Bus Inf Syst Eng 58(4):249–259 (2016)
DOI 10.1007/s12599-015-0407-z

RESEARCH PAPER

A Majority Vote Based Classifier Ensemble for Web Service Classification

Usman Qamar · Rozina Niza · Saba Bashir · Farhan Hassan Khan

Received: 11 January 2014 / Accepted: 30 May 2015 / Published online: 4 November 2015

Abstract Service oriented architecture is a glue that allows web applications to work in collaboration. It has become a driving force for the service-oriented computing (SOC) paradigm. In heterogeneous environments the SOC paradigm uses web services as the basic building block to support low costs as well as easy and rapid composition of distributed applications. A web service exposes its interfaces using the Web Service Description Language (WSDL). A central repository called universal description, discovery and integration (UDDI) is used by service providers to publish and register their web services. UDDI registries are used by web service consumers to locate the web services they require and metadata associated with them. Manually analyzing WSDL documents is the best approach, but also most expensive. Work has been done on employing various approaches to automate the classification of web services. However, previous research has focused on using a single technique for classification. This research paper focuses on the classification of web services using a majority vote based classifier ensemble technique. The ensemble model overcomes the limitations of conventional techniques by employing the ensemble of three heterogeneous classifiers: Naïve Bayes, decision tree (J48), and Support Vector Machines. We applied tenfold cross-validation to test the efficiency of the model on a publicly available dataset consisting of 3738 real world web services categorized into 5 fields, which yielded an average accuracy of 92 %. The high accuracy is owed to two main factors, i.e., enhanced pre-processing with focused feature selection, and majority based ensemble classification.

Keywords Ensemble · Service oriented architecture (SOA) · WSDL documents · SVM · Naïve Bayes · J48 · Web services

1 Introduction

There are two types of web services, semantic and non-semantic ones (Huang et al. 2013). Semantic web services are described by using Semantic Web Services Ontology (SWSO). SWSO is specified by the use of Semantic Web Service Language (SWSL) (Liu et al. 2014). Non-semantic web services are described by an XML based language called Web Service Description Language (WSDL) (Malki et al. 2015). Non-semantic web services have recently gained more popularity in the software development industry. This is mainly because WSDL is independent of platform, protocol and language. This research targets non-semantic web services.

A central repository called universal description, discovery and integration (UDDI) (Kouki et al. 2014) is used by service providers to publish and register their web services utilizing WSDL files.

Accepted after three revisions by Prof. Dr. Suhl.

Dr. U. Qamar (✉) · R. Niza · S. Bashir
F. H. Khan
Department of Computer Engineering, College of Electrical and Mechanical Engineering (E&ME), National University of Sciences and Technology (NUST), Sector H-12, Islamabad, Pakistan
e-mail: usmanq@ceme.nust.edu.pk

R. Niza
e-mail: rozina.nisa@ceme.nust.edu.pk

S. Bashir
e-mail: saba.bashir@ceme.nust.edu.pk

F. H. Khan
e-mail: farhan.hassan@ceme.nust.edu.pk

A WSDL document specifies:

- details of operations required to invoke and consume the web service,
- its physical location,
- binding information of several transport formats as well as protocols between web service and web service requester.

Figure 1 shows the relationship between different sections of a WSDL file. As it can be seen, definitions in the Type section are used by Messages section, and definitions in the messages section are used by PortTypes. The PortTypes section is referred by the Binding section and in turn it is referred by the Services section. Operation elements are contained by PortTypes. Binding sections and port elements are contained by the Services section.

A description of WSDL features is given below (Elgedawy 2014):

- *Services*: A service element describes the name of the service and aggregates multiple service ports.
- *Messages*: Messages specify data being communicated between consumers and providers of a web service. There are two types of messages supported by a web service, input and output messages. Web service parameters are described by input messages, and data returned by the web service is described by output messages. Each message consists of zero or more part elements. Each part element refers to input/output parameters of a web service operation.
- *PortTypes*: A web service can have multiple ports. Ports refer to web services, end points and bindings define the transport protocol for these ports. A port type element contains a set of operations supported by the web service. Each operation contains input/output parameters.
- *Types*: This element describes language and machine independent data containers for message exchange. In short, data types used by a web service are defined in a Types element. W3C XML Schema specification is used by WSDL to define data types. The Types element is not used if a web service uses a simple design in types such as integers and strings.
- *Bindings*: Binding components provide details of data format and provide transport protocols for a portType operation. It is possible to use multiple transport protocols such as HTTP GET, HTTP POST, or SOAP for binding. Multiple bindings can be specified for a single portType.
- *Documentation*: The Documentation element provides a textual description of the web service functionality. It is not specified in all WSDL documents.

