Predicting Appropriate Semantic Web Terms from Words

Lushan Han and Tim Finin
University of Maryland, Baltimore County
1000 Hilltop Circle, Baltimore MD 21250
lushan1@umbc.edu, finin@umbc.edu

Abstract

The Semantic Web language RDF was designed to unambiguously define and use ontologies to encode data and knowledge on the Web. Many people find it difficult, however, to write complex RDF statements and queries because doing so requires familiarity with the appropriate ontologies and the terms they define. We describe a system that suggests appropriate RDF terms given semantically related English words and general domain and context information. We use the Swoogle Semantic Web search engine to provide RDF term and namespace statistics, the WordNet lexical ontology to find semantically related words, and a naïve Bayes classifier to suggest terms. A customized graph data structure of related namespaces is constructed from Swoogle's database to speed up the classifier model learning and prediction time.

Motivation and Objectives

The Semantic Web is realized as a huge graph of data and knowledge. The graph's building blocks consist of literal values and RDF terms representing classes, properties and individuals. Syntactically, an RDF term is expressed as a URI like *http://xmlns.com/foaf/0.1/Person*, which has two parts: a namespace (*http://xmlns.com/foaf/0.1/*) identifying the ontology defining the term and a local name (*Person*) selecting a term in the ontology. The use of namespaces avoids introducing ambiguity and allows two terms in different ontologies to share a local name.

For example, the class *Party* may be defined in an ontology about politics as well as in another describing daily lives. Qualifying *Party* with a namespace removes the ambiguity. Authoring or querying knowledge Semantic Web is difficult because it requires people to select an ontology from man for the concepts they want to use. The term *Party*, for example, is defined in 352 Semantic Web ontologies known to Swoogle. Moreover, other ontologies define possibly related concepts, with local names *Celebration* and *Organization*. It would be convenient if a user could use natural language words as her vocabulary and an knowledgeable system would suggest appropriate Semantic Web terms based both on the observed experience data of how people used different namespaces together and on the user's prior namespaces or domain information. Such a system could support metadata systems like Flickr's *machine tags* (Schmitz 2006), which also require qualified namespaces whose selection is hard for end users who know nothing or little about ontologies.

Problem Analysis

Given an RDF local name, how do we know which namespaces are suitable to qualify it? A simple solution is to ask a Semantic Web search engine, like Swoogle (Ding et al. 2004) to return the most highly ranked public namespaces that define the term. However, this is based on an exact string match. In many cases, we might desire terms whose local names are semantically related. For example, if a user gives the verb *associate* and no popular namespace defines a matching term, but the word *relate* is defined in a very popular namespace, then generally we can substitute *relate* for *associate*.

Thus, for a given input word we find a set of synonyms or semantically related words from WordNet (Miller 1998). For each one, we use Swoogle to find ontologies that define a term with a corresponding local name. The selection process involves a threshold based on Swoogle's ontoRank (Ding 2005a) metric to limit the choices to popular ontologies. The result is a set of candidate pairs where each pair specifies an RDF namespace representing the ontology and a local name representing a term defined in that ontology.

Given a set of candidate pairs *(namespace, word),* selecting the best one depends on domain information provided by users. In fact, the namespaces themselves convey lots of domain information. If users have some prior used namespaces, these may be used to determine which are the most appropriate based on correlation among namespaces observed on the Semantic Web. Lacking a history of namespace use, users may directly tell the system their general domain with a set of words and phrases, such as *geography* or *ecoinformatics*, etc. The system can then choose the most representative namespace in that domain as an implicit prior namespace for users. In sum, we can focus on one key problem – finding the namespace with the highest conditional probability in the presence of the given namespaces.

Predicting Appropriate Namespaces

Computing exact conditional probabilities is very expensive when the number of nodes is large. Fortunately, we need only know the rank ordering of namespaces with respect to their conditional probability, enabling us to find maximum posteriori (MAP) hypotheses. A naïve Bayes (NB) classifier approach can be used for this purpose. Although this makes a conditional independence assumption that is not true in most cases, it performs very well in text classifications and other problems. Since our training dataset is large in terms of observations and nodes, the use of

NB is a practical approach because of its computational simplicity. Once we obtain an ordering of candidate namespaces with respect to their conditional probability in the presence of the input namespaces, we can select the first one that matches the namespace of any candidate pair.

