Indonesian Journal of Electrical Engineering and Computer Science
Vol. 5, No. 2, February 2017, pp. 441 ~ 450
DOI: 10.11591/ijeecs.v5.i2.pp441-450

Effective Feature Set Selection and Centroid Classifier Algorithm for Web Services Discovery

Venkatachalam K[1], Karthikeyan NK[2]
[1]Sri Krishna College of Engineering and Technology, India
[2]Karpagam College of Engineering, India
Corresponding author, e-mail: shriramkv@gmail.com

Abstract

Text preprocessing and document classification plays a vital role in web services discovery. Nearest centroid classifiers were mostly employed in high-dimensional application including genomics. Feature selection is a major problem in all classifiers and in this paper we propose to use an effective feature selection procedure followed by web services discovery through Centroid classifier algorithm. The task here in this problem statement is to effectively assign a document to one or more classes. Besides being simple and robust, the centroid classifier s not effectively used for document classification due to the computational complexity and larger memory requirements. We address these problems through dimensionality reduction and effective feature set selection before training and testing the classifier. Our preliminary experimentation and results shows that the proposed method outperforms other algorithms mentioned in the literature including K-Nearest neighbors, Naive Bayes classifier and Support Vector Machines.

Keywords: *Document processing, Centroid classifier, Ontology alignment, Semantic web, KNN algorithm, web service annotation*

Copyright © 2017 Institute of Advanced Engineering and Science. All rights reserved.

1. Introduction

Services offered by one electronic device to another, communicating through World Wide Web is called as a web service. Web services includes integration of web applications using Extensible Markup Language (XML), Simple Object Access Protocol (SOAP), Web Service Definition Language (WSDL) and Universal Description, Discovery, and Integration (UDDI) open standards. With the advancements in Internet that use the Internet Protocol suite to interlink billions of devices worldwide and going by the prediction of International Telecommunication Unit which expects about 3.2 billion people to be online by 2016, the need for web services discovery and better user experience is also increasing. It also helps in businesses to acclimate swiftly to changes in the business environment and fulfill the needs of different customers worldwide.

Web services provider publishes the service while the consumer will discover to use the provided service. The first step in Web services discovery is to classify the web services in to multiple groups when they are issued in a registry. Web services are converted in to standard vector format through WSDL document. A subset of them is then selected for training the system. Classification algorithms are deployed in order to classify the web services automatically.

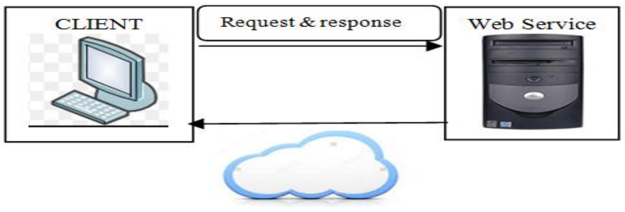

Figure 1. Web Services Illustration

Received September 1, 2016; Revised November 25, 2016; Accepted January 1, 2017

Web service as illustrated in Figure 1 describes a standard way of web based applications using different internet protocol backbone including XML which is to tag data, SOAP to transfer data, WSDL to describe services and UDDI to list what services are available. Defining a web service's domain is needed for web service discovery, composition and semantic annotation. Hence we studied different classification algorithms and identified the best one for web services discovery along with the idea of feature set selection from the data sets used for training and testing. We find that effective feature set selection followed by centroid classifier algorithm works wells for web services discovery when compared with the traditional algorithms.

2. Literature Survey

Data classification has got a wide variety of applications as it tries to distinguish or find the relationship between the feature variables and the target variables of interest. There are two types of classification algorithms – namely the supervised classification and the unsupervised classification. The former refers to the machine learning job of understanding a task from labeled training data while the latter describe hidden structure from unlabeled data. There are multiple classification algorithms available in the literature as discussed below and we find the optimal one along with our novel approach that is best suitable for web services discovery.

