGene in Biocreative II. In: *Proceedings of the II Biocreative Workshop* (2007)
14. Rinaldi, F., Schneider, G., Kaljurand, K., Clematide, S., Vachon, T., Romacker, M.: OntoGene in BioCreative II.5. *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 7(3), 472-480 (2010)
15. Schneider, G., Clematide, S., Rinaldi, F.: Detection of interaction articles and experimental methods in biomedical literature. *BMC Bioinformatics* 12(Suppl 8), S13 (2011), <http://www.biomedcentral.com/1471-2105/12/S8/S13>
16. Williams, A.J., Harland, L., Groth, P., Pettifer, S., Chichester, C., Willighagen, E.L., Evelo, C.T., Blomberg, N., Ecker, G., Goble, C., Mons, B.: Open phacts: semantic interoperability for drug discovery. *Drug Discovery Today* 17(2122), 1188 - 1198 (2012), <http://www.sciencedirect.com/science/article/pii/S1359644612001936>

Live SPARQL Auto-Completion

Stéphane Campinas

Insight Centre for Data Analytics, National University of Ireland, Galway
stephane.campinas@insight-centre.org

Abstract. The amount of Linked Data has been growing increasingly. However, the efficient use of that knowledge is hindered by the lack of information about the data structure. This is reflected by the difficulty of writing SPARQL queries. In order to improve the user experience, we propose an auto-completion library¹ for SPARQL that suggests possible RDF terms. In this work, we investigate the feasibility of providing recommendations by only querying the SPARQL endpoint directly.

1 Introduction

The Linking Open Data movement has brought a tremendous amount of data available to the general user. The available knowledge spans a wide range of domains, from life sciences to films. However, using SPARQL to search through this knowledge is a tedious process, not only because of the syntax barrier but mainly due to the schema heterogeneity of the data. The expression of an information need in SPARQL is difficult due to the schema being generally unknown to the user as well as an heterogeneous of several vocabularies.

A common solution is for the user to manually gain knowledge about the data structure, i.e., what predicates and classes are used, by executing additional queries in parallel to the main one. The paper [3] proposes a “context-aware” *auto-completion* method for assisting a user in writing a SPARQL query by recommending schema terms in various position in the query. The method is context-aware in the sense that only essential triple patterns are considered for the recommendations. To do so, it leverage a data-generated schema. Instead, in this work we propose to bypass this need by executing live SPARQL queries in order to provide recommendations. Thus, this removes the overhead of pre-computing the data-generated schema. The proposed approach exposes a trade-off between the performance of the application and the quality of the recommendations. We make available a library¹ for providing data-based recommendations that can be used with other tools such as YASGUI [8].

In Section 2 we discuss related works regarding auto-completion for SPARQL. In Section 3 we present the proposed approach. In Section 4 we report an evaluation of the system based on query logs of DBpedia.

2 Related Work

Over the years, many contributions have been done towards facilitating the use of SPARQL, either visually [4], or by completely hiding SPARQL from the user [7]. In this work, we aim to help users with a knowledge of SPARQL by providing an auto-completion feature. Several systems have been proposed in this direction. Although

¹ Gosparqled: <https://github.com/scampi/gosparqled>

the focus in [1] is the visual interface, it can provide recommendations of terms such as predicates and classes. In [6] possible recommendations are taken from query logs. The system proposed in [5] provides recommendations based on the data itself, with a focus on SPARQL federation. Instead, we aim to make available an easy-to-use library which core feature is to provide data-based recommendations. In [3] an editor with auto-completion was developed that leverage a data-generated schema (i.e., a *graph summary*). We investigate in this work the practicability of bypassing the graph summary by relying only on the data.

3 Live Auto-Completion

We propose a data-based auto-completion which retrieves possible items with regards to the current state of the query. Recommended items can be predicates, classes, or even named graphs. Firstly, we indicate the position in the SPARQL query that is to be auto-completed, i.e., the *Point Of Focus* (POF), by inserting the character ‘<’. Secondly, we reduce the query down to its *recommendation scope* [3]. Finally, we transform the POF into the SPARQL variable “?POF” which is used for retrieving recommendations. The retrieved recommendations are then ranked, e.g., by the number of occurrences of an item.

Recommendation Scope. While building a SPARQL query, not all triple patterns are relevant for the recommendation. Therefore, we define the scope as the connected component that contains the POF. Figure 1a depicts a SPARQL query where the POF is associated with the variable “?s”: it seeks possible predicates that occur with a “:Person” having the predicate “:name”. Figure 1b depicts the previous SPARQL query reduced to its recommendation scope. Indeed, the pattern on line 4 is removed since it is not part of the connected component containing the POF.

