An Enhanced Text Categorization Technique for Document Classification Using Neural Networks

V.Kumuthavalli
Associate Professor, Department of Computer Science,
Sri Parasakthi College for Women, Courtallam, Tirunelveli, Tamil Nadu, India.
Email: saikumuthavalli@gmail.com
Dr.V.Vallimayil
Associate Professor & Head, Department of Computer Science & Applications,
Periyar Maniyammai University, Vallam, Thanjavur, Tamil Nadu, India.
Email: vallimayilv@gmail.com

Abstract- The text mining gaining more importance recently because of the availability of the increasing number of the electronic documents from a variety of sources. In the current scenario, text classification gains lot of significance in processing and retrieval of text. Due to rapid development of the information technology, large numbers of electronic documents are available on the internet instead of hard copies. Therefore, proper classification and knowledge discovery from these resources is an important area for research. Automated document classification becomes a key technology to deal and organize huge volume of documents and its frees organizations from the need of manually organizing document bases. A traditional approach to text categorization requires encoding documents into numerical vectors. This type of traditional document encoding causes two main problems are huge dimensionality and sparse distribution. This research proposes a new representation of documents, where string vectors are received as input vectors, instead of numerical vectors and a self organized neural network using the representation for its input vector. The experimentation and results with various documents and compared with existing methods and it provides better results.

Keywords- Text Categorization, Support Vector Machine, Neural Networks, Part of Speech,BP,RBF,NN.

1. INTRODUCTION

Today, information have a great value and the amount of information has been expansively growing during last year's. Especially, text databases are rapidly growing due to the increasing amount of information available in electronic forms, such as electronic publications, e-mail and the World Wide Web. There are so much information around us, that it becomes a problem to find those that are related for our necessary. Because of this, there are many databases and catalogues of information classified into many categories, helping the viewer to easily navigate to the information what they would like to obtain. Most of information in the world are texts and here the text categorization comes to the scene.

Data mining has attracted a great deal of attention in the information industry and in society as a whole in recent years, due to the wide availability of huge amounts of data and the imminent need for turning such data into useful information and knowledge. Generally, data mining is the process of analyzing data from different perspectives and summarizing it into useful information.

Text mining has been defined as "the discovery by computer of new, previously unknown information, by automatically extracting information from different written resources". Text mining, which is sometimes referred to "text analytics" is one way to make qualitative or "unstructured" data usable by a computer. Text mining can help an organization derive potentially valuable business insights from text-based content.

1.1 Text Categorization

Text categorization is one of the well studied problem in data mining and information retrieval. Categorization is the process in which ideas and objects are recognized, differentiated and understood. Categorization implies that objects are grouped into categories, usually for some specific purpose. A category illuminates a relationship between the subjects and objects of knowledge. The data categorization includes the categorization of text, image, object, voice etc. With the rapid development of the web, large numbers of electronic documents are available on the Internet. Text categorization becomes a key technology to deal with and organize large numbers of documents. Text categorization is the assignment of natural language documents to one or more predefined categories based on their semantic content is an important component in many information organization and management tasks [1][2]. The techniques of text categorization are necessary for improving the quality of any information system dealing with textual information, although they cover only a fraction of document management.

Recently, automatic text categorization [3] has receiving a great deal of attention because it is becoming impossible to manually classify the large amount of document for the purpose of category-based retrieval. Automatic text categorization is treated as a supervised learning task. The goal of this task is to determine whether a given document belongs to the given category or not by looking at the synonyms or prefix of that category.

2. REVIEW OF LITERATURE

In this research work, discussed about the methodology for feature extraction and document classification. In order to propose this works are have analyzed various Literatures which are very much relevant and helpful to do this work. The literature where are have retrieved and analyzed are presented in the following section.

2.1 Feature Extraction

There are many feature selection methods and text classifiers using machine learning techniques have already been reported by the researchers. Here going to mention some of the representative and popular approaches related to this thesis.

An algorithm to choose noun phrases from a document as phrases has been proposed in [11]. The frequency of noun are the features used in this work. Noun phrases are extracted from a text using a base noun phrase skimmer and an off the-shelf online dictionary.

HaCohen-Kerner et al [24] proposed a model for phrase extraction based on supervised machine learning and combinations of the baseline methods. They applied J48, an improved variant of C4.5 decision tree for feature combination.

Hulth et al [25] proposed a phrase extraction algorithm in which a hierarchically organized thesaurus and the frequency analysis were integrated. The inductive logic programming has been used to combine evidences from frequency analysis and thesaurus.