Ximing Li, Jihong Ouyang and Xiaotang Zhou [1] has discussed about document classification in their research paper. They feel that the native algorithm suffers from over-fitting and linear inseparability problems and to overcome they have proposed a kernel-based hypothesis margin centroid classifier algorithm.

Mamoun M., et al., [2] proposed a generic non-functional based web services discovery classification algorithm. They have the results proven mathematically and also experimentally. George Potamias [3] and his team have enhanced web services by combining document classification along with the user profile. They have demonstrated the effectiveness of their proposed approach through experimental results.

V. Vineeth Kumar and Dr. N Satyanarayana [4] feel that majority of the web services are missing precise description. To overcome the short comings, they have proposed a self-adaptive semantic classification method by means of service data ontology and web user log frequent patterns. Marco Crasso, et al., [5] has described that semantic web services technologies are not widely adopted due to the efforts required in annotating semantically ordinary services. They have combined text preprocessing, document classification and ontology alignment techniques to extract information from standard service descriptions.

Hongbing Wang along with others [6] have found that the present methods for web service discovery study only for small data set. They have proposed to use a new feature selection method for better classification and their results were convincing as well.

K. Venkatachalam, N. K. Karthikeyan and S. Lavanya [7] has described the need for machine understandability. They have derived an approach for web services descriptions based on the informal user request made in a natural language. E. Karakoc and P. Senkul [8] has described in detail about web composition approach in which a rich set of restraints can be well-defined on the composite service.

Tamer A. Farrag and others [9] feel that the matchmaking between the user requests and semantic web services is the main function in web services discovery mechanism. They have proposed a semantic distance based algorithm for the same.

Adala A., and Tabbane N., [10] detailed that to promote the automation of web services discovery, multiple semantic languages have been created which allows to describe the services functionality in a form understood by the machine using semantic web technologies. They have also provided a solution using discovery framework which enables web services discovery based on keywords in natural language.

Ankolekar, et al., [11] presented DAML-S, ontology for detailing the properties of web services. They provide web services descriptions at the application layer and details what a service can do. They focus on the grounding which connects ontology with XML based descriptions of the web services.

Jyotishman Pathak and others [12] described a framework for ontology based flexible discovery of web services. They convert user queries in to queries that can be processed by a matchmaking engine that knows relevant domain ontologies and web services.

Ramakanta Mohanty and others [13] have employed different classification algorithms for document classification including the neural networks, classification and regression trees etc. They have carried out their experiments on QWS data set and reported their findings for web services classification.

Jyotishman Pathak, et al., [14] have discussed about web services provider and consumer number increase along with the need for effective web services classification tool. Their tool automatically selects a service through filtering and certain user requirements. In the second step they classify the services using formal concept analysis approach and also keep track of the items selected by the user for future usage.

Tamer A. Farrag and team [15] have found a semantic distance based matchmaking algorithm and measured the distance among the user request and the web service tested as a pointer of the degree of relevance amongst them. They have a deep evaluation process carried out to validate this approach as well.

3. Design

Machine learning algorithms build systems that learn from experience. From web services discovery point of view, the examples could be:

a) Vendor interested in knowing consumer group to target for selling the products
b) Doctor interested in identifying the root cause for a particular disease
c) Fans interested in retrieving the pictures of their film stars etc. to name a few.

Classification algorithms are hence needed to study these requirements and come up with a model for prediction. Typically they contain two phases including the training phase and the testing phase. During the training stage, a model is constructed from the training vectors and during the testing phase, the described model will classify the test instance in to one of the trained classes. Data classification can be solved using multiple techniques including decision trees, neural networks, Support Vector Machines (SVM), nearest neighbor and probabilistic methods.

Web services classification can be solved according to the domain considering both the textual description and the semantic annotations. It is a twostep process including the keywords retrieval and the document classification. Though there are multiple algorithms available for document classification, the best ones are discussed in this paper along with our solution for effective web services discovery.

