Feature Generation using Ontologies during Induction of Decision Trees on Linked Data

Yordan Terziev

University of Duisburg-Essen, Essen, Germany
yordan.terziev@paluno.uni-due.de

Abstract. Linked data has the potential of interconnecting data from different domains, bringing new potentials to machine agents to provide better services for web users. The ever increasing amount of linked data in government open data, social linked data, linked medical and patients' data provides new opportunities for data mining and machine learning. Both are however strongly dependent on the selection of high quality data features to achieve good results. In this work we present an approach that uses ontological knowledge to generate features that are suitable for building a decision tree classifier addressing the specific data set and classification problem. The approach that we present has two main characteristics - it generates new features on demand as required by the induction algorithm and uses ontological knowledge about linked data to restrict the set of possible options. These two characteristics enable the induction algorithm to look for features that might be connected through many entities in the linked data enabling the generation of cross-domain explanation models.

1 Introduction

Humans can apply knowledge from one situation to another where concepts with similar properties occur. For example, the similarity of cold and flu as diseases is known to humans and when decision about the treatment of these diseases is made, they know that the knowledge about the one can be reused on the other because they have similar causes and treatment that mainly mitigates symptoms like pain or fever.

Machine learning (ML) algorithms for classification on the other hand don't automatically explore possible relationships between properties of different instances to build better models. While it is possible to identify correlation of particular data features to the class prior to the execution of the ML algorithm, the algorithms don't attempt to generate better features during model induction.

For example, if the training data of ML algorithm contains data of two patients that were treated with paracetamol, the one because he had flu and the other because he had cold – the relationship between the common symptoms and treatment, will remain undiscovered in the induced model. One reason for this is that ML techniques are typically applied on data, where the attributes are preselected and their values are considered simple types (e.g. the attribute disease and it's values cold and flu).

adfa, p. 1, 2011.

The training data for the ML algorithms is typically prepared manually by the user through three steps: data selection, data preprocessing and data transformation. In the first two steps the data instances for learning are selected, cleaned and formatted in the format required for the ML algorithms. In the last step the features (data attributes) used in the ML algorithm are selected and/or generated through scaling, decomposition or aggregation of existing features. Back to our example, a user of ML algorithm might expand the feature disease in the data set with the associated feature symptoms in order to achieve better prediction model and capture the "hidden" causality of treating symptoms with paracetamol instead of treating diseases.

The relationships between the features and their values such as between cold and flu and their symptoms is however frequently available in ontologies. This leads to the central idea of the work – to use existing ontological relationships between concepts to generate new features that improve the quality of the induced ML model.

This work is structured as follows: In the next section a formal representation of the problem is introduced, followed by an overview of the approach in section 3. Afterwards in section 4 we discuss the relevancy of the problem and what benefits would the approach bring. Section 5 presents the related work and differentiates the problem and our solution from the existing ones. In section 6 we present the research questions we plan to address and hypotheses we have. In section 7 the evaluation plan is presented followed by reflections in section 8 that conclude this work.

2 Problem Statement

To investigate the problem of using external ontological knowledge on the attributes and their values, we consider a classical supervised learning problem where we have a training set S of N training examples of the form $\{(x_1, y_1), \dots, (x_N, y_N)\}$ such that x_i is the feature vector of the $i-th$ example, described by set of discrete features X, and y_i is its discrete class label (i.e., class).

In classical supervised learning the feature vectors x_i are of the form $[f_1, f_2, \dots, f_n]$, where the features are either binary (i.e.,$f_i \in \{true, false\}$), numerical (i.e., $f_i \in \mathbb{R}$), or nominal (i.e., $f_i \in SYM, where\ SYM\ is\ a\ finite\ set\ of\ symbols$). The goal is to produce a model y = f(x) from the N training examples, so that it will predict the classes y of future unknown examples x with high accuracy. The function f(x) may also contain logical operations between the attributes. For the rest of the work we use the notation v_{ij} to represent the value of feature f_j of the $i-th$ instance.

In this work, we consider a scenario where f_j is concept in the ontology and its feature values are vertices in a linked data graph (e.g. SWRC Ontology [1] and the AIFB Dataset [2]). Linked data graph is a directed multigraph represented in the form G= $\{V, E\}$ and is built of set of triples in the form $\{(\mathrm{s}, \mathrm{p}, \mathrm{o})\}$, where the subject $s \in V$ is an entity, the predicate p denotes a property, and the object $o \in V$ is either another entity or literal. Each edge $e \in E$ in the graph is defined by a triple (s, p, o).