A graph based model for keyphrase extraction has been presented in [26]. A document is represented as a graph in which the nodes represent terms, and the edges represent the co-occurrence of terms. Whether a term is a keyword is determined by measuring its contribution to the graph.

A Neural Network based approach to phrase extraction has been presented in [27] that exploits traditional term frequency, inverted document frequency and position (binary) features. The neural network has been trained to classify a candidate phrase as key-phrase or not.

A key-phrase extraction program called Kea, developed by Frank et al. [28][29] used the Bayesian learning technique for key-phrase extraction task. A model is learned from the training documents with exemplar key-phrases and corresponds to a specific corpus containing the training documents.

Recently, Kamal Sarkar et al., [12] proposed a neural network based model for key-phrase extraction by extracting noun phrases from a document. The neural network has been trained to classify a candidate phrase as key-phrase or not.

2.2 Text Classifier

Even if many machine learning approaches to text categorization already proposed, here mention some of approaches related to this work. The KNN was initially created by Cover and Hart in 1967 as a genetic classification algorithm. It was initially applied to text categorization by Massand et al at 1992 in [13]. The KNN algorithm is quite simple: given a test documents, and uses the categories of the K neighbors to weight the category candidates.

The Naive Bayes may be considered as another approach to text categorization. It was initially created by Kononenko in 1989, based on Bayes Rule. Its application to text categorization was mentioned in the textbook by Mitchell. Assuming that the Naive Bayes is the popular approach, in 1999, Mladenic and Grobelink proposed and evaluated feature selection methods [14].

Recently, the SVM was recommended as the practical approach to text categorization model. It was initially introduced in Hearst magazine in the same year, it was applied to text categorization by Joachims [15]. Its idea is derived from a linear classifier perceptron, which is an early neural network. The main idea of SVM is that if a distribution of training examples is not linearly separable, these examples are mapped into another space where their distribution is linearly separable. Furthermore, the SVM is popularly used not only for text categorization tasks but also for any other pattern classification tasks.

In 1995, BP was initially applied to text categorization by Wiener in his master thesis [16]. They applied continually BP to text categorization. They used a hierarchical combination of BPs, called HME (Hierarchical Mixture of Experts), to text categorization, instead of a single BP. They compared HME of BPs with a flat combination of BPs, and observed that HME is the better combination of BPs. Since BP learns training examples very slowly, it is not practical, in spite of its broad applicability and high accuracy, for implementing a text categorization system where training time is critical.

Taeho Jo propose a string vector based text categorization model using neural network in [18]. Its property is shared from a linear classifier perceptron, which is an early neural network. He also develop the neural network based approach for train the classifier to categorize the document, even though for implementing a text categorization system where feature dimensionality still quite large.

Research on machine learning based approaches to text categorization has been progressed very much, and they have been surveyed and evaluated systematically. In previous works, dimension of numerical vectors should reserve, at least, several hundred for the robustness of document classification systems. In order to mitigate sparse distribution, a task of text classification was decomposed into binary classification tasks in applying one among the traditional approaches [19]. This requires classifiers as many as predefined categories, and each text classifier judges whether an unseen document belongs to its corresponding category or not.

This research will be a successful attempt to solve the two main problems by encoding the documents into alternative structured data to numerical vectors with a competitive self organizing neural network which received string vectors as its input data because of its advantages.

3. Existing Methodology

3.1 Document Representation

Documents are unstructured data by themselves, they cannot be processed directly by classifier. They need to be encoded into structured data before of submission to classifier. Document encoding is defined as the process of representing documents into structured data. The reason for document encoding is that documents are themselves, unstructured data on which text categorization cannot be applied directly. Therefore, the two subsections discuss the two document encoding techniques currently in use bags of words and numerical vectors.

3.1.1 Bag of Words

The simplest way of encoding documents is to represent them into bags of words by indexing them. A bag of words is a set of words which is unordered and variable in its size depending on the length of the given document, and denoted by di= { wi1, wi2 ,..., wil} where i is an index of a particular document and l is the size of the set, di. Figure 3.1 presents the process of encoding a document into a bag of words.

