Classification of Web Services using Fuzzy Classifiers with Feature Selection and Weighted Average Accuracy

V. Mohan Patro¹ and Manas Ranjan Patra²
Department of Computer Science, Berhampur University, Berhampur, Odisha, India
¹vmpatro@gmail.com, ²mrpatra12@gmail.com

ABSTRACT

Web services have become an innovative and accepted means of service delivery over the Internet. In recent years there has been astounding growth in the number of web services provisioned by businesses and corporate houses. In the presence of a plethora of web services, a service consumer faces the real challenge of making a right choice based on certain preferences. Therefore, it becomes necessary to classify a set of web services based on certain quality parameters in order to facilitate user choice of web services under different scenarios. Several classification techniques have been proposed by researchers to classify data sets in different application domains. In this work, we have employed three fuzzy classifiers, namely, Fuzzy Nearest Neighbor, Fuzzy Rough Nearest Neighbor, and Fuzzy Rough Ownership Nearest Neighbor to classify web services. We have used the standard QWS dataset for our experimentation. The accuracy of the classifiers has been computed with and without feature selection. In order to further improve classification accuracy, a Weighted Average Accuracy technique has been applied to the confusion matrix obtained after feature selection.

Keywords – Web services, Fuzzy Nearest Neighbor classifier, Fuzzy Rough Nearest Neighbor classifier, Fuzzy Rough Ownership Nearest Neighbor classifier, Weighted Average Accuracy.

1 Introduction

Web Services are emerging technologies that enable machine-to-machine communication and reuse of services over the Web. A Web Service is a software function provided at a network address and can support interoperable machine-to-machine interaction over the web. Different software systems often need to exchange data with each other, and a web service is a means of communication that allows two software systems to exchange data over the internet. With the increasing number of available Web services on the internet, Web service discovery becomes a challenging issue. It is time consuming to traverse the whole of the Internet with a view to find a Web service that matches one’s service requirements. To speed up service discovery, classification can be a useful approach. Researchers have applied different classification techniques to categorize web services based on a set of quality parameters.

Yuan-jie et al. in [1] applied automatic web service semantic annotation and use three classification methods, namely, Naïve Bayes, SVM and REPTree along with ensemble learning. They applied 10 cross-validations of Naïve Bayes, SVM, REPTree and AdaBoost on WSDL files. According to the experiment done on 951 WSDL files and 19 categories, the highest accuracy was 87.39%.

Web Service is an innovative mechanism for rendering services over diversified environment [2]. Efficient result has been taken from QWS dataset using weka tool in the experiment. The experiment results shown in the study are about classification accuracy obtained by J48 as 63%.
Authors in [3] developed various classification models based on intelligent techniques namely BPNN, PNN, GMDH, TreeNet, CART, SVM and J48 to predict the quality of a web service based on a number of QoS attributes. They observed that J48 outperforms other classifiers they used for accuracy calculation.

In [4], authors have shown how SVM is helpful in the classification of web services. They used the SVM (Support Vector Machine) text classification algorithm to classify the service documents based on a standard and widely used taxonomy with feature selection.

Mohanty et al. in [5] employed Naïve Bayes, Markov blanket and Tabu search techniques to classify web services dataset. They noted that the average accuracy of Naïve Bayes classifier is 85.62%, followed by Tabu search of 82.45% and Markov blanket of 81.36%.

In this paper, we have employed three classifiers, namely, Fuzzy Nearest Neighbor (FNN), Fuzzy Rough Nearest Neighbor (FRNN), and Fuzzy Rough Ownership Nearest Neighbor (FRONN) to classify web services dataset. The classification accuracies of the classifiers have been evaluated with and without feature selection. Next, a Weighted Average Accuracy algorithm (from our earlier work [6]) is applied to the confusion matrix obtained after feature selection in order to improve upon the results.

The rest of the paper is organized as follows: Section 2 describes the classifiers used, section 3 presents the experimental set up, section 4 analyzes the results, and section 5 concludes the paper.

