International Journal of u- and e- Service, Science and Technology
Vol.8, No.4 (2015), pp.339-348
http://dx.doi.org/10.14257/ijunesst.2015.8.4.31

Semi-automatic Web Service Classification Using Machine Learning

Jie Yang and Xianzhong Zhou[+]

School of Management and Engineering, Nanjing University, Nanjing 210093, PRC
Dg1115014@gmail.com, zhouxz@nju.edu.cn

Abstract

The number of web services on the Internet is increasing continuously. Determining a web service's domain is very helpful for many issues like web service discovery, composition and semantic annotation. Based on predefined domains, we implemented semi-automatic web service classification with supervised machine learning technique in the paper. First we exploited the process of web service classification with choosing element from web service description document, preprocessing data, mapping to a digital vector and applying classifier. Then service classification accuracy of single element and multi-element based on four machine learning algorithms were tested. At last, experimental results showed that C4.5 classifier had a higher accuracy and efficiency in web service classification. In addition, multi-element classification did not perform better than single element.

Keywords**:** *Web Services; Classification; Machine Learning*

1. Introduction

With the drastic growth of web services on the Internet, web service discovery is a hot topic now. It is time consuming to traverse a whole repository to find the matched service. To improve efficiency of web service discovery, we adopt machine learning technique to classify web services. UDDI is the most widely shared registry, and providers publish web services into it. UDDI defines classification category based on taxonomies. Providers also define category for a web service manually. And in some websites[1], web services are listed by submission time or providers. These are not very helpful for web service discovery and semantic annotation. In this paper, our goal is to classify web services with the identical or similar functionalities into one category semi-automatically and annotate web services semantically.

This paper is organized as follows: section II presents the related works; section III introduces the process of web service classification; section IV gives the empirical study; section V discusses the experimental results; section VI draws the conclusions and discusses future work.

2. Related Work

The existing research on web service classification has primarily focused on two aspects: text classification and element classification.

M Crasso *et al.* [1] mapped the textual description in a WSDL (Web Service Description Language, an xml-based standard for describing functionality web service offers) into a vector by text mining. And they used Rocchio, k-NN (k-nearest neighbor)

[+] Corresponding author.
[1] www.xmethods.com

ISSN: 2005-4246 IJUNESST

and Naïve Bayes to classify web services respectively. Then accuracy of the three algorithms is compared. M Bruno *et al.* [2] built concept lattices of web services based on textual description, and adopted SVM (Support Vector Machine) to perform automatic classification and semantic annotation. Also in [3], H Wang et al mapped descriptive information of categories into high dimension feature space, and performed classification based on SVM. But the mapping from high to low dimension does not perform well. Text classification mainly relies on contents of element —<documentation> in a wsdl. It ignores the other elements, which are useful for representing and classifying a web service too.

Element classification is based on every element in a wsdl. The elements include Message (input and output), Interface (called port type in wsdl 1.1), Operation, *etc.* The works of [4, 5] are based on port type, operation, message, names and comments from documentation. Then they adopted Naïve Bayes, SVM and HyperPipes to implement web service classification. But the number of elements is too much, preparatory work is a burden. And it is not true that the number of elements for classification is more, the accuracy is higher.

M Á Corella *et al.* [6] proposed a web service classification method by comparing a new unclassified service with a set of classified services based on operation similarity. But it is time and memory consuming to compute similarity of an unclassified service with each classified service. E Boujarwah *et al.* [7] built functional domains by using conceptual graph, and classified web services into these domains. In the conceptual graph, the terms in a wsdl are considered as the concept nodes. The terms are extracted from elements, including service name, port type, operation name, etc. Their work adopted an unsupervised machine learning technique. In the paper, we incline to the supervised web service classification. Before implementing classification, categories of part of web services in a repository have been defined manually.

To sum up, accuracy of web service classification is primarily determined by two stages: choosing element from web service description document; applying a proper machine learning algorithm.

3. Web Service Classification

As the statement in Introduction, web services with the identical or similar functionalities are classified into one category. This section will explain the process of web service classification, which can be seen in Figure 1. Web service classification is based on wsdl document. As WSDL is the most widely accepted and used web service description language.