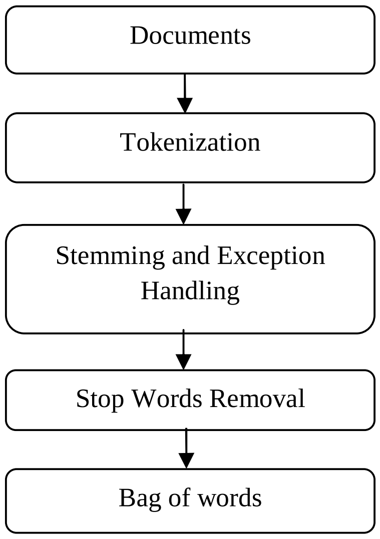


Fig.3.1 Encoding a document into a bag of words

The first step is tokenization, a text is segmented into tokens by white space. Each token is then converted into its root form through the second step, stemming and exception handling. The last step removes stop words, which perform just a grammatical function regardless of the content of the text and stop words include preposition, conjunction, particle, auxiliary verbs. The final output of this process is an unordered list of words representing a document.

3.1.2 Numerical Vector

A traditional strategy of encoding documents for tasks of text categorization is to represent them into numerical vectors. Since input vectors and weight vectors of traditional neural networks such as BP (Back Propagation) and RBF (Radial Basis Function) are given as numerical vectors, and each document should be transformed into a numerical vector for using them for the process of text categorization.

The Fig 3.1 illustrates the process of extracting feature candidates for numerical vectors from documents. If more than two documents are given for categorization, all string of this process is tokenization where the string is segmented into tokens by white space and punctuations. In the second step, each token is stemmed into its root phrase. In the third step, stop words are removed from the collection of words for processing documents more efficiently and reliably for text categorization. Through the three steps illustrated in fig.2, a collection of words are generated as feature candidates.

The features of the numerical vectors representing documents are frequent words occurring in the document collection. If all the words were used as features, then dimension would be more than 10,000 such a dimension is not feasible for text categorization. The words, which are selected as the features of the numerical vectors, are denoted by w1 ,w2,...,wn. There are three common ways to define feature in numerical vectors.

The first way is to use a binary value either, zero or one, for each word; zero indicates its absence and one indicates its presence in the given document. The second way is to use the frequency of each word in each document as its value. In this way, each element is represented as an integer. The third way is to use the weight of each word, wk $(1 \le k \le n)$, as a feature value based on its term frequency and its inverse document frequency. Such a weight is computed from equation (1),

weighti(wk)=tfi(wk)(log2D-log2df(wk)+1) --- (1)

where tfi (wk) is the frequency of wk in the document, di , D is the total number of documents in a given corpus, and df (wk) is the number of documents including the word, wk , in the corpus. Note that the first and second way does not require the reference to a corpus, where as the third way requires the reference for computing elements of numerical vectors using equation (1).

3.1.3 Text Categorization

In this section discussed about text categorization and describe the main traditional approaches, This section covers two existing approaches to text categorization: SVM (Support Vector Machine), and BP (Back Propagation).

3.2 Support Vector Machine

The first approach to classifier training and document classification is Support Vector Machine. This approach is a kernel based supervised learning algorithm for a binary classification problem. SVM defines two hyper planes corresponding to the positive and negative classes in the linear separable feature space mapped from the given original space using a kernel function with maximal margin between them. Support vectors correspond to the training examples that influence the determination of the two hyper planes as the boundaries of the two classes. An item of data, which is targeted for classification, is represented into a numerical vector denoted by x . This vector is classified using equation (2),

f (x) = sign((x · weight) + b) ----(2)

where weight is the weight vector consisting of the parameters of classification, b is the threshold, and sign(.) is the function determining that if the input is more than zero, the output is positive and otherwise, the output is negative. The weight vector, weight is computed using the linear combination of training examples expressed as in equation (3),

$$oj = \sum_{i=1}^{N} \alpha i \, xi \quad ----(3)$$

where α i $1 \le$ i $\le$ N is the Lagrange multiplier corresponding to the training example, x i ,and N is the number of training examples.

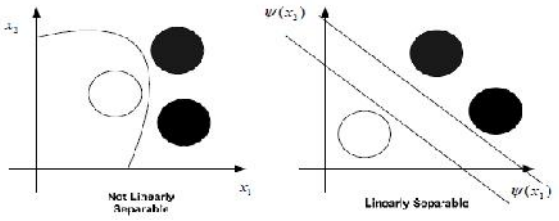


Fig.3.2 Mapping vector space in SVM

Figure 3.2 descibes the process of classifier training and document classification using SVMs. In the process of classifier training, a series of sample documents, which are included in the organization, is given as its input, and the capacity of Lagrange multipliers, C and a kernel function are given as the parameters of SVMs. As a preprocessing step, all the sample documents are encoded into numerical vectors.