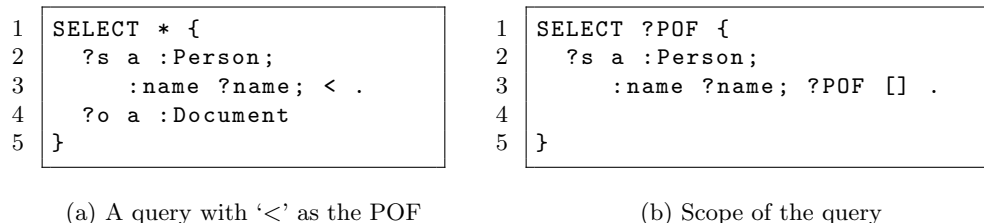


Fig. 1: Query auto-completion

Recommendation Capabilities. The scope may include content-specific terms, e.g, resources and filters, unlike to [3] since the graph summary is an abstraction that captures only the structure of the data. Recommendations about predicates, classes and named graphs are possible as in [3]. In addition, the use of the data directly allows to provide recommendations for specific resources.

4 Evaluation

Systems. In this section, we evaluate the recommendations returned by the proposed system, that we refer to as “S1”, against the ones provided by the approach in [3], which we refer to as “S2”.

Settings. We compare the recommendations with regards to (1) the response-time, i.e., the time spent on retrieving the recommendations via a SPARQL query; and (2) the quality of the recommendations. A run of the evaluation consists of the following steps. First, we vary the amount of information retrieved via the “LIMIT” clause. Then, we compare the ranked TOP-10 recommendations against a gold standard. The ranking is based on the number of occurrences of a recommendation. The gold standard consists in retrieving recommendations directly from the data without the LIMIT clause, and retaining only the 10 most occurring terms. The TOP-10 of the gold standard and the system are compared using the *Jaccard* similarity. We consider that the higher the similarity, the higher the quality of recommendations.

Queries. We used the query logs of the DBpedia endpoint version 3.3 available from the USEWOD2013² dataset. The queries³ were stripped of any pattern about specific resources, in order to keep only the structure of the query. In addition, we removed queries that contain more than one connected component. Queries are grouped according to their complexity, which depends on the number of triple patterns and on the number of star graphs. A group is identified by a string that has as many numbers as there are stars, with numbers separated by a dash '-' and representing the number of triple patterns in a star. For example, a query with two stars and one triple pattern each is then identified with *1-1*. This definition of query complexity exhibits the potential errors, i.e., a recommendation having zero-result, that a graph summary can have, as described in [2].

Graphs. We loaded into an endpoint the English part of the Dbpedia3.3⁴ dataset, which consists of 167 199 852 triples. The graph summary consists of 29 706 051 triples, generated by grouping resources sharing the same set of classes.

Endpoint. We used a Virtuoso⁵ SPARQL endpoint. The endpoint is deployed on a server with 32GB of RAM and with SSD drives.

Comparison. For each group of query complexity QC , we report in Table 1 the results of the evaluation, with $J1$ (resp., $J2$) the average Jaccard similarity for the system S1 (resp., S2); and $T1$ (resp., $T2$) the average response-time in ms for the system S1 (resp., S2). The reported values are the averages over 5 runs. We can see that as the LIMIT gets larger, the higher the Jaccard similarity becomes. Since the graph summary used in S2 is a concise representation of the graph structure, the data sample at a certain LIMIT value contains more terms than in S1. However, this impacts negatively on the quality of S2 as reflected by the values of J2. This shows the graph summary is subject to errors [2], i.e., zero-result recommendations. Nonetheless, it is interesting to remark that in S1 the recommendations can lead the query to an “isolated” part of the graph, from which the way out is through the use of “OPTIONAL” clauses. In S2, the graph summary allows to reduce this effect. The response-times for either system is similar, with S2 being slightly faster than S1. This indicates that directly querying the endpoint for recommendations is feasible. However, the significant difference in sizes between the graph summary and the original graph would become increasingly pre-dominant as the data grows.

