METEOR-S Web Service Annotation Framework with Machine Learning Classification

Nicole Oldham, Christopher Thomas, Amit Sheth, and Kunal Verma

LSDIS Lab, Department of CS, University of Georgia, 415 GSRC, Athens, GA 30602
{oldham, cthomas, amit, verma}@cs.uga.edu

Abstract. Researchers have recognized the need for more expressive descriptions of Web services. Most approaches have suggested using ontologies to either describe the Web services or to annotate syntactical descriptions of Web services. Earlier approaches are typically manual, and the capability to support automatic or semi-automatic annotation is needed. The METEOR-S Web Service Annotation Framework (MWSAF) created at the LSDIS Lab at the University of Georgia leverages schema matching techniques for semi-automatic annotation. In this paper, we present an improved version of MWSAF. Our preliminary investigation indicates that, by replacing the schema matching technique currently used for the categorization with a Naïve Bayesian Classifier, we can match web services with ontologies faster and with higher accuracy.

1 Introduction

With the growing popularity of Web services, the discovery of relevant Web services is a problem. The current approach for discovery is to perform a keyword based search of the UDDI. Like other keyword based search techniques this suffers from problems such as ambiguity, synonymy, etc. One solution to this problem is to make Web services descriptions more meaningful by annotating the service descriptions with machine understandable metadata from shared ontologies [5]. Semantic search engines can then take advantage of this metadata. The most common approach to annotation is to relate elements in the WSDL description to domain specific ontologies. This process will enable the discovery, interoperation, and composition of Web services to be much more efficient. The current research of Web service annotation largely focuses on manual annotation [1], which, looking at the growing numbers of Web services and ontologies, will be time consuming and expensive.

Furthermore, individual ontologies can be very large (e.g. the world-fact-book ontology contains more than 1100 concepts. It has also been reported that real world populated ontologies often exceed 1 million instances [6]. This makes even the discovery of the corresponding concepts within the ontology a tedious task. Further complications arise when a WSDL can be matched to multiple ontologies. Taking these problems into consideration, it is imperative that an efficient tool for (semi-)automatic annotation be used.

We describe the architecture, implementation, and functionality of MWSAF as well as our machine learning approach to improve MWSAF in this paper. The main contributions of our work are:

J. Cardoso and A. Sheth (Eds.): SWSWPC 2004, LNCS 3387, pp. 137 – 146, 2005.

- Addressing the need for semantics in the Web services framework, and providing a detailed approach that identifies four types of semantics for describing Semantic Web services.
- Identifying the technical challenges in (semantic) annotation of Web services.
- Implementing a machine learning approach to quickly classify Web services into domains.

MWSAF is an approach for semi-automatic annotation. It annotates WSDL descriptions of the services with metadata from relevant ontologies. Our approach will use MWSAF for annotation, but will replace the method used by MWSAF for classifying a Web service into a domain.

This paper will provide an explanation of how MWSAF currently tackles the complicated task of automatically matching in Section 2. An evaluation of the accuracy and performance of MWSAF will comprise Section 3. In Section 4 we will present our machine learning approach for enhancing MWSAF and evaluate this approach in Section 5. Finally, in Section 6 we will discuss the related and future work in this area.

2 Meteor-S Web Service Annotation Framework (MWSAF)

Consistent with Jim Hendler's hypothesis "a little semantics goes a long way", METEOR-S augments the current Web services technology by adding semantic metadata to the syntactical WSDL descriptions. Since WSDL is a de facto standard, this method seems more practical to us than completely changing the paradigm of Web services descriptions. As identified in [1], the four categories of semantics in the complete Web process lifecycle are:

- Data Semantics (semantics of inputs / outputs of Web services),
- Functional Semantics (what does a service do),
- Execution Semantics (correctness and verification of execution),
- QoS Semantics (performance/cost parameters associated with service), MWSAF focuses on data semantics.

2.1 MWSAF Matching Issues and Techniques

In order to annotate, concepts from the WSDL must be matched to concepts from an appropriate ontology. Therefore, the suitable ontology must be identified out of an ontology store. A Web service must be categorized into a domain and the ontology most appropriate for annotation must be determined. Due to the difference in expressiveness of WSDL and OWL, it is difficult to directly match the two formats. WSDL files describe bindings for Web services while ontologies represent concepts and the relationships between them. MWSAF proposes to bridge the gap by converting each to a common representation called schemaGraphs. A schemaGraph is simply a graph or tree representation of the XML or DAML-S document. The conversion rules are explained in-depth in [1].