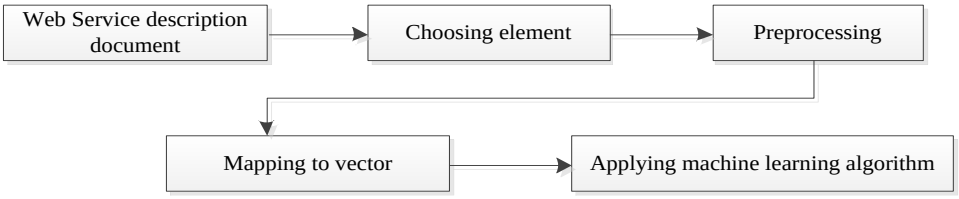


Figure 1. Steps of Web Service Classification

3.1. Choosing Element

Choosing element is the first step of web service classification. Elements include static and dynamic. For example, QoS (Quality of Service) is a dynamic element. QoS refers to the quality of a web service, and may include availability, reliability, response time, etc. These measures change with web service's invoked times frequently. We choose to utilize static elements rather than dynamic, because static elements represent web service's

functionality better than dynamic. Words are extracted from contents of chosen elements in description document.

3.2. Preprocessing

This step is to preprocess extracted words. There are four aspects: splitting, eliminating stop words, stemming and removing specific tags. Splitting: combined words are split into verbs and nouns. Because of naming convention, verb and noun are combined in operation names, service names and messages of web services. Eliminating stop words: words are filtered by stop list [8]. The stop list contains a few common words (including preposition, article, etc.), which are recognized that they are not useful in information retrieval. And in web service classification, stop words could not represent any category either. Stemming [9], verbs are taken back to infinitive; nouns are taken from plurals to singles; English words are taken to American words. Removing tags and specific words: they are not helpful for classification. For example, "web", "service" and service provider's name frequently occur in wsdl file, but they cannot discriminate web services. And tags such as <input>, <output>, exist in almost every wsdl file.

3.3. Mapping to a Vector

After preprocessing, these words represent the web service, and will be mapped into a vector using word weighting. Word weighting is to determine that how important a word is for a web service. Formed vector represents the web service to implement classification by applying machine learning algorithm.

3.4. Applying Machine Learning Algorithm

Applying machine learning algorithm in the supervised web service classification contains two stages: training vectors and classifying vectors. We could obtain model by training dataset. And the model is applied to unclassified dataset. Machine learning algorithm could be SVM, BPNN, Naïve Bayes, and so on. Accuracy of classification is a measure for these algorithms.

4. Empirical Study

We chose the elements service name, operation name, input and output of wsdl for web service classification in the experiments. And combination of service name and operation name, input and output were tested too.

4.1. Preprocessing of Web Service Classification

The stages of preprocessing are seen in Table 1 and Table 2. They show the contents of service name, operation name before and after preprocessing respectively.

Table 1. Service Name and Operation Name before Preprocessing

Service name	Number of Operation	Operation name
TranslatorWebService	2	getEnCnTwoWayTranslator, HelloWebXml
Weather Web Service	5	getSupportCity, getSupportProvince, getSupportDataSet, getWeatherbyCityName, getWeatherbyCityNamePro
Cocoma Video Web Service	2	GetVideo, GetExtendedVideo
Cocoma Google Search Web Service	5	GetSiteCount, GetLinkCount, GetRelatedCount, GetKeywordRankding, GetKeywordSuggest
US Zip Validator	1	ValidateZip

Table 2. Preprocessed Service Name and Operation Name

Service name	Number of Operation	Operation name
Translator	2	English Chinese Two Way Translator, Hello Web Xml
Weather	5	Support City, Support Province, Support Data Set, Weather City Name, Weather City Name Pro
Video	2	Video, Extend Video
Google Search	5	Site Count, Link Count, Relate Count, Keyword Rank, Keyword Suggest
US Zip Validator	1	Validate Zip

