To appear in the proceedings of the SPIE Defense and Security Symposium, Orlando, April 9-13, 2007

An Evaluation of Case-Based Classification to Support Automated Web Service Discovery and Brokering

Roy Ladner[a], Elizabeth Warner[a], Fred Petry[a],
Kalyan Moy Gupta[b], David W. Aha,[c]
[a]Naval Research Laboratory, Stennis Space Center, MS
[b]Knexus Research Corp., Springfield, VA,
[c]Naval Research Laboratory, Washington, DC

ABSTRACT

In this paper we evaluate the use of case-based classification to resolve a number of questions related to information sharing in the context of an Integrated Web Services Brokering System (IWB). We are developing the IWB to independently decompose and analyze ad hoc Web Services interface descriptions in order to identify Web Services of interest. Our approach is to have the IWB cache information about each service in order to support an autonomous mediation process. In this mediation process, the IWB independently matches the user's data request to the correct method within the appropriate web service, translates the user's request to the correct syntax and structure of the Web Service request, dynamically invokes the method on the service, and translates the Web Service response. We use case-based classification as a means of automating the IWB's analysis of relevant services and operations. Case-based classification retrieves and reuses decisions based on training data. We use sample Web Service Description Language (WSDL) files and schema from actual web services as training data in our approach and do not require the service to pre-deploy an OWL-S ontology. We present our evaluation of this approach and performance ratings in the context of meteorological and oceanographic (METOC) Web Services as it relates to the IWB.

Keywords: Web Services, Ontology, Case-based Classification, METOC, Service Oriented Architecture, Automated Web, Semantic Web

1. INTRODUCTION

In this paper we evaluate the use of case-based classification to resolve a number of questions related to information sharing in the context of an Integrated Web Services Brokering System (IWB). We are developing the IWB to independently decompose and analyze ad hoc Web Services interface descriptions in order to identify Web Services of interest. Our goal is for a system that can make reasoned judgments on Web Services interfaces. Our approach is to have the IWB cache information about each service in order to support an autonomous mediation process. In this mediation process, the IWB independently matches the user's data request to the correct method within the appropriate web service, translates the user's request to the correct syntax and structure of the Web Service request, dynamically invokes the method on the service, and translates the Web Service response. This approach avoids the need to manually categorize web services that are found and manually resolve structural and syntactic differences inherent in varying Web Services interfaces.

Our approach to the IWB does not assume that shared ontologies have been adopted or that Web Services descriptions have been semantically annotated as with OWL-S in order to support service discovery and integration by a data broker [1]. This semantic annotation with OWL-S is intended to provide a means for prospective clients to identify whether the service's capabilities match the client's requirements [2]. In contrast to annotating Web Services descriptions, we incorporate ontologies into the development of the IWB [3]. In addition we use case-based classification for specific IWB subtasks related to the discovery, integration and mediation processes. There are aspects of this approach that may prove beneficial in this area. Case-based classification retrieves and reuses decisions based on training data. We use

sample Web Service Description Language (WSDL) files and schema from actual web services as training data in our approach. When new, previously unanticipated cases are obtained, the classifier may be retrained. No new software programming is needed.

The remainder of this paper is organized as follows. First, we describe the architecture of the IWB. We then explain our approach for using a case-based classification. We present our evaluation of this approach and performance ratings in the context of meteorological and oceanographic (METOC) Web Services as it relates to the IWB. We close with a discussion of future research goals.

2. IWB ARCHITECTURE

This section briefly describes the architectural features of an automated brokering system for the specific instance of our prototype application context, the METOC Web Services domain. Our interest in the IWB is for a brokering system that creates a dynamic knowledge base from Web Service interface specifications that it discovers on the fly and that uses the dynamic knowledge base to assist it with mediating requests to data providers that have ad-hoc interfaces.

The functional components of the IWB are shown in Figure 1. The IWB can begin mediating user requests once it has discovered Web Services and begun populating the Dynamic Index portion of its Knowledge Base. The Web Patroller component discovers Web Services interface specifications by searching known registries and the Web. The Case-Based Broker Learner processes the interface specifications and provides outputs to the Method Indexer. The Method Indexer component of the IWB uses these outputs to add to the Dynamic Index only those newly discovered Web Services operations that are relevant to retrieval of data of interest.