For each cluster, a list of these encoded sample documents is partitioned into positive examples, indicating the members of the cluster, and negative examples, indicating the nonmembers of the cluster. The SVM corresponding to each cluster learns these positive examples and negative examples its Lagrange multipliers and its threshold are optimized in classifier training. The final output of this process is a series of SVMs corresponding clusters with their optimized Lagrange multipliers and their thresholds.

The advantage of SVM is that it is tolerant to huge dimensionality of numerical vectors. Its advantage leads to make it very popular approach not only in text categorization, but also other type of classification problems [13]. In spite of the advantage of SVM, it has two demerits. First one is that it is applicable to only binary classification if a multiple classification problem is given, it should be decomposed into several binary classification problems for using SVM. The other one is sparse distribution, which means the inner products of its input vector and training set generates zero values frequently.

3.3 Back Propagation

The second approach to classifier training and document classification is BP (Back Propagation), called MLP (Multiple Layers Perceptron). The reason for referring to MLP as BP is that its parameters for classification are optimized from output layer to input layer, in the backward direction, and this approach will be mentioned as Back Propagation in this proposal. Among supervised neural networks, Back Propagation is the most popular model, because of its easy implementation, its tolerance to noise in the training data, and its application to a wide scope of problems.

Fig.3.3 describes the architecture of back propagation. There exist three layers, input layer, hidden layer, and output layer in the architecture.

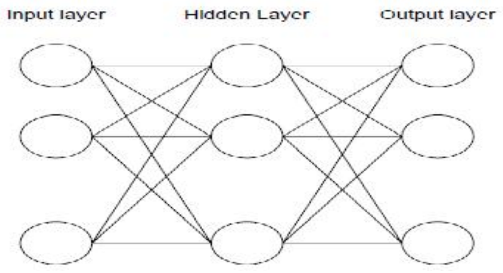


Fig.3.3 Architecture of Back propagation

Two adjacent layers are completely connected to each other, and each connection indicates a weight as a parameter for classification. Learning in BP is the process of optimizing each weight corresponding to each connection to minimize the error between the target category and the classified category of each training example. In the design of BP for a particular classification problem, the number of input nodes corresponds to the dimension of each numerical vector representing a training example; the number of output nodes corresponds to the number of categories; and the number of hidden nodes is determined arbitrary. There is no standard rule of determining the number of hidden nodes. In the role of each layer for classification problems, the input layer receives each input vector for classification or learning, the hidden layer defines the quadratic boundary between categories, and the output layer generates the result of classification.

In BP, two weight matrices are defined. The first one is from the input layer to the hidden layer and the other is from the hidden layer to the output layer. The supervised learning algorithm, BP, optimizes the weights included in these two weight matrices from its learning process, for the minimization made on the error of training data. The error function is defined as the object function that is to be minimized when modifying these weights. The object function indicating training error is expressed by equation (4),

$$E= 1/2\sum_{j=1}^{|o|} (tj-oj)2 \qquad \text{----(4)}$$

where tj is the target value of the input vector, x , corresponding to a category and is given as a binary value, zero or one. One indicates that the input vector, x , is a member of the corresponding category and zero indicates that it is not a member. Note that oj is given as a continuous value between zero and one, unlike the target value, tj .

The process of computing the values of the output nodes and updating all the weights of BP for each sample document is repeated with the iteration number given as a parameter. The final output of classifier training using BP is the two matrices of optimized weights, WEIGHToh and WEIGHThi .

In SVM, the given space is changed into another space where the distribution is linearly separable, whereas in BP, a quadratic boundary is defined by adding one more layer called hidden layer. Since back propagation learns training set very slowly, it is not practical, in spite of its broad applicability and high accuracy, for implementing a text categorization system where training time is critical.

3.4 Proposed Methodology

A major problem of traditional strategy of encoding is the high dimensionality of the feature vector. The feature vector with a large number of key terms is not only unsuitable for neural networks but also easily to cause the over fitting problem. The idea of this research as the solution to the problems is to encode the documents into string vectors and apply it to the self organized neural classifier as a string vector.

The goal of a classifier is to assign a category to given unseen documents. In general, the processing of automatic text categorization involves two important problems. The first is the extraction of feature terms that become effective keywords in the training phase, and the second is the actual classification of the document using these feature terms in the test phase. In the present work, we refer to the former as a feature selection stage and the latter as a document classification stage.