Once the schemaGraphs have been created, matching algorithms are executed on the graphs in order to determine similarities. Once a concept is matched against all the concepts in an ontology, the best mapping according to the criteria is chosen for annotation. Several algorithms have been defined for matching.

2.1.1 ElemMatch and Schema Match

MWSAF can perform both element and structure level schema matching. The ElemMatch function performs the element level matching based on the linguistic similarity of the names of the two concepts. Note that the two concepts must have similar names for the match to be recognized. A Porter Stemmer [7] is used to extract the root word. The synonyms are checked using WordNet®. An abbreviation dictionary is referenced to handle acronyms and abbreviations. ElemMatch also uses an NGram algorithm to determine element level linguistic similarity. The results of these algorithms determine an ElemMatch score.

The SchemaMatch function examines the structural similarity between two concepts. A concept in an ontology is usually defined by its properties, superclasses and subclasses. Since concept labels are somewhat arbitrary, examining the structure of a concept description can give more insight into its semantics. SchemaMatch accounts for this by calculating the geometric mean of Sub-concept Similarity and the Sub-concept Match. The Sub-concept Similarity is the average match score of each individual property of the concept. Sub-concept match can be defined as the fraction of the total number of properties of a concept that are matched.

Both ElemMatch score and SchemaMatch score are then used to determine the final match score. Formulas, implementation details and results are shown in [1].

2.2 Web Service Classification

Once the best set of matches has been determined, the Web service can be classified to a domain based on the previous calculations. A set of mappings is created for each ontology. Two measures are derived from these sets of mappings; the first is the Average Concept Match and the second is the Average Service Match.

The Average Concept match tells the user about the degree of similarity between matched concepts of the WSDL schema and ontology. This measure is used to decide if the computed mappings should be accepted for annotation. The Average Service Match helps to categorize the service into categories. It is calculated as the average match of all the concepts of a WSDL schema and a domain ontology. The domain of the ontology corresponding to the best Average Service Match also represents the domain of the Web service.

2.3 Web Service Annotation

After the matching has been completed, the annotations can be added to the WSDL. The best match set is presented to the user to choose to accept or reject the matches. Concepts can also be matched manually if the automatic technique failed in some or all matches. Upon acceptance, the mappings are written back to the WSDL file. The output of the tool is the semantically augmented WSDL file.

3 Evaluation of MWSAF

MWSAF was tested using the Weather and Geographical domains. Although the ontologies used are not comprehensive enough to cover all the concepts in these domains, they are sufficient enough to serve the purpose of categorization. For overall results for matches, annotations, and more details regarding the test bed see [1]. We will focus on the results of tests of categorizing services into domains.

3.1 Classification Accuracy

The accuracy of the MWSAF classification depends on the number to which a threshold is set. The services are categorized based on the categorization threshold (CT), which decides if the service belongs to a domain. If the best average service match calculated for a particular Web service is above the CT then the service is predicted to belong to the corresponding domain. Figure 1 depicts the categorization obtained by applying the algorithm on a set of 24 (15 Geography, 9 Weather) Web services for different CT values. In the case of CT = 0.5, the recall was 58%. Whereas for CT = 0.4, although all Web services are categorized, two services from the weather domain have been wrongly categorized in the geographical domain. By tweaking the CT value, the user can either improve precision or recall of the system.

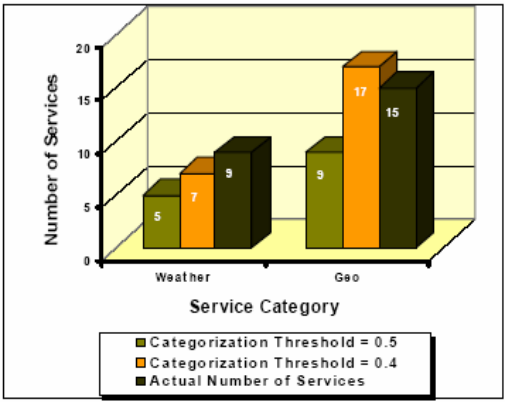

Fig. 1. Categorization statistics of Web services

3.2 Performance

In order to find matches, MWSAF performs an exhaustive search. Each node of a WSDL schemaGraph is compared to every node of each ontology schemaGraph. For this reason the current matching techniques are very expensive in terms of time. This cost is greatly increased as more ontologies are added to the store, thus rendering the algorithm unscalable. We propose replacing this algorithm with a machine learning technique to enhance the speed and efficiency of choosing the appropriate ontology for a Web service by eliminating the node by node matching that MWSAF classification requires.