So for example the concept *Employee* might have the property *birth date* which is connected to a literal of type *Date*, but it might also have a relationship to another concept *Project,* which connects the concept Person to the project he/she is working

at. The ML task might be to train a classifier on existing linked data in order to find out suitable affiliation for new employees. The linked data and ontology graphs provide useful information for the model learning. For example, in the affiliation prediction task it might be the conferences at which the employee has published papers.

Finding suitable features for building a good classification model poses a difficult problem because a good feature might be referenced through many other types of concepts. For example, for the task of assigning a social network user to a particular group, a good feature might be the religious view of the front man of the music group liked by the user. This information is however 3 hops away from the origin user vertex in the linked data graph. Furthermore, in other classification problems the features providing good data predictions might be much further away than 3 hops. Not knowing the depth to which to explore the RDF graph causes further problems:

- The number of features to consider, grows exponentially with every hop away from the origin.
- A suitable halting criteria is required so that the approach doesn't search indefinitely throughout the ontology and linked data graph.
- Evaluating features with the entire set S might be too computationally expensive.

3 Approach

In this section we present an approach that is capable of generating new features from linked data and associated ontology during the induction of a decision tree. In decision tree induction, a feature f_x is said to be better than another f_y, if by splitting on its values the impurity (entropy) of the dataset is reduced more than if the split would be executed on the feature f_y. The entropy reduction is known as information gain (IG), and measures the information gained by separating on a feature in bits. IG is only one possible split evaluation function - to denote split evaluation functions in general we use $G(.)$. Since the values of f_x are vertices in the linked data graph G, our goal is to find related features that have higher value of G(.) than the original feature.

To explore the related features, we expand the original features (concepts in the ontology graph) with related concepts connected through outgoing properties and add these new featurec to the feature vector (cf. **Fig. 1**). Doing this for all connected features repeatedly, would eventually lead to exponentially growing number of options that should be evaluated. This requires heuristics that can restrict the number of possible features. The solution we propose is to consider only entities that are semantically related to the origin over some threshold value.

Semantic relatedness calculation in ontologies hasn't been widely studied and only few works exist. Mazuel et al. [3] calculate the semantic relatedness between ontological concepts pair only on semantic correct paths as defined in [4]. Therefore in our work we expand features from the origin feature in a breadth first search (BFS) manner considering the rules for semantically correct paths defined by Hirst & St-Onge [4]. The authors associate a direction in Upward(U), Downward(D) and Horizontal(H) for each property type and give three rules to define a semantically correct path

in terms of the three directions. Finally, Hirst & St-Onge enumerate 8 patterns of semantically-correct paths which match their three rules: {U, UD, UH, UHD, D, DH, HD, H}. Only concepts on outgoing paths from the origin entity conforming to these patterns are considered as possible features in the further process.

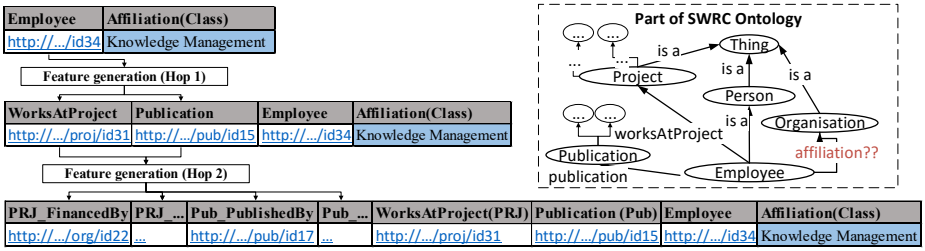

Fig. 1. Feature Generation using SWRC Ontology

Another problem we are addressing is the selection of suitable sized subset, as searching with the entire dataset might be impossible if the dataset is too large. To address this issue we propose the application of a statistical technique called Hoeffding bound [5] used in a streaming algorithm for decision tree induction [6]. As extending each value of feature f_x and calculating the resulting split value with the function G(.) for each related feature would be to expensive, we propose the use of the Hoeffding bound to select how much instances are enough to make the decision whether an entity should be expanded or not (i.e. expand to connected features).