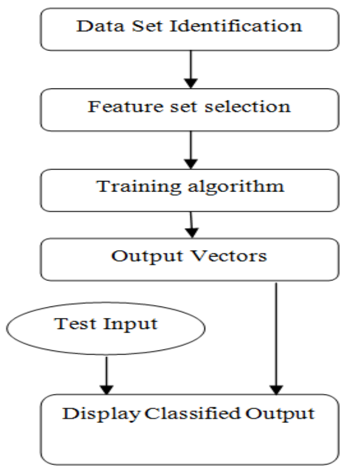

Figure 2. Data Classification flow diagram

Feature selection is the first phase of most classification algorithms as shown in Figure 2. We propose a new approach in feature set selection followed by discussion on three different classification algorithms including the KNN algorithm, Centroid classifier algorithm and Support Vector Machines.

For supervised learning, the following steps needs to be performed. Training data needs to be identified first. A set of input and output objects are gathered either from experts or from measurements. The input objects are converted in to feature vectors and a subset is selected for training. The learning algorithm is then selected and deployed with the identified subset of feature vectors. Run the learning algorithm on the training set and get the required output. Accuracy of the learned function is evaluated and then tested with the new set of test vectors.

The classification error of a learning algorithm is linked to the sum of the bias and the variance of the system. Typically, there is a compromise between bias and variance. A classification algorithm with low bias must be flexible to fit the data well. Dimensionality of the training vectors is a major issue and we address the same in the following sections along with the best classification algorithm for web services discovery.

3.1. K – Nearest Neighbor Algorithm

KNN, expanded as K – Nearest Neighbor Algorithm in pattern recognition is the simplest and most commonly used algorithm for classification. KNN also is deployed at places for estimation and prediction. People also regard it as lazy learning algorithm. KNN is very simple to understand and is having a very vast range of applications. KNN is also referred as instance based learning. The training data set is stowed; the classification for the unclassified record can be achieved by just comparing it to the similar records available in the data set. This algorithm is referred as lazy as it does not make use of the training data points to carry out any generalization. This can also be conveyed as the training phase is non-existent or very negligible. This enables the training phase to be very fast. Since the training is non-existent or very minimal, it has an impact in the time one should spend on testing. The testing phase would need a lot of time and also memory.

Whenever there is a need to classify a new point, one would find its K nearest neighbors from the training data set, adhering very much the name. The distance is normally calculated with one of the following methods as:

a. Euclidean Distance
b. Minkowski Distance
c. Mahalanobis Distance

A simple example is taken as an instance to explain the way k-nearest neighbor's algorithm works, below.

Assume that the query instance is Xp, Let X1, X2, X3, X4 …. XT signify the T instances from the training examples that are in the closer vicinity to Xp. The return would be in such a way that the class represents the maximum of the T instances. Figure 3 represents the KNN instance along with the scenario discussed above.

Assuming T = 6, taking this scenario as shown in above example, the case query instance Xp would be classified as negative. This is based on the simple rule that the classification should be based on nearest neighbors. Here, in this case, four of its nearest neighbors are negative and hence Xp shall also be getting classified as negative, leaving behind positive.

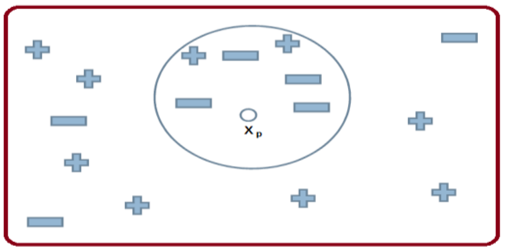

Figure 3. KNN – An Instance

The distance is the key here and it relates to all the features. All the features or attributes are assumed to have the same impact on the distance. While using KNN one should also be informed about the curse of dimensionality. If the classification is carried out wrong which is predominantly due to the presence of irrelevant attributes is defined as curse of the dimensionality. For multi-class k-NN classification algorithm, Cover and Hart (1967) prove an upper bound error rate of:

$$R^* <= R_{KNN} <= R^*(2 – MR^* / (M – 1))$$

Where R* represents the Bayes error rate, R_{KNN} is the k-NN error rate, and M corresponds to the number of classes in the dataset. For M=2, as we see that the Bayesian error rate R* approaches close to zero, this limit decreases to "not more than twice the Bayesian error rate".