4.2. Mapping to Vector

As a result of previous step, TF-IDF is used to word weighting. TF-IDF [10] is a numerical statistic that reflects how important a word is for a document in a collection set. For each term in a document:

$$tf\text{-}idf_{ij} = TF_{ij} \cdot IDF_i \tag{1}$$

TF (Term Frequency) is obtained that the number of occurrence of term t_i in a document d_j divides the number of all terms in d_j, which can be seen in formula (2). TF shows the frequency of a term in a document. IDF (Inverse Document Frequency) means whether a term is rare in a collection. IDF value of a term in a document d is high suggests that the term can represent the document d. In formula (3), $|D|$ is the number of documents in the collection;
$N(t_i, d)$ is the number of documents which include term t_i.

$$TF_{ij} = \frac{n_{ij}}{\sum_k n_{kj}} \tag{2}$$

$$IDF_i = \log \frac{|D|}{N(t_i, d)} \tag{3}$$

A web service is mapped into a vector $S = (e_0, e_1, \cdots, e_n)$. If one web service is similar with another web service, then two services' vectors are close.

$T = (C_1, C_2, \cdots, C_n)$, T is the set of categories of web service; C_i is a category; n is the number of categories. Suppose that $C_i \cap C_j = \varnothing$, means intersection of C_i and C_j is empty. Web service collection in the experiment is seen in Table 3. Data source is from OWLS-TC4[2] and it includes nine domains: communication, economy, education, food, geography, medical, simulation and weapon.

In the web service collection, Tf-idf value of part of words is seen in Table 4.

Table 3. Web Service Dataset (the Number of Web Services in Each Category)

Category	Quantity	Category	Quantity	Category	Quantity
communication	30	economy	29	education	55
food	23	geography	44	medical	41
simulation	16	travel	57	weapon	11

[2] http://projects.semwebcentral.org/

Table 4. Tf-idf Value of Part of Words

Word	Tf-idf vlaue	Category	Word	Tf-idf value	Category
mass	0.025	weapon	physical	0.016949153	education
organization	0.013071968	weapon	phone	0.008474576	medical
destruction	0.025	weapon	clinic	0.016949153	medical
lecture	0.008474576	education	provide	0.011846949	medical
pioneer	0.004237288	education	provide	0.0029617373	education

4.3. Overview of Employed Classifiers

We adopted four classic machine learning algorithms for web service classification and compared them with accuracy. The classifiers include SVM, Naïve Bayes, Decision Tree and BPNN.

4.3.1 SVM: SVM (Support Vector Machine) is provided by Vladimir N. Vapnik for pattern recognition in 1995 [11]. SVM has become popular in regression and classification recently. In SVM model, samples are represented as points of space. The established model is to find a hyperplane to separate categories as wide as possible. The hyperplane is considered as the best classifier to decide points' category based on which side of hyperplane they fall on. This is based on two category divisions, as linear classify. When it is not linear separable in plane space, the points are mapped into high dimension space by using kernel function. In high dimension space, the points are linear separable. Selecting kernel function depends on the problems. SVM's advantages are as followings: it has the capability of dealing with scared samples, nonlinear and high dimension problems; it doesn't trap in the local minimum value.

4.3.2 Naïve Bayes: Naïve Bayes classifier is simply to classify collection based on Bayes' theorem in supervised classification. Naïve Bayes assumes that all properties of a category are strongly mutual independent. But this supposition is not existent in a real world. Naïve Bayes' principle is that, it will compute the probability of unclassified data which belongs to each category; category of the most probability is the one unclassified data belongs to. Naïve Bayes is proven by many experiments that it has a good effect.

Trained dataset: $C = \{c_1, c_2, \ldots, c_n\}$, unclassified set of properties:

$$P(D \mid c_i) = \prod_{j=1}^{m} P(d_j \mid c_i) \tag{4}$$

Bayes' theorem:

$$P(c_i \mid D) = \frac{P(D \mid c_i)P(c_i)}{P(D)} \tag{5}$$

To make $P(c_i \mid D)$ be the largest，means $P(D \mid c_i)P(c_i)$ to be the largest. Then we will obtain unclassified sample's category.

Naïve Bayes classifier's advantages are as followings: it is based on solid mathematic foundation; estimating parameters are rare; it is not sensitive to missing data; it has a good and stable classification effect.