2 Classification Techniques Used

2.1 Fuzzy Nearest Neighbors

The Fuzzy Nearest Neighbor (FNN) algorithm [7, 8] was introduced to classify test objects based on their similarity to a given number K of neighbors, and these neighbors’ membership degree to (crisp or fuzzy) class labels. For the purpose of (FNN), the extent \( C(y) \) to which an unclassified object \( y \) belongs to a class \( C \) is computed as:

\[
C(y) = \sum_{x \in N} R(x, y)C(x)
\]

where \( N \) is the set of object \( y \)’s K nearest neighbors, and \( R(x, y) \) is the \([0,1] \)-valued similarity of \( x \) and \( y \).

---

**The Fuzzy K-Nearest Neighbors Algorithm**

FNN \((X, C, y, K)\)

**Input:** \( X \), the training data set; \( C \), the set of decision classes; \( y \), the objects to be classified; \( K \), the number of nearest neighbors.

**begin**

\( N \leftarrow \) get Nearest Neighbors \((y, K)\)

for each \( C \in C \) do

\( C(y) = \sum_{x \in N} R(x, y)C(x) \)

**output:** \( \arg \max_{C \in C} C(y) \)

**end**

---

2.2 Fuzzy-Rough Nearest Neighbor

In Fuzzy-Rough Nearest Neighbor (FRNN) algorithm the nearest neighbors are used to construct the fuzzy lower and upper approximations of decision classes, and test instances are classified based on
their membership to these approximations. FRNN algorithm combines fuzzy-rough approximations with the classical FNN approach [7,8]. The rationale behind the algorithm is that the lower and upper approximation of a decision class, calculated by means of the nearest neighbors of a test object $y$, provides good clues to predict the membership of the test object to that class. The algorithm is dependent on the choice of a fuzzy tolerance relation $R$. Given the set of conditional attributes $A$, the fuzzy tolerance relation $R$ is defined by

$$R(x, y) = \min_{a \in A} R_a(x, y)$$

(2)

where $R_a(x, y)$ is the degree to which objects $x$ and $y$ are similar for attribute $a$. Here we choose

$$R_a(x, y) = 1 - \frac{|a(x) - a(y)|}{|a_{\max} - a_{\min}|}$$

(3)

If $(R \downarrow C)(y)$ is high, it reflects that all of $y$’s neighbors belong to $C$. A high value of $(R \uparrow C)$ means that at least one neighbor belongs to that class.

**The Fuzzy Rough Nearest Neighbors Algorithm:***

<table>
<thead>
<tr>
<th>FRNN $(X, C, y)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X$, the training data set; $C$, the set of decision classes; $y$, the object to be classified; begin</td>
</tr>
<tr>
<td>$N \leftarrow$ get Nearest Neighbors $(y, K)$</td>
</tr>
<tr>
<td>$\tau \leftarrow 0$, Class $\leftarrow \emptyset$</td>
</tr>
<tr>
<td>for each $C \in C$ do</td>
</tr>
<tr>
<td>if $(\nabla R(C)(y) + (R \uparrow C)(y))/2 \geq \tau$ then</td>
</tr>
<tr>
<td>$\tau \leftarrow (\nabla R(C)(y) + (R \uparrow C)(y))/2$</td>
</tr>
<tr>
<td>end</td>
</tr>
<tr>
<td>output Class</td>
</tr>
<tr>
<td>end</td>
</tr>
</tbody>
</table>

### 2.3 Fuzzy Rough Ownership Nearest Neighbors

Fuzzy-Rough ownership is an attempt to handle both “fuzzy uncertainty” and “rough uncertainty”[7, 8]. The fuzzy-rough ownership function $\tau_c$ of class $C$ is defined for an object $y$ as,

$$\tau_c(y) = \sum_{x \in X} \frac{R(x, y)C(x)}{IXI}$$

(4)