4 Machine Learning Approach for Web Service Classification

To overcome the apparent drawbacks of the exhaustive search performed in the current version of MWSAF, we decided to replace the schema matching approach for classification with a machine learning approach that uses the Naïve Bayes Classifier implemented in the WEKA toolkit [2] to predict the domain a particular Web service belongs to. This classifier determines the probability that a service belongs to a category by taking the product of the probabilities of each single word belonging to this category. Naïve Bayes Classifiers have quadratic time complexity during the training phase and linear complexity for domain prediction, making this approach highly scalable. Furthermore, these classifiers have been successfully deployed in text classification tasks, even though the independence assumption for distinct features that the classifier makes is clearly violated in natural language, where the context of a word in a sentence is actually quite important.

The features we use to classify the WSDL descriptions into domains are the method names and the argument names, for which we can assume contextual independence, which renders the Naïve Bayes an ideal classifier for our purposes.

The representation we use for a WSDL description as input to the classifier is a feature vector. Each feature represents the frequency of a particular word in the corresponding WSDL file. The position of the word is determined a priori, by parsing all WSDL files and building a dictionary of all the words used in the corpus of WSDL files. Each word has thus a unique ID which corresponds to its position in the feature vector. We use two different measures for word frequency. The first is the raw number of occurrences of a word in the description, the second is a TF-IDF representation [8] of this raw frequency.

4.1 Feature Extraction

The extraction of words from the WSDL descriptions is straightforward. The WSDL XML is parsed. Method names and attribute names are first extracted and then split by our rule of thumb that conventionally a new word in a method name is introduced either by a capital letter or by an underscore/dash. The resulting words are then stemmed [7]. In addition to usual stop words, common terms found in method names such as ‘get’ and ‘set’ are removed. The result of this is a bag of word stems. For every occurrence of a word the corresponding entry in the feature vector is incremented. For TF-IDF, this number is then replaced by the TF-IDF measure.

4.2 Training

Since classification is a supervised learning task, we extract training sets and test sets from our corpus which was provided by Andreas Hess and N. Kushmeric [3]. The training set given to the classifier is a feature matrix. The rows correspond to the feature vectors. Each column of a vector contains the frequency of the word that corresponds to that position of the vector with the exception of the last column which contains the classification of the Web service corresponding to the row vector.

The classifier is then built with this training data. From this, the classifier learns the words and the frequencies of words common to a particular domain.

4.3 Prediction

The second step in our approach is to predict the domain of a given WSDL based on the word frequencies for that WSDL. A set of unclassified WSDL files is given to the classifier.

In order to predict the domain of the given WSDL, the Naïve Bayes classifier only requires a vector of word frequencies. It then compares these frequencies to the training data. During training the classifier learned words that are significant for each domain. Based on the frequencies for the given WSDL and previous training regarding domains and associated words, the classifier then produces numeric predictions for each known domain. The ontology corresponding to the domain with the highest probability is then used for annotation. The classifier generally predicts a domain with almost complete certainty, but in some situations a WSDL will contain high frequencies of words from two domains. In this situation the predictions for both domains are high. This is particularly useful because MWSAF has difficulty handling situations where WSDL matches to multiple ontologies.

5 Application of MWSAF with Machine Learning

Our proposed approach functions by combining all of the phases named in Section 4 with very little involvement from the user. The training of the classifier occurs automatically when MWSAF is loaded. The parser extracts all of the relevant words and passes the frequencies of these words to the classifier along with the associated domain. From this data, the classifier learns which words are highly associated with a domain. Next, the user must load the WSDL file of the Web service to be annotated. The parser then extracts the relevant words from the loaded WSDL and passes the frequencies of those words to the classifier. The classifier uses the Naïve Bayes algorithm to calculate the probability that the Web service might belong to each of the known domains by comparing the word frequencies for this WSDL to what it has previously learned. MWSAF presents the domain for which the classifier calculated the highest probability to the user. If the user accepts the domain, then the corresponding ontology is displayed and the user may then begin selecting matches between the WSDL file and the ontology. If the user rejects the predicted domain, he may chose to predict the domain using the previous technique for classification described in Section 2. When the user has selected the matches between the ontology and WSDL file, the annotations are written to the WSDL.

6 Evaluation of Machine Learning Approach

The machine learning approach described above is advantageous to the MWSAF tool because it is not limited by some restrictions and flaws that limit the MWSAF classification technique. We will discuss the accuracy of the approach for correctly predicting the domain of a service. Then we will discuss why this approach is much faster than the MWSAF approach.