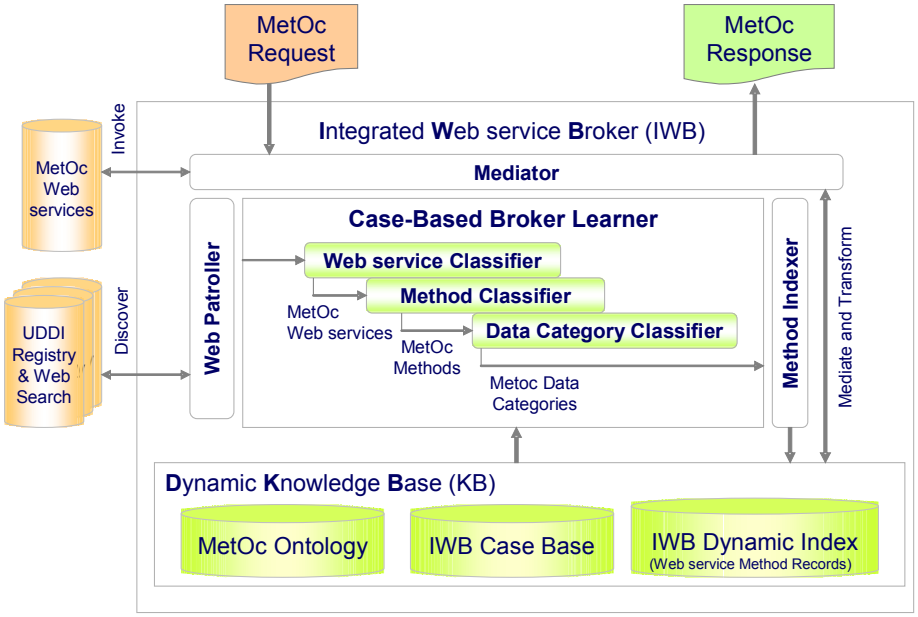

Fig. 1. The IWB Architecture.

Three applications of classifier technology are found in the Case-Based Broker Learner. These are Web Services identification, Web Services method identification, and data category or type specialization identification. The IWB employs these in serial form. That is, output from the first the first classifier supplies the basis for new inputs to the

second. These classifiers are trained on METOC and non-METOC Web Services examples contained in the Case Base portion of the Knowledge Base.

The Web Services classifier identifies whether the web service's interface specification is related to the data of interest, in this case METOC. Any categories previously assigned to a web service by a registry are ignored. For those interfaces that are identified as being of interest, the Web Services method classifier examines each of the methods or operations defined in the interface specification and identifies those methods that return data of interest. For example, a web service may define two operations, one that supplies weather data and another that provides a list of airport codes. The classifier would identify the former as relevant but not the latter. The data category classifier operates on those methods that the second classifier identified as METOC relevant and assigns labels related to specialized type information for the data returned from the method. Specialized type information will vary by data domain. Examples of such specializations for METOC may include whether the data supplied by the web service is from an actual observation or forms part of a forecaster's analysis. The information obtained from this serialized use of classifiers is used to populate the IWB's Dynamic Index.

After the IWB's Dynamic Index is initialized it is ready to process user requests to the appropriate web service or multiple services. The Mediator is the component of the IWB that provides the necessary transforms for this to occur. Clients submit data requests to the Mediator in an IWB XML format. The Mediator uses mappings that were created by the Method Indexer to translate the client request into a candidate web service format specified in the web service's interface and submits the request to the web service provider. As the recipient web service sends the data response back to the Mediator, the Mediator transforms the web service response to the client's expected format and forwards it to the client. This is the inverse of the request mapping process. For more detailed explanation of the Dynamic Index and the mediation/translation processes, the interested reader is referred to [4].

3. CASE-BASED CLASSIFICATION

In this section, we generally describe case-based classification used in our evaluation for particular IWB processes. A more thorough discussion is provided in [5].

The classification process is depicted in Figure 2. Case-based classification reuses the classifications of previously classified objects (i.e., *cases*) that have characteristics similar to the new object. To assess the similarity of one case with another, the classifier uses a similarity metric. The cases that are the most similar to the unclassified object are called its nearest neighbors. The classifier considers the classes of the k nearest neighbors from the case base when predicting the class label of an unclassified object. A metric is used by the classifier to select a subset of attributes from the object that have a potential to improve classification performance. Each nearest neighbor votes on the decisions based on its classification with each vote being weighted by the similarity of the voting neighbor. The classification label with the most weighted votes is assigned as the class of the new case. If the class assigned to the new case is the same as its actual class, then this is counted as a correct classification. Classifier performance is measured by the percentage of cases classified correctly.