The fuzzy relation $R$ is determined by

$$R(x, y) = \exp(-\sum_{a \in A} K_a(a(y) - a(x))^2/(m - 1))$$

(5)

where, $m$ controls the weighting of the similarity and $K_a$ is a parameter that decides the bandwidth of the membership. $K_a$ is defined as

$$K_a = \frac{IXI}{2 \sum_{x \in X} ||a(y) - a(x)||^2/(m - 1)}$$

(6)

$\tau_c(y)$ is interpreted as the confidence with which $y$ can be classified to class $C$. The algorithm does not use fuzzy lower or upper approximations to determine class membership.
3 Experimentation

3.1 Data Set

The QWS (Quality of Web Service) dataset [9-11] consists of data from over 5000 web services out of which the public dataset consists of a random 364 web services. The service descriptions were collected using the Web Service Crawler Engine (WSCE) [12]. The majority of Web services were obtained from public sources on the Web including Universal Description, Discovery, and Integration (UDDI) registries, search engines, and service portals. The public dataset consists of 364 web services each with a set of nine Quality of Web Service (QWS) attributes that have been measured using commercial benchmark tools. WSRF is used to measure the quality ranking of a web service based on the nine quality parameters (1-9 in Table-1).

In Table 1, the service parameters 1-9 are used for computation of classification accuracy with respect to four “Service Classification” values, namely, “Platinum” (high quality), “Gold”, “Silver” and “Bronze” (low quality) equivalent to 1 through 4 respectively.

### Table 1: QWS Parameter description

<table>
<thead>
<tr>
<th>P-ID</th>
<th>Parameter Name</th>
<th>Description</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Response Time</td>
<td>Time taken to send a request and receive a response</td>
<td>ms</td>
</tr>
<tr>
<td>2</td>
<td>Availability</td>
<td>Number of successful invocations/total invocations</td>
<td>%</td>
</tr>
<tr>
<td>3</td>
<td>Throughput</td>
<td>Total Number of invocations for a given period of time</td>
<td>Invokes per second</td>
</tr>
<tr>
<td>4</td>
<td>Success ability</td>
<td>Number of responses / number of request messages</td>
<td>%</td>
</tr>
<tr>
<td>5</td>
<td>Reliability</td>
<td>Ratio of the number of error messages to total messages</td>
<td>%</td>
</tr>
<tr>
<td>6</td>
<td>Compliance</td>
<td>The extent to which a WSDL document follows WSDL specification</td>
<td>%</td>
</tr>
<tr>
<td>7</td>
<td>Best Practices</td>
<td>The extent to which a Web service follows WS-I Basic Profile</td>
<td>%</td>
</tr>
<tr>
<td>8</td>
<td>Latency</td>
<td>Time taken for the server to process a given request</td>
<td>ms</td>
</tr>
<tr>
<td>9</td>
<td>Documentation</td>
<td>Measure of documentation (i.e. description tags) in WSDL</td>
<td>%</td>
</tr>
<tr>
<td>10</td>
<td>WSRF</td>
<td>Web Service Relevancy Function: a rank for Web Service Quality</td>
<td>%</td>
</tr>
<tr>
<td>11</td>
<td>Service Classification</td>
<td>Levels representing service offering qualities (1 through 4)</td>
<td>Classifier</td>
</tr>
<tr>
<td>12</td>
<td>Service Name</td>
<td>Name of the Web service</td>
<td>None</td>
</tr>
<tr>
<td>13</td>
<td>WSDL Address</td>
<td>Location of the Web Service Definition Language (WSDL) file on the Web</td>
<td>None</td>
</tr>
</tbody>
</table>
3.2 WEKA Workbench

We have used the WEKA (Waikato Environment for Knowledge Analysis) machine learning platform [13] for our experimentation. The WEKA workbench consists of a collection of implemented popular learning schemes, which can be used for practical data mining and machine learning.