Though KNN appears to be good for classification, the main disadvantage is that it is a lazy learner. It uses the training data itself for prediction rather than learning from it. This will slow down the prediction time when the data set is higher. Also the algorithm does not get generalized well and the robustness to noise is also not good. For these reasons, we study the next algorithm called Centroid classifier and use it in web services discovery.

3.2. Centroid Classifier Algorithm

The Centroid of a plane figure is the average of all the data points in the shape. It can be extended to n-dimensional space as well. Automatic text categorization is an important task for us in web services discovery and linear-time centroid based classification algorithm would be best suited for this application due to its simplicity and robust performance.

In data retrieval, ‘tf-idf’ represents term frequency-inverse document frequency which is a measure of importance of a word in a document. It increases proportionately to the number of times a word is present in the text. One of the technique used in data mining to measure cohesion within clusters is the cosine similarity. It measures the orientation and not the magnitude which makes sure that we are not just interested in the ‘tf-idf’ but the angle between the documents. Cosine similarity of 1 indicates same orientation while -1 indicates diametrically opposed ones.

During the training phase of the algorithm, if the labeled training samples are {(x1, y1),…,(xn, yn)} with class labels yi € Y, then we compute the centroids per class as follows:

$$\mu = (1/cl) \sum xi \quad (1)$$

Where c is the set of indices of samples belonging to that particular class.

During the testing phase of the algorithm, we identify the class of the test vector x as follows:

$$Y = \arg\min \|\mu - x\| \quad (2)$$

Where y is the predicted output and µ corresponds to the centroids computed. Figure 4 represents the Nearest Centroid classifier.

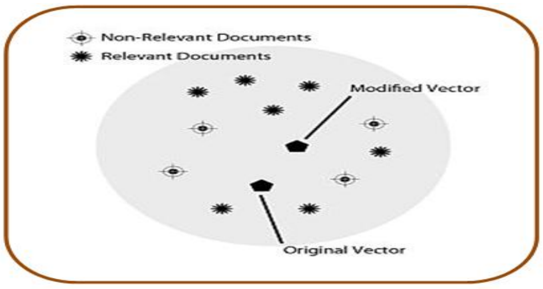

Figure 4. Nearest Centroid Classifier

For web services discovery, the documents are first represented in the vector space before proceeding with the calculations. In other words, every document is denoted by the term frequency values dtf = {tf1, tf2,…,tfn}. The similarity or the distance between the two documents is computed using the cosine function as follows:

$$\cos(di, dj) = di.dj / (\|di\| * \|dj\|) \tag{3}$$

Where "." means the dot product of the two vectors from the different documents.

The centroid classifier algorithm is not complex to compute when compared to most traditional algorithms and in learning phase, it is linear. When a new test vector comes, the amount of time required to find the type of class is O(km) where m represents the number of terms present in the new document x.

3.3. Feature Set Selection for Training

Feature selection is the first step of all classification algorithms. Most time data is collected by non-domain experts and some of them could be irrelevant. These noisy data could result in poor modelling and should be eradicated. Feature selection could be made either through filtering models or through wrapper models. Filtering model selects the subset using intrinsic characteristics of the data while the wrapper model requires an algorithm to find the feature subset. We propose to use a new algorithm for wrapper model and then combine both the filter and wrapper model as shown in Figure 5 for feature subset selection making it a new hybrid model approach in web services discovery.