6.1 Classification Accuracy

Due to the fact that the machine learning approach does not rely on an extensive ontology to match to, the accuracy is generally better than the accuracy of classification in MWSAF.

Testing for comparison with MWSAF was done with 37 Web services (16 Weather, and 21 Geography). For quality evaluation purposes, we extracted several training sets of different sizes to measure the quality of the results for cases ranging from much prior knowledge of the domains to be classified (90% training set size) to poor prior knowledge (10% training set size). Five random distributions are then evaluated for every training set size. An average of the five rounds was then calculated for that percentage. The testing results are shown in Graph 2.

Notice that a TF-IDF approach had no positive impact on the results. It is likely that after removing stop words most remaining terms are similarly significant.

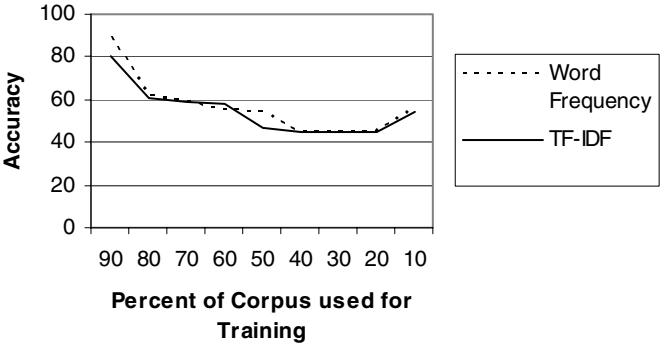

Fig. 2. Machine Learning Classification Statistics

As less training data is provided the accuracy drops first, but remains stable then. This is an encouraging result, because it shows that the number of training examples does not have much impact on the classifier, making it suitable for classifying large sets of unknown services. Note that there are some situations where a WSDL contains many words from both domains. In that type of situation the classifier may give the number 55 for Weather and 45 for Geography. The service might be considered "incorrectly classified" in the results of Figure 2. The results shown in the graph include these situations where a service is wrongly classified but by numbers indicating that it belongs to both domains.

Figure 2 corresponds to a test run on the Geography and Weather domains for comparison with the MWSAF results, but tests were performed for additional domains. The averages drop if the WSDL files in a particular domain are not related enough to have common frequencies. For example, the test bed contains a business domain where the WSDL files and the words used within them are unrelated. The classifier performs poorly when categorizing these business services. Figure 3 illustrates the performance when testing with several domains but excluding business.

The backbone of this approach is finding common words and frequencies of words between WSDL files; therefore, if the terms used for the description of different domains show a high overlap, the classifier cannot identify distinct features and thus may predict the wrong domain. Future work will be to investigate a solution to this problem.

Another limitation of this approach occurs when Web services do not have meaningful names. It is common for a Web service to have attributes with irrelevant and meaningless names. For example, one input might be named "in1". In a situation like this it is impossible for an automatic matcher of any type to successfully match this attribute with a concept from an ontology. Likewise, a machine learning approach that relies on similar words within the same domain would be hindered by a service containing too many irrelevant words. One solution to this could be to match the extracted word stems to a dictionary such as WordNet and only use terms for classification that appear in the dictionary and are longer than some predefined threshold.

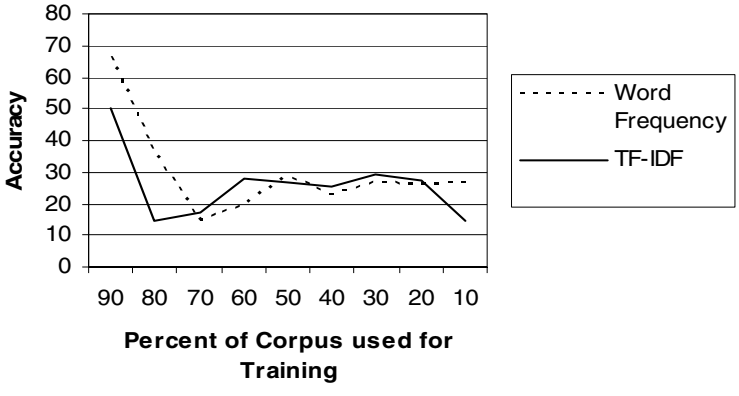

Fig. 3. Classification Statistics for Several Domains

6.2 Performance

As mentioned in section 4, the machine learning approach is significantly faster than the classification approach used by MWSAF because it is not necessary to match every concept of WSDL to every concept of each ontology. This approach does not require the use of an ontology for classification at all. This eliminates the risk of having wrongly categorized services because the ontologies are not comprehensive enough to contain all of the words for matches. The most timely part of our approach is the training phase, however, the training of the classifier itself does not take time. The time used for word extraction and the calculation of word frequencies is minimal. The learner is trained once and the actual categorization of the Web service takes only milliseconds.