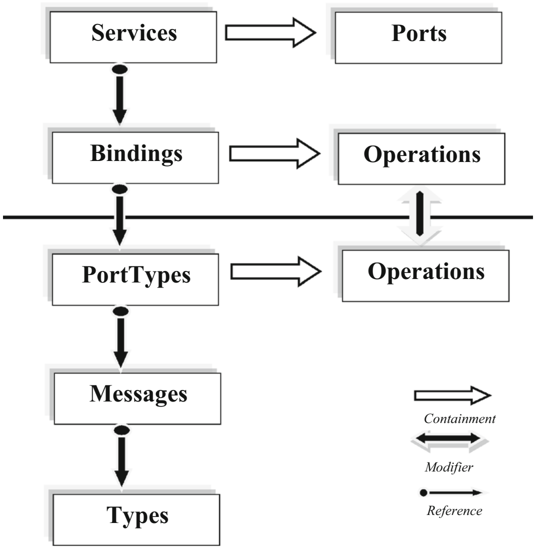

Fig. 1 Block diagram illustrating the relationship between WSDL sections

1.1 Motivation

The majority of service providers publish their web services using their own websites instead of public registries or service brokers. Therefore, there is an increasing trend to search web services through search engines. Although the search engines provide keyword or query based web service search facilities, the growing number of web services on the Web raises the challenging problem of locating the desired web services. They partially match search queries with terms in web service names, locations, and other attributes specified in the WSDL file. These search queries do not capture ultimate semantics of web services. Users must be aware of correct and concise keywords to retrieve the most relevant set of web services. For example, if a user is searching for ZipCode, WSDL files containing the keywords postal code and zip will not be returned in the search result.

In recent years, text mining has gained a lot of attention due to the increasing need for managing vast amounts of information in text documents (Liu et al. 2014). Semantic text analysis of WSDL files can help determining the functionality and capability of web services. Plebani and Pernici (2009) adopted SAWSDL (Semantic Annotation for WSDL) for annotating WSDL documents with enriched semantics. Di Martino (2009) presented a schema match making an approach to discover semantic web services using WSDL descriptions, Web Service Modeling Ontology, Web Ontology Language, and Semantic Web Services Framework. But manually annotating large sets of available web services is not a feasible task.

However, semantic information of a web service may be extracted from its WSDL file. Web service semantics can be inferred by mining service description, messages, operations and schemas in the WSDL file. Ping Bai (Bai and Li 2009) proposed an improved naïve Bayesian classification algorithm for classifying web text. Marcello Bruno (Bruno et al. 2005) suggested an automated web service classification technique to map services to specific domains using a support vector machine (SVM). Liu and Wong (2009) proposed a web service clustering approach by text mining web service description features such as WSDL contents, web service context, host name, and service name. However, these techniques use only a single classification technique that has shown acceptable levels of accuracy for web service classification. There is still room for improvement. One way to overcome the limitations of a single classifier is to use an ensemble model. The mechanism of ensemble classifiers is given in Fig. 2. The basic idea behind ensemble classifiers is to weigh several individual classifiers and then combine them to obtain the result which outperforms every individual classifier. The classification performance and prediction accuracy of an ensemble classifier is higher than that of single classifiers (Rokach et al. 2014).

The aim of this research is to improve the discovery of web services by proposing an ensemble model for web services classification. The main aim of the ensemble model is to increase the accuracy of classification.

Therefore, our main contributions are as follows:

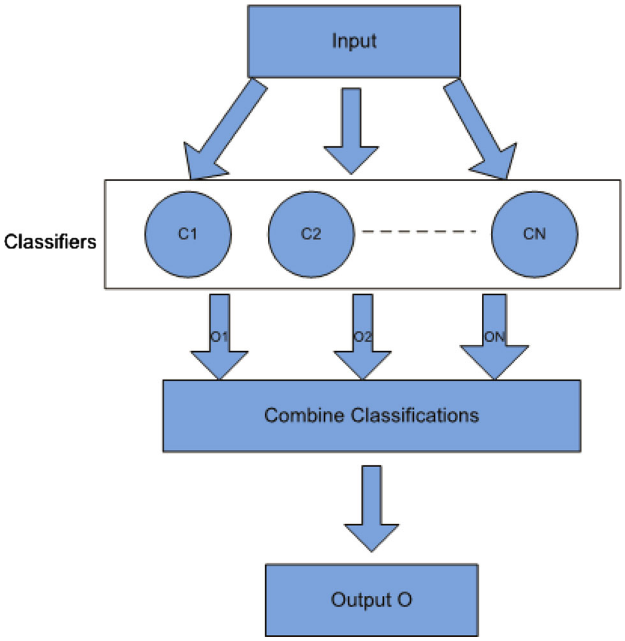

Fig. 2 Ensemble classifier obtained by combining several individual classifiers

- We present an approach that extracts contents (specific information such as service name, service documentation, WSDL messages, WSDL ports, and WSDL schema) from WSDL documents.
- These features are then used in our framework which is a novel combination of heterogeneous classifiers that perform classification. Naïve Bayes, decision tree based on Gain Ratio, and support vector machine algorithms are used as base classifiers. However, the ensemble model can be constructed using any set of homogeneous classifiers, heterogeneous classifiers, or a combination of both.
- We have used a majority vote based ensemble technique to combine the results of individual classifiers. The ensemble model achieves high classification and prediction accuracy by improving the precision and recall which in turn improves the f-measure and accuracy to a reasonable extent as compared to conventional individual models.
- We compare our approach with existing classification and clustering techniques to prove the superiority of our technique.