² <http://usewod.org/>

³ <https://github.com/scampi/gosparqled/tree/master/eval/data>

⁴ <http://wiki.dbpedia.org/Downloads33>

⁵ Virtuoso v7.1.0 at <https://github.com/openlink/virtuoso-opensource>

QC	$J1$	$J2$	$J1$	$J2$	$J1$	$J2$	$J1$	$J2$	$J1$	$J2$	$J1$	$J2$	$J1$	$J2$	$J1$	$J2$		
	2		3		4		5		6		9		10		1-1		1-2	
10	0.12	0.12	0.17	0.21	0.15	0.21	0.16	0.19	0.14	0.16	0.17	0.19	0.16	0.19	0.11	0.09	0.19	0.18
100	0.15	0.17	0.28	0.26	0.27	0.28	0.28	0.29	0.24	0.26	0.25	0.26	0.25	0.27	0.12	0.11	0.24	0.22
500	0.24	0.27	0.34	0.29	0.34	0.30	0.36	0.35	0.38	0.31	0.42	0.27	0.43	0.26	0.15	0.18	0.29	0.29
QC	1-3		1-4		1-5		2-2		3-4		1-1-2		1-1-3		1-1-4			
10	0.62	0.64	0.23	0.22	0.15	0.19	0.17	0.17	0.15	0.06	0.55	0.38	0.50	0.49	0.38	0.43		
100	0.62	0.60	0.38	0.32	0.24	0.32	0.19	0.19	0.24	0.10	0.57	0.39	0.53	0.52	0.44	0.40		
500	0.62	0.59	0.60	0.34	0.25	0.29	0.25	0.22	0.21	0.12	0.57	0.40	0.55	0.51	0.47	0.46		
QC	$T1$	$T2$	$T1$	$T2$	$T1$	$T2$	$T1$	$T2$	$T1$	$T2$	$T1$	$T2$	$T1$	$T2$	$T1$	$T2$	$T1$	$T2$
	2		3		4		5		6		9		10		1-1		1-2	
10	107	81	119	82	127	81	129	82	144	85	314	197	688	468	97	79	103	80
100	108	81	180	84	147	95	202	86	173	88	311	198	701	458	122	84	140	83
500	141	91	192	96	144	79	172	99	149	101	337	207	701	467	127	89	133	111
QC	1-3		1-4		1-5		2-2		3-4		1-1-2		1-1-3		1-1-4			
10	101	108	108	87	104	93	102	80	114	83	107	391	106	87	105	87		
100	103	105	102	94	105	84	106	80	142	85	115	385	112	89	105	96		
500	126	105	141	92	136	97	158	94	137	117	126	400	133	99	139	102		

Table 1: Average Jaccard similarity ($J1$ for system S1 and $J2$ for S2) and response-times in ms ($T1$ for system S1 and $T2$ for S2) for each group of query complexity QC , and with the LIMIT varying from 10 to 500. The reported values are the averages over 5 runs.

Acknowledgement

This material is based upon works supported by the European FP7 projects LOD2 (257943).

Bibliography

- [1] Ambrus, O., Mller, K., Handschuh, S.: Konduit vqb: a visual query builder for sparql on the social semantic desktop
- [2] Campinas, S., Delbru, R., Tummarello, G.: Efficiency and precision trade-offs in graph summary algorithms. In: Proceedings of the 17th International Database Engineering & Applications Symposium. pp. 38–47. IDEAS '13, ACM, New York, NY, USA (2013)
- [3] Campinas, S., Perry, T.E., Ceccarelli, D., Delbru, R., Tummarello, G.: Introducing rdf graph summary with application to assisted sparql formulation. In: Proceedings of the 2012 23rd International Workshop on Database and Expert Systems Applications. pp. 261–266. DEXA '12, IEEE Computer Society, Washington, DC, USA (2012)
- [4] Clark, L.: Sparql views: A visual sparql query builder for drupal. In: International Semantic Web Conference. pp. –1–1 (2010)
- [5] Gombos, G., Kiss, A.: Sparql query writing with recommendations based on datasets. In: Yamamoto, S. (ed.) Human Interface and the Management of Information. Information and Knowledge Design and Evaluation, Lecture Notes in Computer Science, vol. 8521, pp. 310–319. Springer International Publishing (2014)
- [6] Kramer, K., Dividino, R.Q., Gröner, G.: Space: Sparql index for efficient autocompletion. In: Blomqvist, E., Groza, T. (eds.) International Semantic Web Conference. CEUR Workshop Proceedings, vol. 1035, pp. 157–160. CEUR-WS.org (2013)
- [7] Lehmann, J., Böhmann, L.: Autosparql: Let users query your knowledge base. In: Proceedings of the 8th Extended Semantic Web Conference On The Semantic Web: Research and Applications - Volume Part I. pp. 63–79. ESWC'11, Springer-Verlag, Berlin, Heidelberg (2011)
- [8] Rietveld, L., Hoekstra, R.: Yasgui: Not just another sparql client. In: SALAD@ESWC. pp. 1–9 (2013)