6.3 Comparison of MWSAF Classification and Machine Learning Classification

Since the MWSAF schema matching technique has not been evaluated with more than 2 domain ontologies, it is hard to do a comparison in terms of accuracy. In our test case for 2 domains, the classifier's accuracy was on average comparable to the schema matching. Training on a large corpus of documents increased the accuracy radically above the schema matching. In the more realistic case of 10 domains, training with a large corpus produced good results around 68% correctly identified domains, decreasing the training set size let the accuracy drop but stabilize at around 28%. Two conclusions can be drawn from this: a) The more web services are classified, the more accurate will the classification of new services be. b) the approach is scalable. Even with a low percentage of training data the classification is stable.

Overall, the machine learning approach for classification has the potential to be a major enhancement for MWSAF. It is always much faster because it does not require any matching to an ontology. It relies only on the training data obtained from classified WSDL files. Domain prediction is achieved almost instantly. We believe that this approach has the potential to be a much more accurate classifier, and plan to investigate this with future research.

7 Related and Future Work

There is work currently being done in the area of machine learning for the classification and annotation of Web services. Assam [4] is a tool for semi-automatically annotating Web services. The tool provides the user with assistance and suggestions for matches based on the results of machine learning techniques. Assam implements an ensemble learning approach to improves the results [3]. They also researched and tested the approach of using previously annotated WSDL files as training data. Not only did this dramatically improve the results for classification, but it also enabled them to annotate operations, inputs and outputs [3]. The approach presented here improves the first step of MWSAF's schema matching – finding the appropriate ontologies to annotate the Web services. MWSAF's next step is to also annotate operations, inputs, and outputs, for which it still uses a schema matching technique. This is much faster now, because MWSAF only has to compare concepts of a single ontology with method and attribute descriptions in the WSDL file. Extending our machine learning technique to replace MWSAF's matching algorithm for specific concepts to be used for annotation would further enhance the performance of the tool.

Future work includes improving the accuracy of our current machine learning approach so that the averages are higher with less training data. We will investigate ensemble learning techniques. We must also provide a solution to situations where there is large overlap between the significant terms of multiple domains, as explained in section 5.1. Since the approach was very successful for two domains, we will investigate the potential for it to perform better for many domains.

8 Conclusion

As Web services become more widely used it is imperative that semantic metadata is added to Web services. This will facilitate the discovery, composition and interoperation of these services. Since manual annotation is far too expensive and time consuming to be an option, there is a great need for the ability to annotate them automatically or semi-automatically. MWSAF is an effective tool for annotation but is limited by its slow exhaustive schema matching technique. We proposed a machine learning approach that uses a Bayesian classifier to predict the domain of a Web service based on previous training data. This approach is significantly faster than matching every concept of the WSDL to every concept of each ontology in the store. Replacing the classification technique currently used by MWSAF with the naïve Bayesian classifier described in this paper significantly enhances the speed and performance of the tool. How successful machine learning techniques in our annotation framework can be will be shown by our future work of also matching individual classes and properties to methods and attributes.

References

1. Patil, A., Oundhakar, S., Sheth, A., Verma, K.: METEOR-S Web service Annotation Framework., Proceeding of the World Wide Web Conference, (2004)
2. Witten, I., Frank, E.: Data Mining: Practical machine learning tools with Java implementations. Morgan Kaufmann, San Francisco
3. Heß A., Kushmerick N., Machine Learning for Annotating Semantic Web Services. University College Dublin, Ireland.
4. Heß, A., Johnston, E., Kushmerick, N. ASSAM: A Tool for Semi-Automatically Annotating Semantic Web Services, Computer Science Department, University College Dublin, Ireland, ISWC04, (2004)
5. Sivashanmugam, K., Verma, K., Sheth, A., Miller, J., Adding Semantics to Web Services Standards. LSDIS Lab, University of Georgia. ICWS03 (2003)
6. Sheth, A., Ramakrishnan, C., Semantic (Web) Technology In Action Ontology Driven Information Systems for Search, Integration and Analysis. Data Engineering special issue on the Semantic Web, (2003)
7. Porter, M., An algorithm for Suffix Stripping, Program – Automated Library and Information Systems, 14(3):130-137, (1980)
8. Salton, G., Buckley, C., Term Weighting Approaches in Automatic Text Retrieval, Information Processing and Management, Vol. 24, No.5, P513, (1998)