The remaining paper is organized in the following order. Section 2 discusses the related work. We present the design and methodology of our classification approach in Sect. 3. Section 4 discusses results and a detailed comparison of our approach with existing techniques. Finally, Sect. 5 concludes this paper.

2 Related Work

Classifying documents into a predefined set of categories is the task of text classification (Manning et al. 2008). More formally, text classification classifies each document *Di* in a set of documents {D1, D2, D3, D4, D5,…,Dn} to a category C*i* in set of categories {C1, C2,…,Ck}.

The Naïve Bayesian classifier is often used in automatic text categorization because of its effectiveness and simplicity (Bai and Li 2009). Bai and Li (2009) have proposed an improved Naïve Bayesian classification algorithm for classifying web text. In a Naïve Bayesian classifier all terms in a text are equally important, but Bai and Li suggested that terms in each title are more significant. This approach has improved the precision of text classification results.

Different web service classification and clustering techniques have been proposed based on mining the textual details in WSDL documents. Saha et al. (2008) proposed a two-step process of web service classification by using a Tensor space model and rough set based ensemble classifier. However, only three features from WSDL files are extracted for classification, namely service name, operation names, and input/output names, as well as their descriptions. They do not consider the WSDL Schema and WSDL Messages which reveal important information about the functionality of the web service.

Katakis et al. (2009) performed automated classification of web services taking into account both the textual description and the semantic annotations of OWL-S. They presented their results by applying machine learning algorithms to the textual and semantic descriptions, and proposed methods for increasing the overall classification accuracy through an extended feature set.

Bruno et al. (2005) proposed an automated web service classification technique to map services to specific domains. This approach also identifies key concepts in WSDL documents and builds a network of relationships between different web service annotations. In this approach, web service classification is performed by the State Vector Machine algorithm using web service documentation and user queries as classification features.

Zhuang et al. (2005) proposed a framework for the classification of web services. Each web service has its own WSDL file which allows the definition of services by providing elements such as name and description of web service, and information on operations and their inputs/outputs. Liu and Wong (2009) proposed a web service clustering approach by text mining web service description features such as WSDL contents, web service context, host name, and service name. Elgazzar et al. (2010) modified the Liu and Wong approach by selecting a different set of web service features. These features include WSDL contents, types, messages, ports, and web service name.

Table 1 summarizes WSDL features used by each web service classification/clustering approach.

Although extensive research has been conducted in the area of web service classification, there is still room for improvement to achieve higher levels of prediction accuracy. As already stated, one way to overcome the limitations of a single classifier is to use an ensemble model. An ensemble model is a multi-classifier combination model that is precise in its decision because the same problem is solved by different trained classifiers and reduces the variance of error estimation.

3 Research Methodology

The approach in this research uses WSDL features to classify web services using the majority vote based ensemble model. The framework can be divided into two main modules:

(a) WSDL feature extraction and pre-processing, and
(b) Ensemble based classifier.

Figure 3 shows the detailed description of framework. The first module is the feature extraction and pre-

Table 1 Summarization of WSDL features used by existing techniques

	Technique	Features
1.	An approach to support web service classification and annotation (Bruno et al. 2005)	Web service documentation (WSDL and other provided documents) User queries
2.	Classification of web services using Tensor (Saha et al. 2008)	Service name and description WSDL operation name and description Operation parameter names and description (input/output) UDDI descriptions
3.	Web service classification (Zhuang et al. 2005)	Service name Service documentation Operation names and description Operation parameters (input/output)
4.	Web service clustering using text mining techniques (Liu and Wong 2009)	WSDL contents WSDL context Service host Service name
5.	Clustering WSDL documents to bootstrap the discovery of web services (Elgazzar et al. 2010)	WSDL contents WSDL types WSDL messages WSDL ports Service name

processing. It is a process of extracting contents from WSDL documents, pre-processing them, and then refining the feature space into a form that is useful for subsequent analysis. The preprocessing step involves tokenization, word splitting, stop word removal, function word removal, and lemmatization.

There are two steps involved in creating the ensemble model which is shown in Fig. 4. First, the individual classifier's decision is generated, and second, the decision of individual classifiers is combined to make a new final decision.

Let N be the number of classifiers denoted by $C_1 \ldots C_N$, and M is the number of output classes. The classifier ensemble method is defined as: find the vector V which is a Boolean array representing the binary vote based ensemble. The size of V can be denoted by N × M. In Boolean array, V(i, j) signifies the decision whether ith classifier has voted in favor of jth class or not. V(i, j) = True/1 presents that ith classifier has voted for jth class; whereas V(i, j) = False/0 shows that the ith classifier has voted for another class. If 2 out of 3 classifiers predict the same class then the final ensemble will output the majority voting class. The main focus of the ensemble classifier is to make a heterogeneous ensemble model combining different base classifiers that differ in their decisions. Different classification decisions are obtained based on their functionalities that change the final decision and can output diverse results.