We have 2.5 million entries in our dataset, which comes from all RDF documents indexed by Swoogle (Ding et al. 2004), a crawler-based Semantic Web search engine that discovers and indexes documents containing RDF data. Running since 2004, Swoogle has indexed nearly 2.5M such documents, about 10K of which are ontologies that define terms. As new Semantic Web documents are discovered, Swoogle analyzes them to extract their data, compute metadata and derive statistical properties. The data is stored in a relational database and an information retrieval system (currently Lucene). In addition, a copy of the source document is stored and added to an archive of all versions of every document discovered.

Each entry contains several namespaces, which are represented by their Swoogle IDs. We apply the ten-fold method to the dataset. For each time, the dataset is split into a training set and a test set. The training set is used to train a model that will output the ten most probable namespaces when given a group of input namespaces. We test the model Test by selecting an entry from the test set, randomly removing one of its several namespaces, and using the remaining namespaces as the input namespaces to the model. The target namespace is just the namespace we removed. We evaluate the model by observing if the target is among the top ten suggestions and how high on the list it appears. Judging if the target is among the top ten provides a simple test of the soundness of the model in computing MAP hypotheses.

Our dataset has more than 20K distinct namespaces, which would require 20K categories for classification, making NB computationally expensive. Instead of brute force counting, we use another approach exploiting namespace locality. We observe that the namespace co-occurrence graph is very sparse – a few namespaces such as FOAF (Ding 2005b) are very widely used, but most are used with a small set of domain-related namespaces. Consequently, it makes no sense to iterate over all possible classifications to find the most probable ones.

Namespace contexts

A graph of namespaces and their connections enables us to take advantage of a node's locality information. Given a namespace, we can quickly find all its neighbors and the degrees of the connectivity to its neighbors. For the graph data structure, based on the fact that most of namespaces have a limited number of neighbors, we use a modified adjacency-list representation in that the neighbor list is not a linked list but a hash map in order to speedup the operation of finding the degree of connectivity (the number of connected edges, which is stored with node objects in the neighbor hash map). The training program reads entries from the training dataset in sequential order. For those namespaces included in one entry, we add the edges of the complete graph of them, a clique, to the graph data structure. We can compute the basic prior and conditional probabilities P(vj) and P(ai | vj) with our graph data structure by using degree of connectivity between nodes. We don't precompute these probabilities in the training process, but compute them as required when doing prediction.

Given a group of input namespaces A1, A2…An, how do we get the most probable classifications Vjs, in our case, the namespaces? The MAP formula

$$v_{MAP} = \arg\max_{v_j \in V} P(v_j \mid a_1, a_2 ... a_n)$$

implies that possible Vjs must have connections to every Ai which again means the possible Vjs must be neighbors of every Ai. This fact enables us to choose the least connected node (having fewest neighbors) in A1, A2…An, and its neighbors must include all possible Vjs. The benefit is that this reduces the possible classifications from 20,000 to the number of neighbors of the least connected namespace. This is significant reduction because most Semantic Web namespaces have a limited number of neighbors.

Results and Conclusion

Our initial evaluation shows that the approach is effective at predicting appropriate RDF terms given a word that is semantically related to the term's local name. The probability that the correct namespace is the top prediction is 0.556, in the top three suggestions 0.675, and in the top ten 0.919. The approach is also efficient and practical for the current and expected scale of the Semantic Web. It takes only about 30 seconds to read 2.5M entries from the training dataset and build the graph model in a Celeron 3.0 GHz computer with one GB memory. Opportunities for future work include conducting more careful evaluations, studying the trade off between precision and recall and enhancing the context mechanism.

References

L. Ding, et al. 2004. Swoogle: A Search and Metadata Engine for the Semantic Web, 13th ACM Conf. on Information and Knowledge Management, Washington, D.C.: ACM.

L. Ding et al., 2005a. Finding and Ranking Knowledge on the Semantic Web, Proc. 4th Int. Semantic Web Conf.

L. Ding et al., 2005b. How the Semantic Web is Being Used: An Analysis of FOAF Documents, 38th Hawaii Int. Conf. on System Sciences.

L. Ding and T. Finin, 2006. Characterizing the Semantic Web on the Web, Proc. 5th Int. Semantic Web Conf. Athens, GA.

T. Mitchell. 1997. Machine Learning. Reading, McGraw Hill.

S. Russell and P. Norvig. 1995. Artificial Intelligence – A Modern Approach. Reading, Prentice Hall

G. Miller, 1998. WordNet, Cambridge, Mass: MIT Press.

.J. Sachs, et al. Oct. 2006. Using the Semantic Web to Support Ecoinformatics. AAAI Fall Symp. Semantic Web for Collaborative Knowledge Acquisition.

P. Schmitz. 2006. Inducing ontology from flickr tags, Collaborative Web Tagging Workshop at WWW2006, Edinburgh.