4.3.3 Decision Tree, C4.5 Algorithm: C4.5 algorithm is provided by J.R. Qiunlan in 1986 and can be used for classification, data mining, *etc.* [12]. C4.5 is the extension of ID3 algorithm. In ID3, classification model is established by constructing a binary decision tree. Each node of the tree is split by attribute. Information Gain is considered as a criterion to select attribute for each node. The attribute with the largest information gain is chosen firstly. If a sample's attribute value is smaller or equal to information gain, the sample is as left branch. If it is greater than information gain, it is as right branch. According to above, IF-THEN rule is formed. In C4.5, attribute selection is with information gain ratio. If an attribute has more values, information gain leans toward the attribute. For this, C4.5's attribute selection is with information gain ratio other than information gain.

After decision tree is established, pruning is the next step. Pruning is to eliminate isolated point and noisy part in the tree. C4.5 has the following advantages: it is easy to understand its classification rule; classification accuracy is high; it is widely used in many fields, like data mining, text classification.

4.4.4 BPNN: Back Propagation Neural Network (BPNN), is the most widely used learning algorithm in Artificial Neural Network (ANN). David E. Rumelhart, Geoffrey E. Hinton, Ronald J. Williams provided it in 1986 [13]. It made a research boom in 1990s, because it is a non-linear network and with self-learning ability. BPNN is a multilayer perception network. There are three layers: input layer, output layer and hidden layer. Each layer includes multiple neurons (as nodes). Inputs of each node in hidden layer generate outputs of each node in input layer; in the same way, output layer's inputs generate hidden layer's outputs. The samples are as input, processed in hidden player, and actual outputs are obtained in output layer.

In the course of perception learning, network will continually adjust value of parameters (threshold and weight) to make actual outputs match with desired outputs. BPNN has the following advantages: it is a non-linear mapping between input and output; it has the property of generalization, and it can handle data which is not in the samples and find the output; it is fault tolerant, like minor errors in samples don't influence the mapping rule from input to output.

5. Evaluation

We will give the measurement of web service classification in this section.

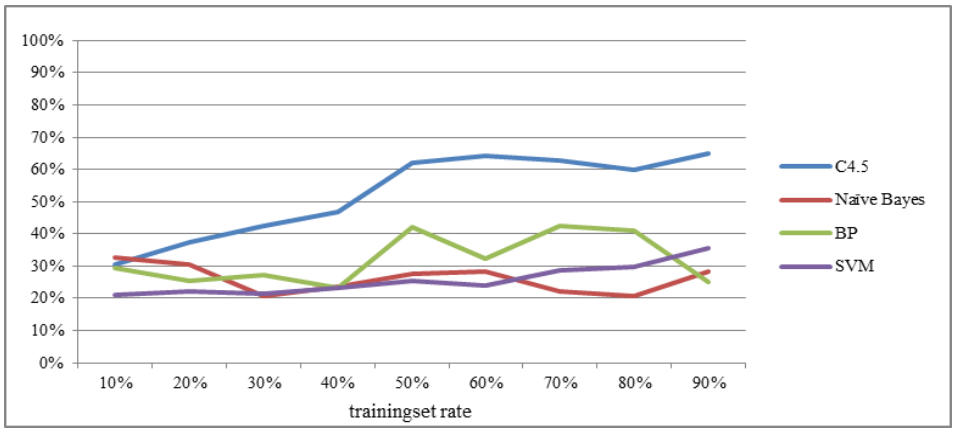
The accuracy of classification is defined in formula (6) and formula (7). Classification accuracy of collection is the average value of all categories' accuracy.