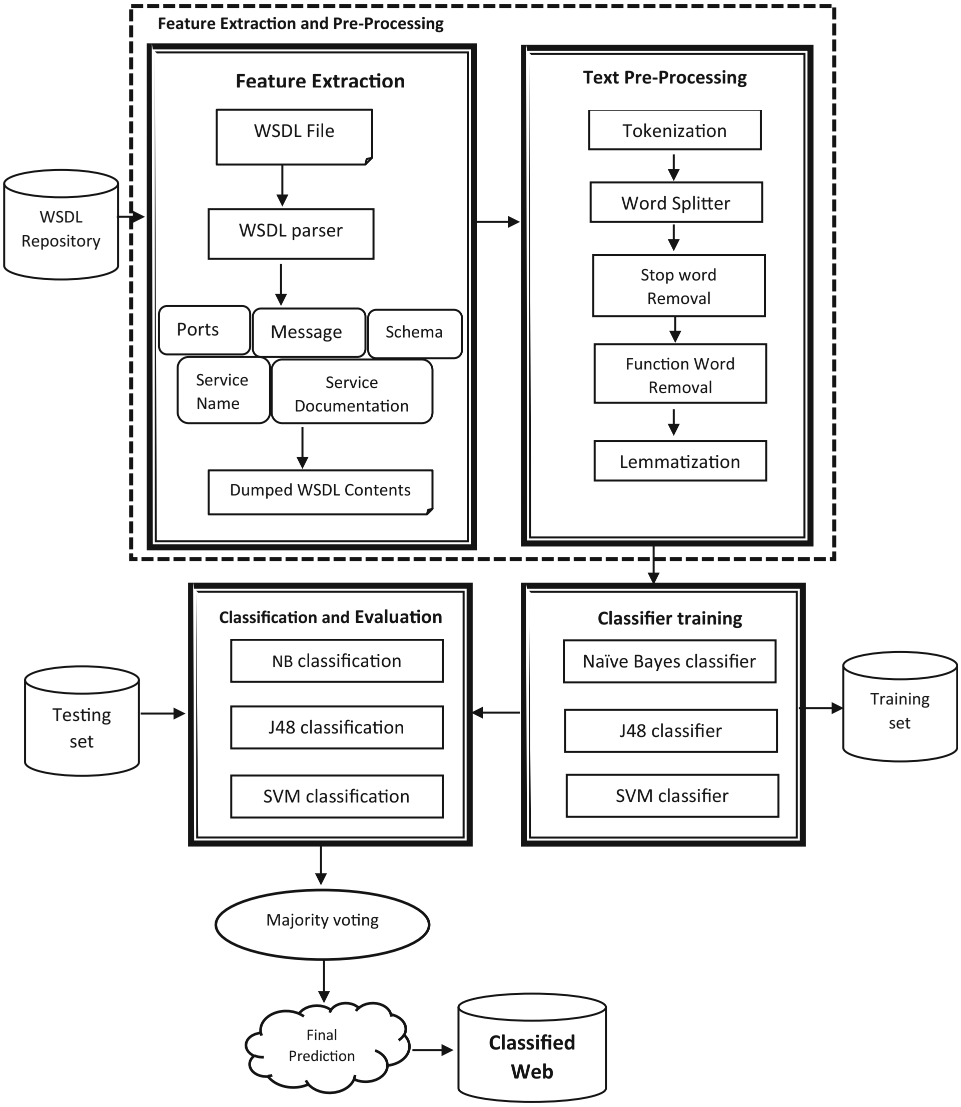

Fig. 3 Detailed description of the framework

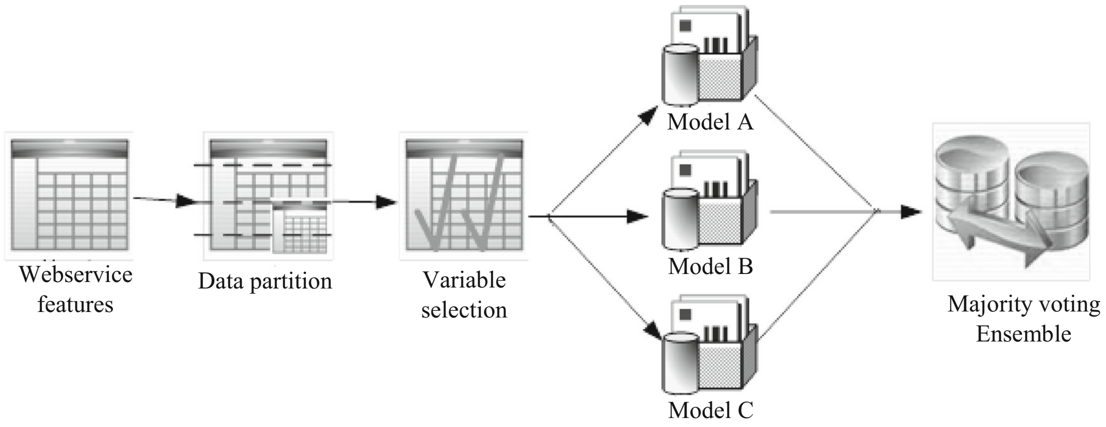

Fig. 4 Framework of the ensemble model

3.1 Feature Extraction and Pre-Processing

The feature extraction and pre-processing phase consists of two components.

