Exploring Linked Open Data with Tag Clouds

Xingjian Zhang, Dezhao Song, Sambhawa Priya, and Jeff Heflin

Department of Computer Science and Engineering, Lehigh University
19 Memorial Drive West, Bethlehem, PA 18015, USA
{xiz307,des308,sps210,heflin}@cse.lehigh.edu

Abstract. In this paper we present the contextual tag cloud system: a novel application that helps users explore a large scale RDF dataset. Unlike folksonomy tags used in most traditional tag clouds, the tags in our system are ontological terms (classes and properties), and a user can construct a context with a set of tags that defines a subset of instances. Then in the contextual tag cloud, the font size of each tag depends on the number of instances that are associated with that tag and all tags in the context. Each contextual tag cloud serves as a summary of the distribution of relevant data, and by changing the context, the user can quickly gain an understanding of patterns in the data. Furthermore, the user can choose to include RDFS taxonomic and/or domain/range entailment in the calculations of tag sizes, thereby understanding the impact of semantics on the data. The system runs on the BTC2012 dataset with more than 1.4 billion triples from which we extract over 380,000 tags. Several scalability challenges must be overcome in order to achieve a responsive interface.

Keywords: Tag Cloud, Data Visualization, Data Exploration

1 Introduction

We present the contextual tag cloud system[1] as an attempt to address the following questions: How can we help casual users explore the Linked Open Data (LOD) cloud? Can we provide a more detailed summary of linkages beyond the LOD cloud diagram[2]? Can we help data providers find potential errors or missing links in a multi-source dataset of mixed quality?

In analogy to traditional Web 2.0 tag cloud systems, an instance is like a web document or photo, but is "tagged" with formal ontological classes, as opposed to folksonomies. Tags are then another name for the categories of instances. We extend the expressiveness and treat classes, properties and inverse properties as tags that are assigned to any instances that use these ontological terms in their triples. The font sizes in the tag cloud reflect the number of matching instances for each tag, indicating the distribution of tags used by instances in this dataset. We allow the user to change their focus on a specific subset of instances in the

[1] http://gimli.cse.lehigh.edu:8080/btc/
[2] http://lod-cloud.net/

dataset by specifying a combination of ontological terms as the context on the fly. The context is a set of tags or the negation of tags, where the negation of a tag means that the tag is not assigned to the instances. Then the resulting contextual tag cloud will resize tags to display the conditional distribution of tags for the instances that satisfy this context.

Although materialization can lead to many interesting facts, a single erroneous axiom in the uncurated dataset could generate thousands of errors. Rather than attempting to guess which axioms are worthwhile, our system supports multiple levels of inference; and at any time a user can view tag clouds with the same context under different entailment regimes, which helps users understand the dataset better and helps data providers investigate the errors in the dataset.

To demonstrate the scalability of our system, we load the entire BTC2012 dataset. This complex dataset contains 1.4 billion triples, from which we extract 198.6M unique instances, and assign more than 380K tags to these instances. This multi-source, large-scale dataset brings us challenges in achieving acceptable performance and user-interface design as well as affordable preprocessing. Details of the technical aspects can be found in our research paper[3].

2 System Features

The initial tag cloud has no context (or semantically `owl:Thing`), and the tags in the cloud reflect the absolute sizes of instances related to each tag. We put classes and properties into two separate views, so that users will not treat a property called "author" (which may have domain Publication) as a class name by mistake. To emphasize that difference, we also add an icon with "C" or "P" in front of each tag. If a tag is clicked, it will be added to the current context, and then a new tag cloud will be shown for the updated context. A user can add/remove any tags to/from the context, and explore any dynamically defined types of instances. A user can also switch to Instance View to investigate the detailed triples of instances specified by the context.

A user can also change the inference regime: (1) No Inference; (2) Taxonomy Inference; (3) Domain/Range Inference; and (4) Both Inference. If a set of tags are entailed to be equivalent, we choose a cannoncial tag to group them under. We display a $\equiv$ after the canonical tag to indicate this; clicking it will display the equivalent tags. Also for any tag cloud, we can turn on the negation mode, and then the tag sizes indicate how many instances do not have this tag under the current context and inference level. A negation tag can be also added to the context, which mathematically means the relative complement. Fig. 1 shows an example of the property tag cloud with negation tag in the context.

We show tag clouds in pages when there are too many tags. To help users locate specific tags in the tag cloud, we initially sort the tags by their local names alphabetically. When the system receives a request (context and inference level),

[3] Xingjian Zhang, Dezhao Song, Sambhawa Priya and Jeff Heflin. Infrastructure for Efficient Exploration of Large Scale Linked Data via Contextual Tag Clouds. 12th International Semantic Web Conference. Sydney, Australia. 2013.