$$accuracy(T) = \frac{a(C_1) + a(C_2) + \cdots + a(C_n)}{n} \tag{6}$$

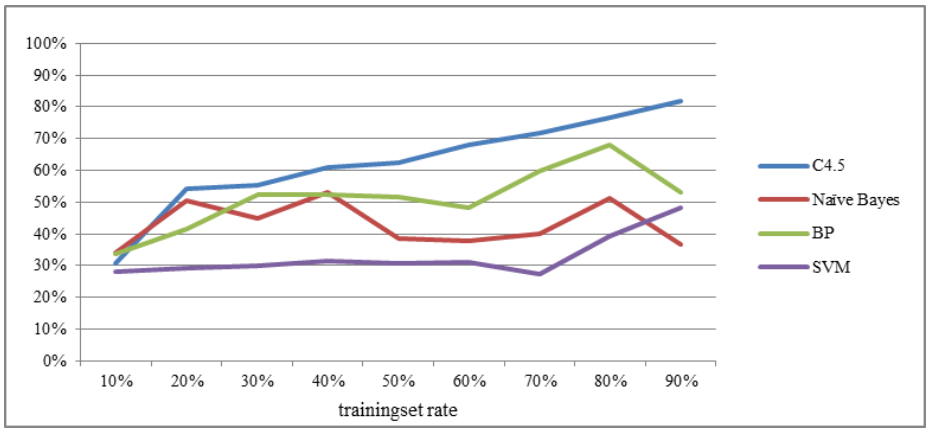
$$a(C_i) = \frac{N(x \mid m)}{m} \tag{7}$$

$a(C_i)$ is the accuracy of C_i m is the number of unclassified web services which belong to C_i. $N(x \mid m)$ is the number of web services which are classified into C_i in m.

Our experimental dataset is based on OWLS-TC4, which provides many web service documents. The contents of web service's elements (service name, operation, input and output), are extracted. After preprocessed, four bags of words are formed. And we will obtain each word's tf-idf value in each category. Based on the gained data, machine learning algorithms are employed to implement classification.



(a) element service name



(b) element operation name

Figure 2. Classification Accuracy of Four Classifiers Based on Elements of Service Name and Operation Name

Weka[3], is a machine learning tool , as our experimental platform. First we measured and compared the accuracy of four classifiers based on service name and operation name respectively. From Figure 2, we can see that C4.5 has higher accuracy than the other three classifiers, no matter based on service name or operation name. Both Naïve Bayes and SVM have lower accuracy than the other two classifiers. Then we conducted a second experiment to discuss whether multi-element's accuracy is higher than single element's with the same data. The experimental results are seen as Figure 3 and Figure 4. C4.5 is employed as classifier since we proved that C4.5 algorithm has better performance from the first experiment. From Figure 3, the accuracy, when both service name and operation name are as classification elements, is higher than the accuracy, when only service name is as classification element. But it is not higher than the accuracy, when operation name is as classification element. In Figure 4, we can get the result that the accuracy, when both input and output are as classification elements, is not higher than accuracy, when input or output is as classification element. When the training set ratio is less than 50%, element input has the highest accuracy; when the training set ratio is more than 50%, element output has the highest accuracy. So the second experimental answer is NO. From the view of less time and space complexity, we choose single element for web service classification. Actually, we also did the experiment that the classification elements are the average value of tf-idf value of service name's words and the average of tf-idf value of operation's words, seen in formula (8). But the result is not good.

[3] http://www.cs.waikato.ac.nz/ml/weka/

$$S = (nam, op) = (\frac{nt_1 + nt_2 + \cdots + nt_n}{n}, \frac{opt_1 + opt_2 + \cdots + opt_n}{n}) \quad (8)$$

The third experiment is about that which element's classification accuracy is the highest with the same data. The experimental result is seen as Figure 5. The result tells us that input has the highest accuracy when training set ratio is less than 50%; output has the highest accuracy when training set ratio is more than 50%.

Above three experiments are based on classification accuracy. From the perspective of model complexity, time for building model of four classifiers is seen in Table 5. BPNN's time is longer than the other three and C4.5 has the least time cost among the four. Therefore, we can get the result that C4.5 classifier is not merely outstanding in accuracy but also in efficiency.