3.3 Cross-Validation

Cross-validation calculates the accuracy of the model by separating the data into two different subsets, namely, training set and validation set or testing set. The training set is used to perform the analysis and the validation set is used to validate the analysis. This validation process is continued k times to complete the k-fold cross validation procedure. We have used 10-fold cross-validation wherein the dataset is partitioned into 10 subsets, of which 9 subsets are used as the training fold and the 10th subset is used for testing. The process is repeated 10 times such that each subset is used as a test subset once. The estimated accuracy is the mean of the estimates for each of the classifiers.

3.4 Feature Selection (FS)

Feature selection refers to the process of selecting relevant attributes and reducing redundant and irrelevant attributes in the dataset to improve upon classification accuracy. Therefore, suitable attribute selection method for selecting the most prominent features (attributes) from the dataset is of paramount importance to enhance the performance of classification accuracy and reduce the computation time. In this study, we have applied two feature selection techniques, namely, Information Gain Attribute Evaluator and Gain Ratio Attribute Evaluator.

3.4.1 Information Gain (IG)

It evaluates the worth of an attribute by measuring the information gain with respect to a class. Information gain measure is used to determine how accurately a particular attribute classifies the training data. Information gain is based on the concept of entropy which is widely used in the Information theory domain.

Let node N represents the tuples of partition D. The attribute with the highest information gain is chosen as the splitting attribute for node N. This attribute minimizes the information needed to classify tuples in the resulting partitions and reflects the least randomness or impurity in these partitions [14].

The expected information needed to classify a tuple in D is given by

\[ \text{Info}(D) = -\sum_{i=1}^{m} p_i \log_2(p_i) \]  

where \( p_i \) is the probability that an arbitrary tuple in D belongs to class \( C_i \) and is estimated by \( |C_{i,D}|/|D| \). \( \text{Info}(D) \) is the average amount of information needed to identify the class label of a tuple in D.

\[ \text{Info}_A(D) = \sum_{j=1}^{r} \frac{|p_j|}{|D|} \times \text{Info}(D_j) \]  

The term \( \frac{|p_j|}{|D|} \) acts as the weight of the j-th partition. \( \text{Info}_A(D) \) is the expected information required to classify a tuple from D based on the partitioning by A. Information gain is defined as the difference between the original information requirement and new information requirement. That is

\[ \text{Gain}(A) = \text{Info}(D) - \text{Info}_A(D) \]
Using Information Gain Evaluation with Ranker Search on the QWS data set, top 4 attributes (WSRF, WSDL Address, Service Name and Reliability) are selected for classification.

### 3.4.2 Gain Ratio (GR)

It evaluates the worth of an attribute by measuring the gain ratio with respect to the class. It applies a kind of normalization to information gain using a “split information” value. The split information value represents the potential information generated by splitting the training data set \( D \) into \( v \) partitions corresponding to \( v \) outcomes on attribute \( A \), and is expressed as [14]:

\[
\text{SplitInfo}_A(D) = \sum_{j=1}^{v} \left| D_j \right| \times \log_2 \left( \frac{|D_j|}{|D|} \right)
\]

(10)

The gain ratio is defined as

\[
\text{GainRatio}(A) = \frac{\text{Gain}(A)}{\text{SplitInfo}(A)}
\]

(11)

The attribute with the maximum gain ratio is selected as the splitting attribute.

Using Gain Ratio Evaluation with Ranker Search on the QWS data set, top 5 attributes (WSRF, Throughput, Response Time, Reliability and WSDL Address) are selected for classification.