(a) WSDL feature extraction
(b) Text pre-processing

3.1.1 WSDL Feature Extraction

WSDL documents will be processed for extracting relevant features. In this phase, the contents of WSDL documents extracted are named as:

- Service name
- Service documentation
- WSDL schema
- WSDL messages
- WSDL port types.

Elements of the WSDL schema consist of name and attribute type. Following information from the WSDL schema is extracted as part of the WSDL content.

- Name attribute of complex type.
- Documentation content of element in sequence of complex type.
- Documentation content of complex types.
- Documentation content of elements in schema.

From each message element, following information is extracted.

- Name attribute of part.
- Element attribute of part.

Information extracted from port types include:

- Name attribute of portType.
- Documentation content of portType.
- Name attribute of each operation in portType.
- Documentation content of each operation in portType.
- Name attribute of input/output parameters in each operation.

This information including Service Name and Service Documentation is transferred into a text file. These selected WSDL contents are used as a base for web service classification.

3.1.2 Text Pre-Processing

The goal of the pre-processing phase is to enhance the quality of information available for classification. Information in a text file might be inconsistent and may lead the mining process to inaccurate results. During this phase different pre-processing steps are applied to extract accurate and consistent information. We use following steps for pre-processing the WSDL contents.

- *Tokenization*: Tokenization is the very first stage of morphological analysis. It is the process of splitting a stream of text into words, symbols and phrases called tokens. Although the text is stored in machine readable format, meaningless characters like hyphens, commas, brackets etc. need to be eliminated by tokenization. We have used a freely available Java tokenizer and parser tool which can be downloaded from http://www.sourceforge.net/projects/jtopas/.
- *Word splitter*: The word splitter splits concatenated words based on the capitalization of letters For example, a web service has an operation named *ValidateAddressResponse*. This name is meaningless unless it is split into words *Validate*, *Address* and *Response*. The word splitter is introduced as a new pre-processing step which is not used by existing web service clustering and classification techniques.

- *Stop word removal*: The next step in pre-processing is to filter stop words from textual information. Stop list contains prepositions and articles which are insignificant and can be easily removed from a document. A publically available SMART stop word list (ftp://ftp.cs.cornell.edu/pub/smart/) was used to eliminate stop words.
- *Function word removal*: A stop words list typically eliminates function words but there are a few function words which are not stop words. This approach eliminates the remaining function words by performing a comparison of the SMART stop word list and the publically available function word list at http://www.flesl.net/
- *Lemmatization*: The last step of pre-processing is lemmatization. Lemmatization is the process of reducing varied or derivational forms of a word to a common root form. Lemmatization reduces words to a valid dictionary form by using vocabulary and performing a morphological analysis of words. This valid dictionary form is known as *lemma.* For example, consider a token *saw*, stemming will reduce it to the letter *s*, whereas lemmatization will return it to either *saw* or *see* depending on whether the word is used as a noun or as a verb. In short, different inflectional forms of word (lemma) are reduced by lemmatization whereas only derivationally related words are reduced by stemming. Lemmatization is performed using *Stanford CoreNLP* (http://nlp.stanford.edu/software/).

3.2 Web Service Classification

During this phase, web services are classified into functionally similar categories. The classification of web services into domain specific groups is performed using the ensemble which is a combination of three techniques, Naïve Bayes, J48 Decision tree and Support Vector machine. The description of each classifier is given as follows:

3.2.1 Naïve Bayes Classifier

Naïve Bayes classifier is a family of probabilistic classifiers which focuses on applying the Bayesian algorithm with strong independence between different attributes. The Naïve Bayes algorithm is a popular text categorization machine learning algorithm with high classification accuracy (Youquan et al. 2011).

The Naïve Bayes algorithm is a highly scalable algorithm and requires a linear number of parameters with a number of variables in machine learning. It uses linear time instead of iterative approximation as used by many other algorithms.

Let X is a vector that needs to be classified and C is the output class. It is necessary to determine the probability of X belonging to class C_k. Following Bayesian rule is used to determine $P(C_k|X)$:

$$P(C_k|X) = P(C_k) \times \frac{P(X|C_k)}{P(X)} \tag{1}$$

The training dataset is used to determine class probability $P(C_k)$ but due to sparseness of data, it is not possible to directly estimate of $P(X|C_K)$.

Therefore, it is decomposed as follows:

$$P(X|Ck) = \prod_{j=1}^{d} P(X_j|C) \tag{2}$$

where X_j represents the jth element of vector X. Combining Eq. (1) and (2), we have

$$P(C_k|X) = P(C_k) \times \frac{\prod_{j=1}^{d} P(x_j|C_k)}{P(X)} \tag{3}$$

Naïve Bayes has the advantage that only a small amount of datasets is required for training and estimation, such as central tendency and input of parameters. Due to the independence of attributes, only few attributes are needed for an estimation, instead of the entire covariance matrix (Patil and Pawar 2012).