To present the Hoeffding bound, consider a real-valued random variable r whose range is R. Suppose we have made n independent observations of this variable, and computed their mean $\bar{r}$. The Hoeffding bound states that, with probability $1-\delta$, the true mean of the variable is at least $\bar{r}-\epsilon$, where

$$\epsilon = \sqrt{\frac{R^2 \ln(\frac{1}{\delta})}{2n}} \tag{1}$$

In our work similar to [6], we use the Hoeffding bound to find lowest possible value for G(.) with the considered number of instances and probability $1-\delta$. While the Hoeffding bound in [6] is used to select between two possible features in our work we use it to make the decision whether the algorithm should expand the feature search one hop further away from the origin in the ontology graph. This decision is made based on two further parameters specified by the user: ER represents the expected entropy reduction represented in percent and DF(.), which is a decreasing function calculated based on ER and the number hops from the origin (NH). Basically the decision to expand the search a hop further is made in case that there are no features in the current depth that fulfil the expectations set by DF. The DF function is decreasing so that the expectations on the new features are lowered with increasing number of hops and the algorithm terminates. Setting high ER expectations and slowly decreasing DF function would eventually lead to much deeper exploration of the graph.

Before expanding to related concepts, it should be however ensured that a good estimate of the G(.) of the currently available features has been calculated. We do that by checking if the value of ϵ is smaller than a user defined threshold value σ. If this

condition is fulfilled this means that the induction algorithm has collected enough samples for a good estimate of G(.), but the best feature in the current hop is just not suitable for the classification problem. In that case, we expand all the features one hop further. To present the modifications we've made on the Hoeffding tree algorithm [6] (cf. **Table 1**), we further introduce the parameter ES, which is a dynamically managed expansion schema constructed of new attributes added during induction. It is required for sorting the new examples in the tree (expansion of the original feature vector x).

Table 1. Procedure: SemanticHoeffdingTree (S;X;G;δ;σ;ER;DF)

```
Let HT be a tree with a single leaf  l_1 (the root).
Let X_1 = X ∪ {X_∅}
Let Ḡ_1(X_∅) be the Ḡ obtained by predicting the most fre-
quent class in S
For each class y_k
  For each value x_ij of each attribute X_j ∈ X
    Let n_ijk(l_1) = 0
For each example (x, y_k) in S
  Expand x according to ES
  Sort (x, y) into a leaf l using HT
  For each x_ij in x such that X_i ∈ X_l
    Increment n_ijk(l).
  Label l with the majority class among the examples at l.
  If examples at l not all of the same class, then
    Compute Ḡ_l(X_j) for each attr. X_j ∈ X_l − {X_∅} using n_ijk(l)
    Let X_a be the attribute with the highest Ḡ_l
    Compute ∈ using Equation 1.
    If Ḡ_l(X_a) − DF(ER, NH) >∈ and X_a ≠ X_∅, then
      Replace l by an internal node that splits on X_a.
      For each branch of the split
        Add a new leaf l_m, and let X_m = X − {X_a}
        Let Ḡ_m(X_∅) be the Ḡ obtained by predicting the
        most frequent class at l_m
        For each class y_k and each value x_ij of each
        attribute X_i = X_m − {X_∅}
          Let n_ijk(l_m) = 0.
    Else If ∈< σ, then
      Expand each X_j ∈ X_l with sem. related concepts
      Add generated attributes(related concepts) to ES
      Adjust n_ijk(l) to the newly introduced features
      NH=NH+1
Return HT.
```

4 Relevancy

The ever increasing amount of linked data in different domains ranging from government open data, to social linked data [7] yet to linked medical and patients data [8] enables cross-domain interlinking. It provides new opportunities to ML communities: such as:

- Analysis of Linked Open Data (LOD): Interesting here is analysis of government open data as presented in [9], but also interesting for medical research e.g. connecting medical gene research and statistics about patients' disease progression.
- Social networks data has inherently graph structure (such as Facebook Graph API), which can be transformed to linked data by using approaches like the one presented in [7]. One example would be to find out why particular group of people clicked on specific advertising e.g. user clicking on holiday ad because their friends recently posted or liked pictures from a holiday resort location.
- A typical application field of ML algorithms are recommender systems. Bouza et al. [10] present an approach which uses semantic information available for items to build item's feature vector and subsequently using the items rating by a user to build user's decision tree model. The model is then capable to assign a rating to items not yet rated by the user. Similarly, our approach can be used to dynamically select suitable features for the user's rating decisions.