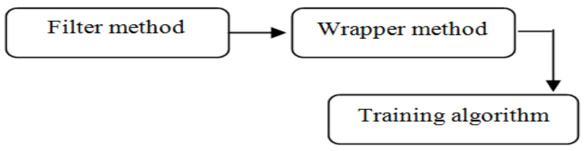

Figure 5. SVM – Hybrid model in feature subset selection

For filter approach, features could be selected based on the statistical properties. They perform two operations including the ranking and subset selection. In ranking, each feature is evaluated without considering the interactions among the other elements of the set and then the subset to be selected is provided. Sometimes filter methods can select the redundant or the noisy variables as well as they do not consider any relationships or models between variables during selection. For these reasons, we go for the next method called wrapper method as the second pass of selecting the feature subset.

The relation between the feature subset selection and the relevance is an important factor in wrapper method. So, in wrapper method, we start with the first input vector and keep adding the features one at a time and test it immediately with the user relevance in the past. If it is relevant, then we add it on feature subset otherwise not. As per the centroid classifier algorithm discussed, the centroids gets adjusted every time a document gets added or removed. This happens regularly during training phase and does not happen frequently during testing.

If there are "N" documents present originally in the set and when a new document (a,b) gets added, then the original centroid (x,y) gets updated as follows:

$$\text{Centroid} = ((Nx + a) / (N + 1), (Ny + b) / (N + 1)) \tag{4}$$

In the similar way, when a document is found to be irrelevant it can be removed from the feature set selected as follows:

$$\text{Centroid} = ((Nx - a) / (N - 1), (Ny - b) / (N - 1)) \tag{5}$$

The wrapper approach is better in terms of performances and at the same time requiring greater computational resources. To avoid this issue, we use the filer method first to reduce the original set from a huge size to a medium size and then deploy the wrapper method on the reduced set for further feature subset selection making both the performance and resource usage better. Feature selection and feature extraction together helps in dimensionality reduction hence.

3.4. Support Vector Machines Based Classification

Support Vector Machines (SVM) are supervised learning models with related learning procedures that explores data used for classification and regression analysis. From the subset selected each marked with one of two categories, SVM algorithm builds a model that assigns test vectors into one class or the other. This is linear classification and SVM is capable of non-linear classification as well just by mapping inputs into high-dimensional feature space through kernel trick.

Support Vector Machines constructs a hyperplane in a high dimensional space which can be used in classification problems further helping in applications like web services discovery. SVM's are useful for large scale data mining applications. Identifying the right hyperplane that best separates the data points is the challenge here. Maximizing the distance between nearest data point and hyperplane called as margin will help us to solve this issue.

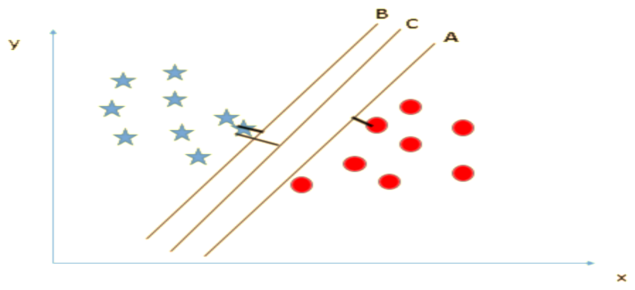

Figure 6. SVM – Hyperplane separating classes

From Figure 6, it is clear that hyperplane C is high when compared with A and B hyperplanes. This also makes the algorithm more robust to outliers. A linear classifier has the form:

$$f(x) = W^TX + b \quad (6)$$

Where W is the normal to the line and b is the bias. W is also known as the weight vector. X refers to the input vector in the above equation. Once the system is trained, the support vectors are sufficient to classify the new test data unlike KNN method where we need to carry the training data throughout classification.