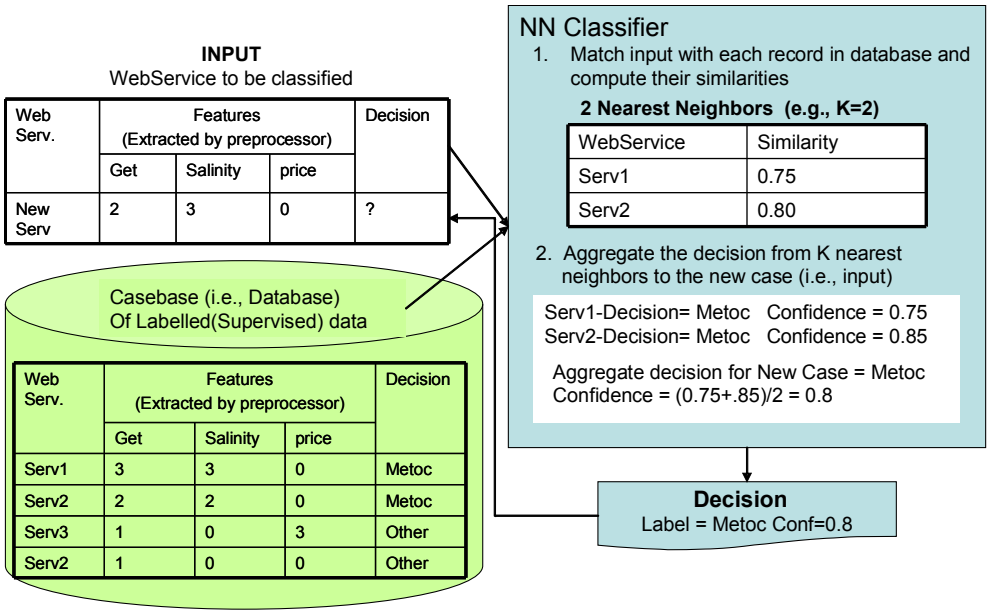

Fig. 2. High Level View of CBR Process Flow.

For our exercise, we employed information gain feature selection. Information gain is an attribute selection metric used to select a subset of attributes with a potential to improve classification performance We used a genetic algorithm to search for the values of the parameters *k* (nearest neighbors) and the number of attributes used in the information gain feature selection algorithm. We used classification accuracy as the fitness function for the genetic algorithm. This provided *k* equal to 5 and the number of attributes equal to 523 out of maximum possible 1790 attributes from the training set of Web Services interfaces.

4. CLASSIFICATION EVALUATION

We use a leave-one-out cross-validation method to evaluate our classifiers' performance. In this methodology, we repeatedly remove one case from the data set for testing and use the remaining cases to train the classifier. The classification accuracy for each test case is recorded. This process of training and classification is repeated for each case in the set to determine the classifier's average classification accuracy. This approach was utilized separately for each of our three classifiers.

Our performance evaluation for each classifier uses the same set of 64 Web Services interfaces found on the Internet. These interfaces include one WSDL file per web service and any referenced or included schema. Our subject matter expert manually classified 26 of these schemas as METOC relevant. From the 26 METOC relevant Web Services interfaces, 74 method definitions were identified. Our subject matter expert manually classified 64 of these as operations that returned METOC data and two applicable data specializations. That is, the methods returned either observational data or a forecasters' analysis.

4.1 Web Service Classifier

We initially want to identify those Web Services that pertain to METOC data, our data of interest. Our initial evaluation showed average precision of 93.75% [5]. We have more recently performed more detailed evaluation of this web service

classification task employing alternate feature selection and classification methodologies. The results of this evaluation are provided in their respective sections below.

4.2 Feature Selection Performance

With the potential complexity found in many Web Services interface definition, many thousands of features can be generated for classifier training. This may introduce a serious computational challenge and can also adversely affect classification performance by introducing noisy and irrelevant features. The feature "http" for example may appear in all cases and provide no useful discriminatory information. Feature selection is a means used to select a subset of features with a potential to improve classification performance. Feature selection metrics include mutual information, information gain, document frequency [6], and rough set methods [7].

In this section, we compare average precision obtained for web service classification using information gain to precision obtained when using rough set feature selection. When using information gain, we limit the number of features or attributes to 523 out of a possible 1790 as described above. This is a considerable reduction. For rough set, in contrast, we used an adaptation of Johnson's reduct algorithm which sequentially selects features by finding those that are most discernible for a given decision feature [7]. The overall results of the comparison are given in Table 1.

Table 1: Comparison of CBR Feature Section Methods for Web Service Classification

Feature Section Method	Precision	Number of Attributes
Information Gain	93.75%	523
Rough Set	65.08%	8

Due to the small case base, Johnson's reduct procedure selected only 8 attributes. Average precision of 65.08% is significantly lower than that previously obtained using information gain (93.75%) for feature selection. Class wise average precision and recall is given in Table 2.