### 3.5 Confusion Matrix

Table 2: Confusion Matrix (2×2)

<table>
<thead>
<tr>
<th>Actual Class</th>
<th>Predicted Class</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( C_1 )</td>
</tr>
<tr>
<td>( C_1 )</td>
<td>True positive</td>
</tr>
<tr>
<td>( C_2 )</td>
<td>False positive</td>
</tr>
</tbody>
</table>

\( C_1 \) – particular class \( C_2 \) – different class

TP – True positive (TP) - The number of instances correctly classified as \( C_1 \)
TN – True negative (TN) - The number of instances correctly classified as \( C_2 \)
FP – False positive (FP) - The number of instances incorrectly classified as \( C_1 \) (actually \( C_2 \))
FN – False negative (FN) - The number of instances incorrectly classified as \( C_2 \) (actually \( C_1 \))

Using the above the following performance parameters are computed:

TP rate (TPR, Sensitivity, Recall) = \( \frac{\text{TP}}{\text{TP} + \text{FN}} \)
Positive Predictive Value (PPV, Precision) = \( \frac{\text{TP}}{\text{TP} + \text{FP}} \)
False Discovery Rate (FDR) = \( \frac{\text{FP}}{\text{FP} + \text{TP}} \)
FP Rate (FPR, False Alarm Rate (FAR), Fall-out) = \( \frac{\text{FP}}{\text{FP} + \text{TN}} \)
TN Rate (TNR, Specificity (SPC)) = \( \frac{\text{TN}}{\text{TN} + \text{FP}} \)
Negative Predictive Value (NPV) = \( \frac{\text{TN}}{\text{TN} + \text{FN}} \)
False Omission Rate (FOR) = \( \frac{\text{FN}}{\text{FN} + \text{TN}} \)
FN Rate (FNR) = \( \frac{\text{FN}}{\text{FN} + \text{TP}} \)
Accuracy (ACC) = \( \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FN} + \text{TN} + \text{FP}} \)
F-Value = \( \frac{(2 \times \text{Precision} \times \text{Recall})}{(\text{Precision} + \text{Recall})} \)

### 4 Result and Discussion

Here, we analyze the performance of three classification techniques, viz. Fuzzy Nearest Neighbor (FNN), Fuzzy Rough Nearest Neighbor (FRNN) and Fuzzy Rough Ownership Nearest Neighbor (FRONN) along with two feature selection techniques, namely Information Gain (IG) and Gain Ratio (GR). Performance is also observed with the application of weighted Average Accuracy (WAA) [6]. The classifiers are tested using 10-fold cross validation.
Classification accuracy values are recorded in table 3. Classification accuracy values obtained with feature selection give better result than that obtained without feature selection. Accuracy with the use of information gain as feature selection is more than that of gain ratio. From table 3 and fig 1 it is clear that Fuzzy Rough Ownership NN classification technique provides better accuracy as compared to other techniques and the value for Information Gain feature selection is best.

After applying weighted average accuracy, it is observed that Fuzzy Rough Ownership NN classification technique provides better accuracy as compared to other techniques and the value for Information Gain feature selection is again the best (Table 4 and Fig 2).

Comparing the results obtained in table 3 with that of table 4, we found that the weighted average accuracy values are better in all cases.
Table 5 shows values of precision, recall and f-value for all classifiers with and without feature selection. It is noticed that in all cases Fuzzy Rough Ownership NN classifier produces the best performance.

![Graphical representation of Precision, Recall and F-value for 3 classifiers](image)

**Figure 3: Graphical representation of Precision, Recall and F-value for 3 classifiers**

5 Conclusion

To enhance the classification accuracy several approaches have been adopted by researchers. Here, we applied Fuzzy Nearest Neighbor, Fuzzy Rough Nearest Neighbor and Fuzzy Rough Ownership Nearest Neighbor techniques without feature selection. Next, the same techniques are used with feature selections and improvement in classification accuracy is observed. Lastly, weighted average accuracy algorithm is used and further improvement is obtained. In all cases i.e. for accuracy, precision, recall, f-value the Fuzzy Rough Ownership Nearest Neighbor classifier and Information Gain feature selection produced better performance.

**ACKNOWLEDGEMENTS:**

We would like to thank Dr. E. Al-Masri and Dr. Q.H. Mahmoud for providing us the QWS dataset, which is used in our experiments.

**REFERENCE**


[13]. www.cs.waikato.ac.nz/ml/weka/ last accessed on 14/11/14