Assuming that the data is perfectly separable, we need to optimize the following for finding the support vectors. Minimize $|| w ||^2$, subject to:

$$(w \cdot x_i + b) \geq 1, \text{ if } yi = 1 \quad (7)$$

$$(w \cdot xi + b) \leq -1, \text{ if } yi = -1 \quad (8)$$

The optimization algorithm that is required to generate the weight vectors will proceed in such a way that the support vectors find the weights as well the boundary.

Though SVM works well with clear margin of separation and effective in high dimensional spaces, it has its own limitations. These cons include performance degradation in large data set due to higher training time and it does not perform well in case of noisy data set.

4. Experimental Results

For testing the different proposed algorithms along with the feature set selection, we have used the OWL data set that is based on the Mooney Natural Language Learning Data. It contains three different data sets including the geography, job and restaurant. We will pick up few examples from the data set to explain the methodology:

a. How big is Texas – Geography domain
b. Are there any big jobs in Alameda? – Jobs
c. Give me some cafes in alameda – Restaurant

Assuming that the above documents are used in training phase and let's name them as d1, d2 and d3 respectively. When we get a new test vector "t" as "Restaurants in Alameda", we observe the following outputs from different algorithms as described below:

1. K-NN Algorithm:

Find the document vectors first for all these documents. We then find the cosine similarity between the test document and the available documents in the web as follows:

$\cos(t, d1) = 0$
$\cos(t, d2) = 1/(\sqrt{4} * \sqrt{3}) = 0.288$
$\cos(t, d3) = 1/(\sqrt{3} * \sqrt{3}) = 0.333$

Cosine similarity is a quantity that is used to find the relationship between two vectors of an inner product space that measures the cosine of the angle between the two vectors. The cosine value of 0° is 1, and for other angles, it is less than one. Cosine similarity value needs to be higher for more proximity. So in this case, the closest document is d3 for the test vector as compared with the other documents. In case of a tie, we need to randomly choose the class and it could be erroneous as well at times which pushes the need for us to test the next algorithm.

2. Centroid Classifier Algorithm:

Let us assume that there are two classes C1 and C2 and the document vectors for both these classes are given as follows:

C1: (2,7), (3,8), (1,6)
C2: (6,4), (8,6), (5,2), (9,4)

We need to find the decision boundary between these two classes so that any new test vector (e.g.: (12, 24)) can be classified based on that separating line. To do that, we first find the centroids as per the algorithm:

$\mu(c1) = (((2+3+1)/3), ((7+8+6)/3)) = (2, 7)$ is the centroid of the first class.
$\mu(c2) = (((6+8+5+9)/4), ((4+6+2+4)/4)) = (7, 4)$ is the centroid of the second class.

Now we need to find the line that separates both the classes as shown in Figure 7. The equation of a straight line is written in the form of $y = mx + c$ where 'm' represents the slope of the line and the place at which this line crosses the y-axis is the constant 'c' value. The slope in this example is found as: $(y2 - y1) / (x2 - x1)$ which is $(7 - 4) / (2 - 7)$ giving the value as $(-3/5)$. So m1 equals to $-3/5$ while m2 equals to $5/3$.

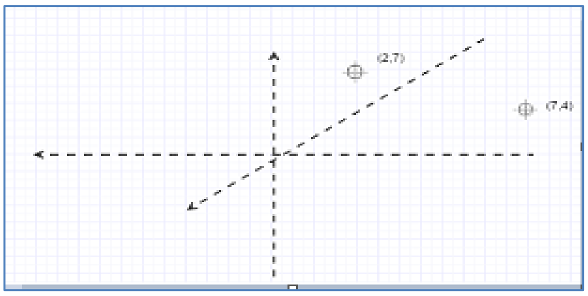

Figure 7. Centroid Classifier output

Thus $y = (5/3)x + c$. The constant value c is calculated as the midpoint or the mean of the centroid values which is (4.5, 5.5). The decision boundary is thus derived in centroid classifier algorithm. We also have the flexibility to adjust the centroids as and when new document gets added to the repository or when legacy documents leave the database.