5 Related Work

Typically learning from linked data (RDF data) is divided in pre-processing, instance extraction optional feature extraction and the actual learning [11]. In the pre-processing step – some of the RDF verbosity is reduced, additionally methods like RDFS/OWL inferencing can be used to more efficiently expose the relevant information. In RDF it is typically accepted that an instance is represented by a resource in the RDF graph [11], however the resource itself doesn't contain any information about the instance. The actual description of the instances is represented by the neighborhood around the resource. Therefore, machine learning approaches such as [12, 13] achieve instance extraction by extracting a full subgraph to a given depth.

After the instance extraction two further options are available either one executes the learning algorithms directly on the extracted subgraphs or extracts feature vectors. In the first option different kernel functions are applied - one representative is the Weisfeiler-Lehman (WL) graph kernel, which computes the number of subtrees shared between two graphs by using the WL test of graph isomorphism. While the WL kernel is designed for all kind of graphs, Lösch et al. propose in their work [13] graph kernels specialized to RDF. Their kernel addresses specifics of the RDF graph such as that RDF node labels are used as identifiers occurring only once per graph and nodes may have a high degree. This differs from e.g. chemical compound graphs that usually have few node labels which occur frequently in the graph and nodes have a

low degree. However, both graph kernel approaches do not create feature vectors and work with a fixed depth of instance extraction – usually 2 hops.

Another approach is [14] where the authors introduce an expressive graph-based language for extracting features from linked data and a theoretical framework for constructing feature vectors from the extracted features. The construction of the feature vector there is mainly driven by the provided queries (SPARQL/NAGA) and their result, which is used as basis for the feature generation.

Yet another work that uses similar technique as ours for creating features vectors out of extracted instances is the one presented by Paulheim [9], where however the main goal is the generation of possible interpretations for statistics using linked open data. The main difference to our work regarding the feature generation technique is that the author proposes an approach that firstly generates all possible properties for all features, followed by feature selection afterwards. Further differences to our work are that no selection of subset of instances for the feature generation step is done as we propose with the Hoeffding bound and all related features are considered.

6 Research Question and Hypotheses

Two main research question should be answered:

- How can ontological knowledge (especially relationships between concepts) be used to automatically generate features for a specific entity and a given classification problem using instance extraction with dynamic depth (cf. section 3)?
- How can ontological knowledge and instance extraction with dynamic depth be used to automatically improve manually generated/selected features in training data so that the accuracy of the induced ML model on unknown examples improves?

The main hypothesis is that enabling machine learning algorithms to dynamically expand the set of possible features through the linked data graph would improve the classification accuracy of the produced classifier on unknown instances in comparison to models induced with static depth of instance extraction.

7 Evaluation Plan

To validate our hypothesis, we envision twofold evaluation: first we would compare our approach to the induction of decision tree using the unmodified Hoeffding tree algorithm [6] with a fixed depth of instance extraction (2 hops) as suggested in [12, 13]. We do this evaluation to minimize the effect of the underlying ML induction algorithm and only measuring the effect of the dynamic expansion depth.

Second our goal is to compare the classification performance of the models constructed with the here presented approach to the state of the art approaches for learning on RDF Data. The works of Lösch et al [13] and Vries,G.de [12] are envisioned as baselines. In particular, we compare the approach to both works, because they restrict the depth of the instance extraction to 2 hops.

We plan to execute the evaluation on the entity classification and link prediction learning tasks presented in [13]. The algorithms would be trained and tested on the two datasets used in [13] and [12] namely: the AIFB dataset [2] (SWRC Ontology), where the main task is classifying the affiliation of a staff member to a research group, and the dataset from LiveJournal.com (Friend of a Friend ontology) where the main task is to classify persons in one of four age classes.

8 Reflections

In this work only the general idea of dynamic feature generation during induction of decision tree is presented and some specifics haven't been addressed such as: how referenced list are handled (e.g. multiple published papers), how semantic similarity can be used in this context or how cycles in the linked data graph are managed. Further research question that is within the research scope, but not directly addressed in this work is how multiple origins would be handled and how would this impact feature selection. All these are subject of future work.