Table 5. Model Building Time of Four Classifiers

classifier	SVM	Naïve Bayes	C4.5	BPNN
Time(s)	0.3	0.1	0.01	0.86

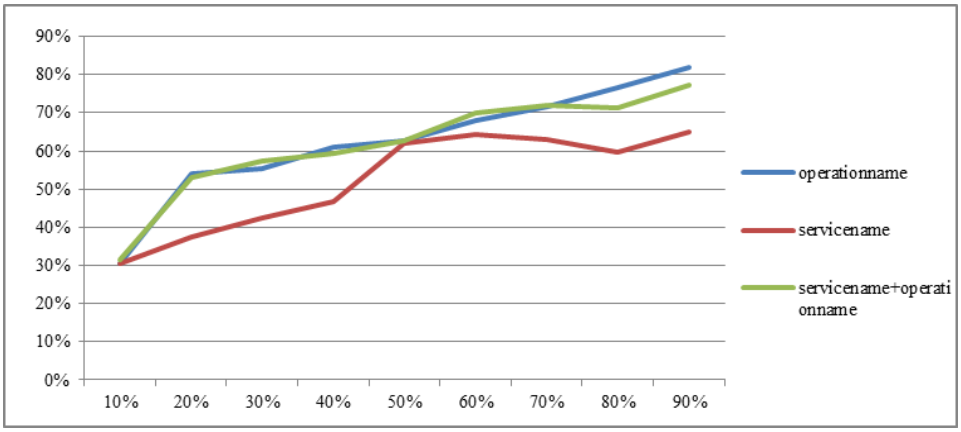


Figure 3. Classification Accuracy of C4.5 Based on Element of Operation Name, Service Name and Service Name+Operation Name

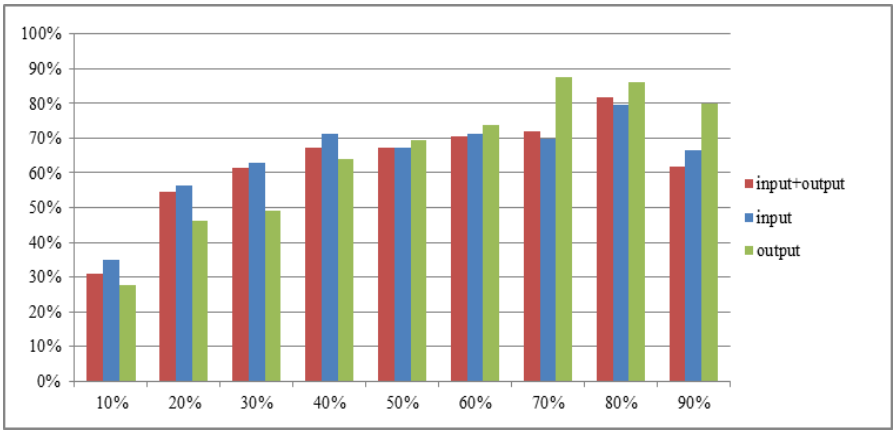


Figure 4. Classification Accuracy of C4.5 Based on Element of Input, Output and Input+output

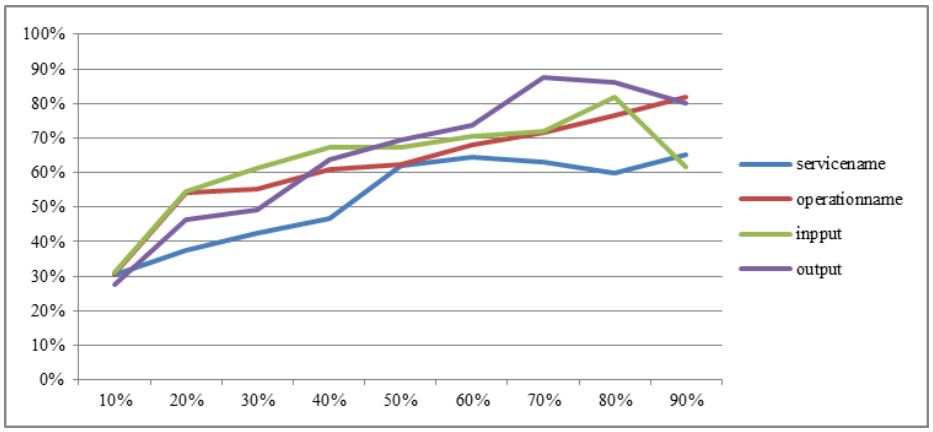


Figure 5. Accuracy of Single Element Classification

We use above experimental results to predict category of web service. C4.5 is adopted as the classifier; output is chosen as classification element; dataset is made up of its tf-idf value. The web services which have been classified incorrectly are extracted. We get these web services' predicted categories. The predicted results of some web services are presented below.