3. SVM Classifier Algorithm:

Let us assume that there are two classes C1 and C2 for support vector machines classification with the following data set values:

C1: {(0,0) (3,4) (5,9) (12, 1) (8,7) }
C2: {(9,8) (6,12) (10,8) (8,5) (14,8) }

As per the working principle of SVM, we first normalize the data followed by forming the diagonal matrix and error matrix. We then find the augmented matrix as [A -E]. Since this is an optimization problem, the Karush–Kuhn–Tucker (KKT) conditions are applied to find the solution. The support vectors are finally calculated as W = A'DU and gamma value as –e'DU. When a new test vector comes during the testing phase, we find the category of the test vector through:

$$f(x) = \text{sign}\ (W'\ x - \text{gamma}) \quad (9)$$

We can fine tune the results of this SVM classifier by modifying or adjusting the learning constants. Though SVM is good in predicting, we observed that the time taken for training the system is too high as compared to the rest of the algorithms discussed for this application. Figure 8 represents the accuracy levels between the SVM and KNN algorithms where it is quite clear that the SVM outperforms KNN.

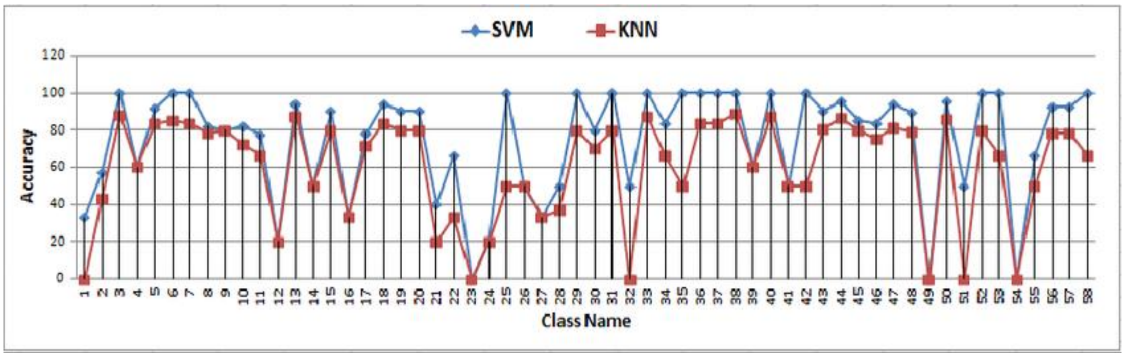

Figure 8. SVM and KNN comparison

We have also studied the time taken to train the system for both these algorithms and found that Centroid classifier algorithm would be much suitable when there are lots of updates happening frequently and training has to happen or adjust accordingly. Fig 9 below represents the training time across algorithms.

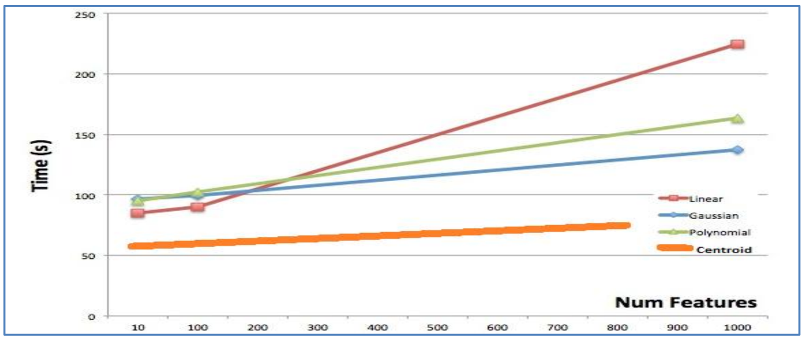

Figure 9. Centroid classifier and SVM training time comparison

We have also measured the precision and recall parameters across algorithms and found that centroid classifier algorithm along with feature set selection excel when compared with the other algorithms.