The posterior probabilities are calculated using the following formula:

$$P(C|F_1, \ldots, F_n) = \frac{P(C)P(F_1, \ldots, F_n)}{P(F_1, \ldots, F_n)} \tag{4}$$

The output class which has high probability will be assigned to the testing tuple. The Naïve Bayes is commonly used for discrete and continuous values. The Naïve Bayes algorithm works as a linear classifier when X is a vector of a discretely valued attribute.

The Naïve Bayes algorithm can be used in multiple domains. It remains a popular method for text categorization where it uses a linear time rather than expensive iterative approximation.

3.2.2 Decision Tree (J48)

The Decision trees are one of the most popular algorithms in machine learning and classification (Bhargava et al. 2013). The instance space is recursively partitioned into subspaces using a decision tree. A decision tree is also termed as directed tree because its root node does not have any incoming edge. All the other nodes have exactly one incoming edge. The internal node has one outgoing edge. All the remaining nodes in the decision tree are called leaf nodes. Each instance space in a decision tree is partitioned into sub-spaces based on some splitting criteria of the input attribute value. There are multiple criteria that can be used to select an internal node as a root node. Most commonly used criteria are the decision tree inductions based on Information Gain, the decision tree inductions based on Gain Ratio and the decision tree inductions based on Gini Index.

The decision tree induction based on Gain Ratio is also called C4.5 or J48 in Weka. C4.5 is a modification of a decision tree based on Information Gain where the effect of bias is reduced towards multi-valued attributes.

When selecting the splitting attribute for constructing a tree, the Gain Ratio criteria note the number and size of branches. Therefore, it considers the intrinsic information of an attribute for decision tree construction and chooses the internal root node (Bhargava et al. 2013).

$$\mathrm{Gain\,Ratio}\ (a_i, S) = \frac{\mathrm{Gain}\ (a_i, S)}{\mathrm{Entropy}\ (a_i, S)} \tag{5}$$

The Information Gain of an attribute A is calculated by following formula:

$$\mathrm{Gain(S,\ A) = E(S) - I(S,\ A) = E(S)} - \sum_{i=1}^{k} \frac{|S_i|}{|S|} E|S_i| \tag{6}$$

The entropy for each attribute is calculated by using the formula (Bhargava et al. 2013):

$$E = -\sum\nolimits_{i=1}^{k} P_i \log_2 P_i \tag{7}$$

where k corresponds to the number of target attribute classes and P_i presents the probability of occurring in class P. The entropy of the entire set of attributes is calculated by weighted average over all sets using following formula:

$$\mathrm{I(S,\ A)} = \sum_{i=1}^{k} \frac{|S_i|}{|S|} E|S_i| \tag{8}$$

J48 outperforms other trees such as Information Gain and Gini Index in terms of both accuracy measures and the handling of complex values. The breadth and uniformity of values is allowed for in J48 by reducing the effect of bias. The attribute with the highest Gain Ratio is selected as the splitting attribute in order to construct a decision tree (Bhargava et al. 2013).

3.2.3 Support Vector Machine (SVM)

A support vector machine is a supervised machine learning method with associated learning. It is used to analyze and recognize patterns in data. SVM is mostly used for classification and regression. A training instance is assigned to one of two classes. A new model is constructed from the SVM algorithm which is then used to classify an unknown instance into either one class or another. SVM is a non-probabilistic binary linear classifier (Wang et al. 2010).

Applying the kernel equations arranges the data instances in such a way within the multi-dimensional space that a hyper-plane is created that separates data instances of one kind from those of another. The kernel function is referred to as any function which transforms linearly non-separable data of one domain into another domain. Its instances become linearly separable in the new domain. There are multiple types of kernel equations such as linear, quadratic, Gaussian or others. After dividing the data into two categories, the best hyper-planes are determined to divide the data into two types of instances.

SVM follows the following classification rule (Varguez-Moo et al. 2013):

$$\mathrm{Sgn\,(f(x,\ w,\ b))} \tag{9}$$

$$\mathrm{f(x,\ w,\ b)} \leq w.x > \ + b \tag{10}$$

where x is the example to be classified and the maximum margin hyperplane (w, b) represents a complex problem with a unique solution. The ultimate solution is to minimize ||w|| within the specified constraints.

$$y_i(\langle w.\mathrm{x}_i\rangle + b) \geq 1 \tag{11}$$

In basic SVM framework, the two classes are classified based on a hyperplane, written as:

$$\mathrm{(w.x) + b = 0}\, w \in \mathrm{R}^n, \mathrm{b} \in \mathrm{R} \tag{12}$$

The linear SVM correctly classifies all training data using following formula:

$$\begin{array}{ll} \mathrm{wx_i + b} \geq 1 & \text{if } \mathrm{y}_i = +1 \\ \mathrm{wx_i + b} \leq 1 & \text{if } \mathrm{y}_i = -1 \end{array} \tag{13}$$

Maximize the margin by:

$$\mathrm{M} = \frac{2}{|\mathrm{w}|} \tag{14}$$

SVM utilizes only binary data for categorization. Even if the data is not binary, SVM handles it as if it were. The analysis of data is performed through binary assessment.