1. actual category | ws_1=geography
   p (predicted category=geography| ws_1)=0.125
   p (predicted category=simulation| ws_1)=0.875
2. actual category | ws_2=geography
   p (predicted category=education | ws_2)=0.125
   p (predicted category=geography | ws_2)=0.125
   p (predicted category=medical | ws_2)=0.75

In above samples, correct classification category exists in the predicted categories. The classifier selects classification category which has the biggest probability. In this part, we could list all the possible categories whose probabilities are more than zero. These results will be submitted to user. When dataset is small, user could choose a web service's category from all the possible categories.

6. Conclusions and Future Work

The paper implemented web service classification with machine learning algorithms. The process includes choosing classification element, preprocessing, mapping to a vector and adopting classifier. According to the experiments, C4.5 classifier performs better than Naïve Bayes, SVM and BPNN. Multi-element classification doesn't act better than single element. And when dataset is small, it is a good way to choose a web service's category by user. In future work, we will introduce classification into web service discovery to satisfy user's functional requirements.

Acknowledgements

This work is supported by National Natural Science Foundation of China (NSFC) under Grant (No. 71171107).

References

[1] M. Crasso, A. Zunino, and M. Campo, "AWSC: An approach to web service classification based on machine learning techniques", Inteligencia Artificial: revista iberoamericana de inteligencia artificial, vol. 12, no. 37, **(2008),** pp. 25-36.

[2] M. Bruno, G. Canfora, M. D. Penta and R. Scognamiglio, "An approach to support web service classification and annotation", Proceedings of IEEE International Conference on e-Technology, e-Commerce and e-Service, **(2005)**, pp. 138-143.
[3] H. Wang, Y. Shi, X. Zhou, S. Shao and A. Bouguettaya, "Web service classification using support Vector Machine". Proceedings of the 22nd IEEE International Conference on Tools with Artificial Intelligence, **(2010)**, pp. 1, 3-6.
[4] A. Heß and N. Kushmerick, "Learning to attach semantic metadata to web services", Proceedings of International Semantic Web Conference, Springer Berlin Heidelberg, **(2003)**, pp. 258-273.
[5] A. Hess and N. Kushmerick, "Automatically attaching semantic metadata to Web Services", Proceedings of Workshop on Information Integration on the Web, **(2003)**, pp. 111-116.
[6] M. Á. Corella and P. Castells, "A heuristic approach to semantic web services classification", Proceedings of Knowledge-Based Intelligent Information and Engineering Systems, Springer Berlin Heidelberg, **(2006)**, pp. 598-605.
[7] E. Boujarwah, H. Yahyaoui, and M. Almulla, "A New Unsupervised Web Services Classification based on Conceptual Graphs". Proceedings of the Eighth International Conference on Internet and Web Applications and Services, **(2013)**, pp. 90-94.
[8] C. Fox, "A stop list for general text". ACM SIGIR Forum, vol. 24, no. 1-2, **(1989)**.
[9] M. F. Porter, "An algorithm for suffix stripping". Program: electronic library and information systems, vol.14, no.3, **(1980),** pp. 130-137.
[10] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval", Information processing & management, vol. 24, no. 5, **(1988),** pp. 513-523.
[11] C. Cortes and V. Vapnik, "Support-vector networks", Machine learning, vol. 20, no. 3, **(1995),** pp. 273-297.
[12] L. Rokach and O. Maimon, "Top-down induction of decision trees classifiers - a survey", Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on, vol. 35, no.4, **(2005),** pp. 476-487.
[13] D E. Rumelhart, G E. Hintont and R J. Williams, "Learning representations by back-propagating errors", Nature, vol. 323, no. 6088, **(1986),** pp. 533-536.

Authors

Jie Yang, she born in 1986, received her Master degree in Computer Science and Technology from Taiyuan University of Technology, PRC in 2011. She is a Ph.D. candidate in School of Management and Engineering at Nanjing University, PRC. Her primary research interests include Web Service, Decision Support System, and Ontology.

Xianzhong Zhou, she was born in 1962, lives in Nanjing, PRC in 2014. He received his Master degree in Systems Engineering and Ph.D. degree in Control Theory and Application from Nanjing University of Science and Technology in 1985, 1996 respectively. He is a professor and tutor of Ph.D. in School of Management and Engineering at Nanjing University. His main research interests include Web Service, Decision Theory and Technology, C^4ISR, and Simulation of Swarm Intelligent Systems.