5. Conclusion

The amount of data available in the internet is huge; especially text documents on Internet, digital libraries etc. have seen a tremendous growth in the past few years. Automatic data classification is hence an important task in web services discovery process. This classification is challenging due to the larger attributes, huge training samples and attribute needs. Data selection or reduction is always a major problem while working with huge data sets. In this paper we have used three different algorithms for classification added with a new solution for feature set selection. We find that with the proposed solution along with centroid classifier algorithm gives better results as compared with the other methods. Moreover the computational complexity of the proposed solution is much cheaper than the other methods in the literature.

Moving forward, we would like to explore further on fine tuning the different features in a supervised learning algorithm. We would also like to extend the solution for mobile devices which have even more constraints added to the complexity of wireless heterogeneous networks.

References

[1] Ximing Li, Jihong Ouyang, Xiaotang Zhou. *A kernel-based centroid classifier using hypothesis margin*. Proceeding of the Journal of Experimental & Theoretical Artificial Intelligence. 2015.

[2] Mamoun M Jamous, Safaai Bin Deris. Web Services Non-Functional Classification to Enhance Discovery Speed. *OALib Journal*. 2011.

[3] George Potamias, Lefteris Koumakis, Vassilis S Moustakis. *Enhancing Web Based Services by Coupling Document Classification with User Profile*. The International Conference on Computer as a Tool, EUROCON. 2005.

[4] V Vineeth Kumar, N Satyanarayana. Self-Adaptive Semantic Classification using Domain Knowledge and Web Usage Log for Web Service Discovery. *International Journal of Applied Engineering Research*. 2016; 11(6): 4618-4622.

[5] Marco Crasso, Alejandro Zunino, Marcelo Campo. Combining Document Classification and Ontology Alignment for Semantically Enriching Web Services. *New Generation Computing*. 2010; 28(4): 371-403.

[6] Hongbing Wang, Shi, Zhou, Xuan Shao, Qianzhao Zhou, Athman Bouguettaya. *Web Service Classification using Support Vector Machine*. 22nd International Conference on Tools with Artificial Intelligence. 2010.

[7] K Venkatachalam, NK Karthikeyan, S Lavanya. *A Framework for Constraint Based Web Service Discovery with Natural Language User Queries*. International Conference on Engineering Technology and Science (ICETS'16). 2016.

[8] E Karakoc, P Senkul. Composing semantic Web services under constraints. *Expert Systems with Applications, Elsevier Journal.* 2009; 36: 11021-11029.

[9] Farrag TA, Saleh AI, Ali HA. Semantic web services matchmaking: Semantic distance-based approach. *Computers and Electrical Engineering*. 2013; 39: 497-511.

[10] Adala A, Tabbane N, Tabbane S. A Framework for Automatic Web Service Discovery Based on Semantics and NLP Techniques. *Advances in Multimedia*. 2011: 1-7.

[11] Ankolekar, et al. *DAML-S: Web Service Description for the Semantic Web*. In Proc. 1st International Semantic Web Conf. Italy. 2002; 2342: 348-363.

[12] Jyotishman Pathak, Neeraj Koul, Doina Caragea, Vasant G Honavar. *A Framework for Semantic Discovery of Web Services*. Proceedings of the 5th International Conference on Ubiquitous and Collaborative Computing. Abertay, Dundee. 2010: 22-27.

[13] Ramakanta Mohanty, V Ravi, MR Patra. Web-services classification using intelligent techniques. *Science Direct*. 2010.

[14] Zeina Azmeh, Marianne Huchard, Chouki Tibermacine. *WSPAB: A Tool for Automatic Classification & Selection of Web Services Using Formal Concept Analysis*. Sixth European Conference on Web Services. 2008.

[15] Tamer A Farrag, Ahmed I Saleh, HA Ali. Semantic web services matchmaking: Semantic distance-based approach. *Computers and Electrical Engineering*. 2013; 39: 497-511.