Table 2: Rough Set Feature Selection Classwise Average Recall & Precision

Class	Correct	False Pos.	False Neg.	Recall	Precision
MetOc	16	3	9	0.64	0.84
Non MetOc	25	7	13	0.66	0.78

The algorithm results in many ties at each iteration and breaks ties by picking the first one in memory. Even with sampling, it picks only 8 features compared to information gain's 523 (where we selected the threshold). Considering only 8 features compared to 523, the classification was quite reasonable. For a fair comparison with Johnson's reduct, to examine which method selects better features, we restricted information gain to select only 8 features. The performance of top 8 features selected by IG is shown in Table 3.

Table 3: Information Gain Feature Selection Classwise Average Recall & Precision

Class	Correct	False Pos.	False Neg.	Recall	Precision
MetOc	21	2	4	0.84	0.91
Non MetOc	0	1	38	0.0	0.0

The average precision for IG is 33.30%, which is significantly lower than rough set approach (65%). This confirms our observation that a rough set approach picks much more informative features than information gain but terminates early in sparse high dimensional data sets with relatively few cases. To explore further, we examine the performance of information gain by varying the number of features (See Figure 3). The average precision drops significantly as the number of features decreases to 10 or lower.

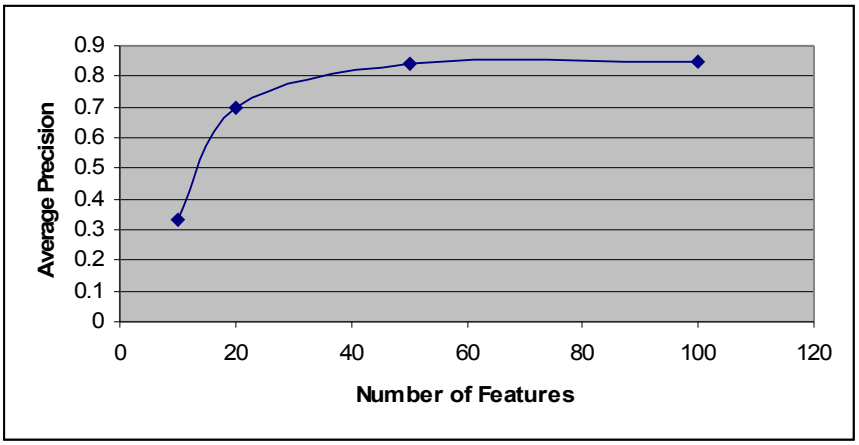

Fig. 3. Illustration of Information Gain Precision versus Number of Features.

4.3 Decision Tree

In this section we compare case-based classification to decision tree for the web service classification task. For this comparison, we implemented the C.4.5 version of the decision tree induction algorithm that uses information gain ratio as the basis for creating the branches [9]. The average precision of these two methods are shown in Table 4 and the classwise average precision and recall of decision tree is given in Table 5. Decision tree provided average accuracy of 84%, somewhat lower than case-based classification. Performance of decision tree may prove higher when working with a larger data set.

Table 4: Comparison of CBR to Decision Tree for Web Service Classification

	CBR	Decision Tree
Precision	93.75%	84.13%

Table 5: Decision Tree Classwise Average Recall & Precision

Class	Correct	False Pos.	False Neg.	Recall	Precision
MetOc	20	5	5	0.80	0.80
Non MetOc	33	5	5	0.87	0.87

4.4 Web Service Method Classifier

A WSDL may define operations that provide weather data along with ones that supply associated information such as airport codes. As explained above, the object of the web service method classifier is to distinguish these two types of operations. The former are relevant to the IWB and should be included by the Method Indexer in the broker's Dynamic Index, while the latter are not.

Two evaluations of Web Service Method Classifier were performed. One used only the method name in training and testing. The other employed the method's context in addition to its name. Method context refers to the terms associated

with the inputs to the method and outputs from the method. These terms are extracted from the structures of the WSDL and any associated schema.

Performance results are shown in Tables 6, 7 and 8. Average precision with method context was found to be 91.89% and without context to be slightly higher at 93.24%. Using context, the classifier correctly classified 63 of 64 possible operations as METOC relevant but only correctly classified five of ten possible non-METOC operations. Without context, results were similar for METOC operations in that 63 of 64 operations were correctly classified. For non-METOC operations, 6 of 10 were correctly classified when context was not used.