While a similar technique of expanding linked data values has been explored in [9] with the goal of generating hypothesis, in the current work we suggest a decision tree induction algorithm that executes an on demand expansion of linked data features considering only semantically related concepts. We expect that through these characteristics of the induction algorithm the graph can be explored in greater depth, thus finding cross-domain explanations, that eventually would be a better representation for the classification problem.

Another difference of our technique to the one presented in [9] is that we explore new features with a subset of the instances using the Hoeffding bound [5] to select how many instances are enough in order to evaluate if deeper exploration is required or the currently selected features are good enough. Thus we envision that the approach is capable of exploring more possible features with less expensive calculation of split evaluation functions.

Acknowledgments

I would like to thank my supervisors Prof. Dr. Volker Gruhn and Dr. Tobias Brückmann for their support and the opportunity for the realization of this work.

References

1. York Sure-Vetter, Stephan Bloehdorn, Peter Haase, Jens Hartmann, Daniel Oberle: The SWRC Ontology - Semantic Web for Research Communities. In: Carlos Bento, Amilcar Cardoso, Gael Dias (ed.) Proceedings of the 12th Portuguese Conference on Artificial Intelligence - Progress in Artificial Intelligence (EPIA 2005), 3803, pp. 218–231. Springer, Covilha, Portugal (2005)

2. Bloehdorn, S., Sure, Y.: Kernel methods for mining instance data in ontologies. Springer (2007)
3. Mazuel, L., Sabouret, N.: Semantic Relatedness Measure Using Object Properties in an Ontology. In: Proceedings of the 7th International Conference on The Semantic Web, pp. 681–694. Springer-Verlag, Berlin, Heidelberg (2008)
4. Hirst, G., St-Onge, D.: Lexical chains as representations of context for the detection and correction of malapropisms. WordNet: An electronic lexical database 305, 305–332 (1998)
5. Wassily Hoeffding: Probability Inequalities for Sums of Bounded Random Variables. Journal of the American Statistical Association 58, 13–30 (1963)
6. Domingos, P., Hulten, G.: Mining High-speed Data Streams. In: Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 71–80. ACM, New York, NY, USA (2000)
7. Weaver, J., Tarjan, P.: Facebook linked data via the graph API. Semantic Web 4, 245–250 (2013)
8. Pathak, J., Kiefer, R.C., Chute, C.G.: Applying Linked Data Principles to Represent Patient's Electronic Health Records at Mayo Clinic: A Case Report. In: Proceedings of the 2Nd ACM SIGHIT International Health Informatics Symposium, pp. 455–464. ACM, New York, NY, USA (2012)
9. Paulheim, H.: Generating Possible Interpretations for Statistics from Linked Open Data. In: Simperl, E., Cimiano, P., Polleres, A., Corcho, O., Presutti, V. (eds.) The Semantic Web: Research and Applications, 7295, pp. 560–574. Springer Berlin Heidelberg (2012)
10. Amancio Bouza, Gerald Reif, Abraham Bernstein and Harald Gall: SemTree: Ontology-Based Decision Tree Algorithm for Recommender Systems
11. Peter Bloem, Gerben de Vries: Machine Learning on Linked Data, a Position Paper co-located with European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD 2014), Nancy, France, September 19th, 2014. In:
12. Vries, G. de: A Fast Approximation of the Weisfeiler-Lehman Graph Kernel for RDF Data. In: Blockeel, H., Kersting, K., Nijssen, S., Železný, F. (eds.) Machine Learning and Knowledge Discovery in Databases, 8188, pp. 606–621. Springer Berlin Heidelberg (2013)
13. Lösch, U., Bloehdorn, S., Rettinger, A.: Graph Kernels for RDF Data. In: Simperl, E., Cimiano, P., Polleres, A., Corcho, O., Presutti, V. (eds.) The Semantic Web: Research and Applications, 7295, pp. 134–148. Springer Berlin Heidelberg (2012)
14. Cheng, W., Kasneci, G., Graepel, T., Stern, D., Herbrich, R.: Automated feature generation from structured knowledge. In: Proceedings of the 20th ACM international conference on Information and knowledge management, pp. 1395–1404 (2011)