The pre-processed data is passed on to Naïve Bayes, the J48 Decision tree and the SVM classifier to train each classifier and to make them useful for classification of an unknown web service. Each trained classifier has the knowledge about the feature space for each dataset and the resultant class label.

3.2.4 Working of the Ensemble

How the ensemble framework works can be easily understood by following steps:

Table 2 Dataset categories and their web services

S. no.	Category	No. of web services
1.	Business	500
2.	Education	251
3.	Science	980
4.	Weather	1200
5.	Media	807

1. Suppose that we need to categorize a web service into a single category such as currency and stock, fax and messaging and microarray etc.
2. The information is obtained from a WSDL document such as service name, service documentation, WSDL schema, WSDL messages and WSDL port types.
3. Pre-processing is applied on the downloaded information in order to refine it. For example tokenization, stop words removal, word splitter, function word removal and lemmatization are applied.
4. Each trained classifier classifies the web service into one category.
5. The output of all three classifiers is combined using majority voting approach.
6. The category which attained the highest vote will be the output of the testing web service.

4 Results and Analysis

In our approach, WSDL documents are used as the main source of data repositories because a web service is completely described by the information carried by these XML files. We have evaluated the accuracy of the framework for a publicly available dataset "WSExpress: Dataset 4, WSDL dataset" (Zheng 2012) consisting of 3738 real world web services categorized into 5 fields. Table 2 gives the names of categories and web services included in each category.

All the classifiers need to be trained with pre-classified datasets of web services. Tenfold cross validation is used to determine the accuracy of the models. Table 3 shows the mean precision and recall values for each category in the dataset.

Table 3 Mean precision and recall for classification of dataset categories

No.	Category	Precision (%)	Recall
1	Business	97	93
2	Education	91	89
3	Science	88	91
4	Weather	98	88
5	Media	91	96

Table 4 Relationship between positive and negative results

State predicted class	Known class	
	A	B
A	*True positives*	False positives
B	False negatives	*True negatives*

Italic words represent true prediction values

Where precision and recall are defined as:

$$\text{Recall} = \frac{\text{True \ positives}}{\text{True \ positives} + \text{false \ negatives}}$$

$$\text{Precision} = \frac{\text{True \ negatives}}{\text{True \ negatives} + \text{false \ positives}}$$

F-Measure presents the harmonic mean of both recall and precision. Mathematically,

$$\text{F}{-}\text{Measure} = 2 \times \frac{\text{Recall} \ \times \ \text{precision}}{\text{Recall} + \text{precision}}$$

Accuracy measures the percentage of correct predictions carried out by the model in comparison with actual measurements performed on test data. Mathematically,

$$\text{Accuracy} = \frac{\text{True \ positives} + \text{true \ negatives}}{\text{True \ positives} + \text{false \ positives} + \text{True \ negatives} + \text{false \ negatives}}$$

The traditional confusion matrix is defined in Table 4. The diagonal elements (true positives and true negatives) in the confusion matrix represent the correctly classified data for each class. Other elements (false positives and false negatives) show incorrectly classified data for respective classes.

Tables 5 and 6 show comparison results of our approach with three state-of-the-art web service classification techniques (Bruno et al. 2005; Liu and Wong 2009; Elgazzar et al. 2010).

Table 5 Accuracy comparison of five categories

Category	Accuracy			
	Liu and Wong approach (%)	Elgazzar et al. approach (%)	Bruno et al. approach (%)	Our approach (%)
Business	82	89	62	94
Education	78	82	57	91
Science	77	85	64	89
Weather	80	84	66	91
Media	71	88	55	94

Table 6 Performance evaluation related to five identified categories

Category	Liu and Wong approach			Elgazzar et al. approach			Bruno et al. approach			Our approach		
	F-measure (%)	Precision (%)	Recall (%)	F-measure (%)	Precision	Precision (%)	F-measure (%)	Recall (%)	Recall (%)	F-measure (%)	Precision (%)	Recall (%)
Business	82	84	81	88	88	89	62	64	61	95	97	93
Education	77	78	77	82	81	84	55	51	62	90	91	89
Science	77	80	74	84	86	84	62	57	68	89	88	91
Weather	82	88	77	83	86	81	69	66	72	93	98	88
Media	70	68	73	89	88	91	55	58	53	93	91	96

Looking at the performance of our approach and existing web service clustering and classification approaches, it can be seen that our approach yields the highest precision and recall for all identified categories. This increase in accuracy is owed to two main factors, i.e., enhanced preprocessing with focused feature selection, and majority based ensemble classification.

Liu and Wong (2009) use WSDL contents, but they do not clearly specify whether WSDL contents refer to service documentation contents or WSDL features. However, our approach clearly states which attributes and documentation contents rely on WSDL content. Elgazzar et al. (2010) improved the Liu and Wong approach by using a different set of features and replacing service context and service host features with WSDL types, messages and ports. Compared to this, we extract service name, service documentation, WSDL types, messages, and ports and combine them to form WSDL contents instead of using them separately. Elgazzar et al. (2010) used type attributes of WSDL types instead of using name attributes, but this might be misleading because two web services belonging to the same category might have different input types for the same type of operations.