Table 6: Web Service Method Classifier	
Method Context Included	Precision
Yes	91.89%
No	93.24%

Table 7: Web Service Method Classifier Classwise Averages with Context					
Class	Correct	False Pos.	False Neg.	Recall	Precision
METOC	63	5	1	0.98	0.93
Non METOC	5	1	5	0.50	0.83

Table 8: Web Service Method Classifier Classwise Averages without Context					
Class	Correct	False Pos.	False Neg.	Recall	Precision
METOC	63	4	1	0.98	0.94
Non METOC	6	0	4	0.60	1.00

4.5 Data Category Classifier

The object of the data category classifier is to is to distinguish the data categories or specializations that pertain to each operation determined by the web service method classifier to be METOC relevant. For example, an operation may supply sea surface temperature data, which may derive from different sources such as observed measurement, a numerical prediction model or a narrative forecast report.

The data category classifier assigned two possible labels, observation or narrative report, as these were the possible specializations contained within our training set. As with the web service method classifier, the evaluation was performed separately with and without context using leave one out testing.

The overall precision of this classifier is given in Table 7. With context, the classifier showed average precision of 93.23%. This increased to 94.08% when method context was omitted.

Table 7: Data Category Classifier	
Method Context Included	Precision
Yes	92.37%
No	94.08%

With context, the classifier correctly assigned the label narrative report in 52 of 53 cases and correctly assigned the label observation in three of six cases. Without context, all narrative report cases were correctly classified, while only three of six observation labels were correctly assigned. Table 8 provides average precision based on label.

Table 8: Data Category Classifier

	Precision by Label	
Method Context Included	Label: Observation	Label: Forecast Report
Yes	60%	98.11%
No	75%	100.00%

6. CONCLUSION AND FUTURE DIRECTION

We described an Integrated Web Services Brokering System, the IWB. The IWB is expected to independently decompose and analyze ad hoc Web Services interface descriptions in order to identify Web Services of interest, in this case METOC, and cache information about each service in order to support an autonomous mediation process. The cached information supports the IWB in its tasks to independently match the user's data request to the correct method within the appropriate web service, to translate the user's request to the correct syntax and structure of the Web Service request, to dynamically invoke the method on the service, and to translate the Web Service response. In this manner, the IWB avoids the need to manually categorize web services that are found and manually resolve structural and syntactic differences inherent in varying Web Services interfaces.

We reported the results of case-based classification used to automate many of the IWB's tasks. Previous work using classification approaches for analyzing Web Services interfaces has focused on multiple categories of services [8]. Our study has focused on a single category that is of use in many Naval applications, meteorological and oceanographic (METOC). We believe that current findings warrant additional work on use of domain relevant ontologies to boost classifier performance.

ACKNOWLEDGMENTS

The authors would like to thank the Naval Research Laboratory's Base Program, Program Element No. 0602435N for sponsoring this research.

REFERENCES

[1] Paolucci, M., Soudry, J., Srinivasan, N., and Sycara, K., "A Broker for OWL-S Web services," *Proceedings of the AAAI Spring Symposium on Semantic Web Services,* 562-567, 2004.

[2] W3C Member Submission (2004) OWL-S: Semantic Markup for Web Services, http://www.w3.org/Submission/OWL-S/.

[3] Ladner, R., Petry, F., Warner, E., and Gupta, K., "Web Services Enhanced Geographic Information Systems," Control and Cybernetics, 19, #2, pp 234-252, 2006.

[4] Sample, J.T., Ladner, R., Ioup, E., Petry, F., Warner, E., Shaw, K., McCreedy, F.P., Shulman, L., "Enhancing the US Navy's GIDB Portal with Web Services," IEEE Internet Computing, 10, #5, pp 53-60, 2006.

[5] Ladner, R., Warner, E., Petry, F., Gupta, K.M., Moore, P., Aha, D.W., Shaw, K., "Case-Based Classification Alternatives to Ontologies for Automated Web Service Discovery and Integration," *Proceedings SPIE Defense & Security Symposium*, Vol. 6201, pp. 620117-1 – 620117-8, 2006.

[6] Yang, Y., and Pederson, J., "A comparative study of feature selection in text categorization," *Proceedings of the Fourteenth International Conference on Machine Learning*, Nashville, TN: Morgan Kaufmann, 412-420, 1997.

[7] Gupta K.M., Aha D.W., & Moore P.G., "Rough set feature selection algorithms for textual case-based classification," *M.G. Göker & T.R. Roth-Berghofer (Eds.) Proceedings of Eighth European Conference on Case-Based Reasoning ECCBR-06*, Ölündeniz, Turkey: Springer, 2006.

[8] Heß A and Kushmerick,N, "Machine Learning for Annotating Semantic Web Services," *Proceedings of the AAAI Spring Symposium*, Semantic Web Services, 341-346, 2004.

[9] Quinlan, J. R. C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, 1993.