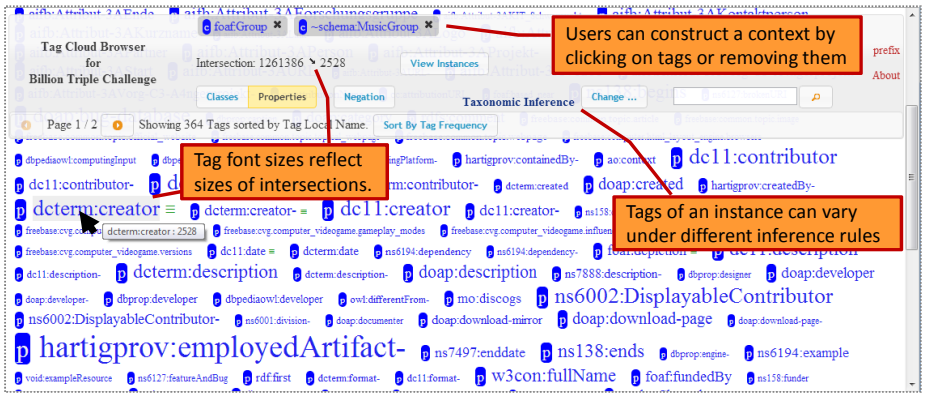

Fig. 1. Property tag cloud shows property usages of instances of `foaf:Group` that are not instances of `schema:MusicGroup`.

it will process tags in alphabetic order, and then stream out whatever is available for the requested page. If the user chooses to browse tags alphabetically, then the streaming of results is generally able to stay ahead of the user by pre-fetching results for tags on subsequent pages. Instead of browsing, a user can also search for tags by keywords. We index the local name, `rdfs:label` and `rdfs:comment` (if it exists) for each tag to support such keyword search. The retrieved tags will then be shown in the tag cloud sorted by their relevance to the keyword with their frequencies under the current context and inference regime. In addition, we provide sorting by tag frequency as another option, so that users can easily see the most popular tags under the current context and inference. However, we have to wait until all the frequencies are computed to enable this sort option. For some contexts, it can take a few minutes for the overall computation of thousands of pages of results. We show a progress bar of the computation and the estimated time left; but before sort is enabled, users can still browse by alphabetical order or search with keywords.

3 Use Cases

We believe our system can be used for multiple purposes. Here we shall briefly describe four scenarios where a user explores the BTC dataset.

Choose the right terms for SPARQL. A user wants to build a SPARQL query on lakes, but does not know what classes about lakes are available. Then by starting with a keyword search "lake", the user is presented with a tag cloud containing all tags that match the keyword, and finds that `dbpediaowl:Lake` contains the most instances. After picking this class, the user wonders what property to use for quering the area of a lake. Then by searching again with keyword "area", the user is presented with the contextual tag cloud with keyword-matched tags whose sizes reflect the intersection of the instances of the tags and

dbpediaowl:Lake. It turns out dbpediaowl:areaTotal is the best choice of the property.

Learn interesting facts. A casual user tries a keyword search on "Manhattan". There are classes of parks, streets, etc. located in Manhattan. However, it also has the class yago:ManhattanProjectPeople; the user adds this to the context to explore in more detail. In the resulting tag cloud, the user finds various categories for such people, and then searches again for "scientist". Then surprisingly there is a tag freebase:computer.computer_scientist. The user is intrigued, because she did not know that any computer scientists were involved in the effort to build the first atomic bomb. By adding that tag and switching to the Instance View, she finally learns that this scientist is John von Neumann.

Detect Co-reference Mistakes. Sometimes when two tags have a small unexpected intersection, it is due to an error, rather than an interesting fact. For example, a user finds the tag yago:BritishComputerScientist has one common instance with dbpediaowl:MusicalArtist (as shown by a very small tag). By adding this tag and looking into the triple details in the Instance View, we can see the two dbpediaowl:abstract values clearly refer to two different people who have the same name but different birth years but have been connected by an erroneous owl:sameAs statement.

Examine ontological errors. Under Domain/Range Inference, a user finds that foaf:Person appears in the tag cloud of context dbpediaowl:Software, implying that some people are software, or vice versa! If the user turns off the Domain/Range Inference, this error will disappear. What is wrong with this inference? If a property is claimed as having domain foaf:Person, then any instance using this property will be classified as the instance of this class. With this assumption in mind, the user adds both foaf:Person and dbpediaowl:Software to the context, selects the property view and Domain/Range Inference, and sorts the properties by frequency. Then the top tag is foaf:homepage, which has all the instances in the current context (by hovering the mouse over the tag, we can see the frequency of this tag). This is very suspicious, and by clicking on the "P" icon before foaf:homepage, the user can see that foaf:Person is an inferred super tag of this tag, and that causes the error. By checking the raw ontology we find that although the domain of foaf:homepage is owl:Thing in the foaf schema, two other sources in the BTC dataset make the claim that the domain is foaf:Person and foaf:Agent respectively.

4 Conclusions

In this paper we introduce the features and use cases of the contextual tag cloud system. The contextual tag cloud system is a novel tool that helps both casual users and data providers explore the BTC dataset: by treating classes and properties as tags, we can visualize patterns of co-occurrence and get summaries of the instance data. From the common patterns users can better understand the distribution of data in the KB; and from the rare co-occurrences users can either find interesting special facts or errors in the data.