The ensemble model also plays a very important role for improving the accuracy of the web service classification. A heterogeneous classifier ensemble model is used by combining entirely different types of classifiers to achieve a higher level of diversity. The Naïve Bayes classifier considers each attribute independently without taking into account the relation between them, whereas the ensemble model can handle dependency and relation between given attribute sets by using the J48 algorithm where neighbors determine the class label for unknown instance. The Linear regression model determines the statistical relationship between various independent and dependent variables and achieves optimal results. The SVM algorithm performs the feature selection by using only a subset of data chosen based on Information Gain. Thus, all of the three selected classifiers complement each other excellently. In any scenario where one classifier has a limitation, another classifier overcomes it, and as a result higher ensemble performance is achieved.

5 Conclusion

Effective web service classification is an important issue for web services. In this paper, we propose a majority vote based ensemble for the classification of web services. The ensemble model overcomes the limitations of conventional techniques by employing the ensemble of three heterogeneous classifiers: Naïve Bayes, Decision Tree (J48) and Support Vector Machines. This approach is validated for a publicly available dataset consisting of 3738 real world web services categorized into 5 fields, i.e., Business, Education, Science, Weather and Media, yielding an average accuracy of 92 %.

References

Bai P, Li P (2009) The improved naive Bayesian web text classification algorithm. Int Symp Comput Netw Multimed Technol 2009:1–4

Bhargava DN, Sharma G, Bhargava R, Mathuria M (2013) Decision tree analysis on j48 algorithm for data mining. Int J Adv Res Comput Sci Softw Eng 3(6):RAJ123

Bruno M, Canfora G, Penta MD, Scognamiglio R (2005) An approach to support web service classification and annotation. In: Proceedings of the 2005 IEEE international conference on e-technology, e-commerce and e-service. Washington, DC

Elgazzar K, Hassan AE, Martin P (2010) Clustering WSDL documents to bootstrap the discovery of web services. In: 2010 IEEE international conference on web services (ICWS), pp 147–154

Elgedawy I (2014) USTA: an aspect-oriented knowledge management framework for reusable assets discovery. Arab J Sci Eng 40(2):451–474

Huang H, He L, Chen X, Yu M, Wang Z (2013) Automatic composition of heterogeneous models based on semantic web services. Int J Parallel Progr 43(3):339–358

Katakis I, Meditskos G, Tsoumakas G, Bassiliades N, Vlahavas IP (2009) On the combination of textual and semantic descriptions for automated semantic web service classification. In: Iliadis L, Maglogiannis I, Tsoumakas G, Vlahavas I, Bramer M (eds) Artificial intelligence applications and innovations III. Springer, New York, pp 95–104

Kouki J, Chainbi W, Ghedira K (2014) A novel framework for bindings synchronization of web services. Serv Oriented Comput Appl 9(1):59–74

Liu W, Wong W (2009) Web service clustering using text mining techniques. Int J Agent-Oriented Softw Eng 3(1):6–26

Liu ZZ, Jia ZP, Xue X, An JY (2014) Reliable web service composition based on QoS dynamic prediction. Soft Comput 19(5):1409–1425

Malki A, Benslimane D, Benslimane SM, Barhamgi M, Malki M, Ghodous P, Drira K (2015) Data services with uncertain and correlated semantics. World Wide Web. doi:10.1007/s11280-014-0317-x

Manning CD, Raghavan P, Schütze H (2008) Introduction to information retrieval. Cambridge University Press, Cambridge

Martino Di (2009) Semantic web services discovery based on structural ontology matching. Int J Web Grid Serv. doi:10.1504/IJWGS.2009.023868

Patil AS, Pawar BV (2012) Automated classification of web sites using Naive Bayesian algorithm. In: Proceedings of the international multiconference of engineers and computer scientists

Plebani P, Pernici B (2009) URBE: web service retrieval based on similarity evaluation. IEEE Trans Knowl Data Eng 21(11):1629–1642

Rokach L, Schclar A, Itach E (2014) Ensemble methods for multi-label classification. Expert Syst Appl 41(16):7507–7523

Saha S, Murthy CA, Pal SK (2008) Classification of web services using tensor space model and rough ensemble classifier. In: Foundations of intelligent systems. Springer, Heidelberg, pp 508–513

Varguez-Moo M, Moo-Mena F, Uc-Cetina V (2013) Use of classification algorithms for semantic web services discovery. J Comput 8(7):1810–1814

Wang H, Shi Y, Zhou X, Zhou Q, Shao S, Bouguettaya A (2010) Web service classification using support vector machine. In: 22nd IEEE international conference on tools with artificial intelligence (ICTAI), vol 1, pp 3–6

Youquan H, Jianfang X, Cheng X (2011) An improved Naive Bayesian algorithm for web page text classification. In: Eighth international conference on fuzzy systems and knowledge discovery (FSKD), vol 3, pp 1765–1768

Zheng Z (2012) WS-DREAM. http://www.wsdream.net/dataset.html. Accessed 28 Jan 2015

Zhuang Z, Mitra P, Jaiswal A (2005) Corpus-based web services matchmaking. In: Workshop on exploring planning and scheduling for web services, grid and autonomic computing