3.4.1 String Vectors

In this section presents an alternative representation of documents to the traditional ones. In first defines the representation in detail, and it then discusses its advantages over the traditional representations, for encoding documents. However, this strategy is applicable to only neural based text classifier, while the previous one is applicable to any traditional machine learning algorithm.

Discovering good features for a classification task is very much an art. The feature candidates identification is an important step in features of string vectors extraction task. Features of string vectors are defined as properties of words to the given document. The following sub-section discusses how to identify noun phrases based feature candidate extraction.

3.4.2 Noun Phrase Identification

To identify the noun phrases (those are becomes a candidate phrases), documents should be tagged. The articles are passed to a POS tagger to extract the lexical information. Through the steps illustrated in figure 3.4, a collection of words are generated as feature candidates.

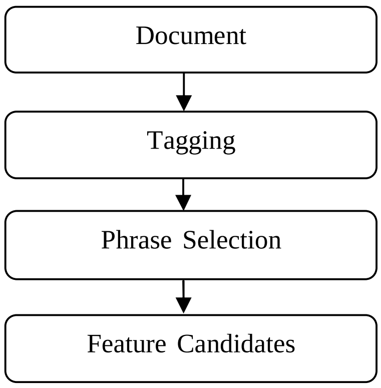


Fig.3.4 Process of encoding a document to extract a bag of words

Figure 3.5 shows a sample output of the tagger for the following text segment:

The idea behinds of grouping, or clustering is simple in its nature and is close to the way of human thinking whenever we are presented with a huge amount of data, generally often summarize this large number of data into a small number of groups in order to further facilitate its analysis. In simple words, a cluster is a collection of objects which are similar and dissimilar between them to the objects belonging to other clusters.

The_DT idea_NN behinds_NNS of_IN grouping_NN ,_, or_CC clustering_NN is_VBZ simple_JJ in_IN its_PRP$ nature_NN and_CC is_VBZ close_JJ to_TO the_DT way_NN of_IN human_JJ thinking_NN ;_: whenever_WRB we_PRP are_VBP presented_VBN with_IN a_DT huge_JJ amount_NN of_IN data_NNS ,_, we_PRP generally_RB often_RB summarize_VBP this_DT large_JJ number_NN of_IN data_NNS into_IN a_DT small_JJ number_NN of_IN groups_NNS in_IN order_NN to_TO further_JJ facilitate_VB its_PRP$ analysis_NN ._. In_IN simple_JJ words_NNS ,_, a_DT cluster_NN is_VBZ a_DT collection_NN of_IN objects_NNS which_WDT are_VBP similar_JJ and_CC dissimilar_JJ between_IN them_PRP to_TO the_DT objects_NNS belonging_VBG to_TO other_JJ clusters_NNS ._.

Fig.3.5 Sample output of the tagger

In figure 3.5, NN,NNS,JJ,DT,VB,IN,PRP,WDT,CC etc. are lexical tags assigned by the tagger. The meanings and the example of the tags are as follows:

Tag	Meaning and Example
CC	Conjunction (and, or)
NN	Noun, Common Singular (Car)
NNS	Noun, Common Plural (Cars)
NNP	Noun, Proper Singular (America)
NNPS	Noun, Proper Plural (Americas)
IN	Preposition (on, of)
DT	Determiner (a, the)
TO	Infinitive Marker (to)
VBD	Verb, Past tense (went)
VBP	Verb (am, are)
VBG	Verb +ing (going)
VBN	Verb Past participle (gone)
VBZ	Verb, -s (goes, is)
VB	Verb, base (go, be)
JJ	Adjective, general (near)
SYM	Symbol of Formula (US$200)
WDT	det, wh- (what, which)
PRP	Pronoun Personal (I, he)
PRP$	Pronoun, Possessive (my, his)
CD	Number (Four, Fourth)

This is not the complete tag set. The above mentioned tags are some examples of tags in tag set. Features of string vectors can define in various, but in this work, features of string vectors are defined based on only frequencies of words for implementing easily and simply the module of encoding documents into string vectors. According to the importance of the terms in the documents measurements must be made for each term in the vocabulary for producing the set of initial features from preprocessed term. This involves assigning each term a weight indicating the relative importance of the term in a document. If a noun phrase is occurring more frequently in a document, the phrase is assumed to more important in the document. Number of times a phrase occurs independently in a document with its entirety has been considered as the phrase frequency.

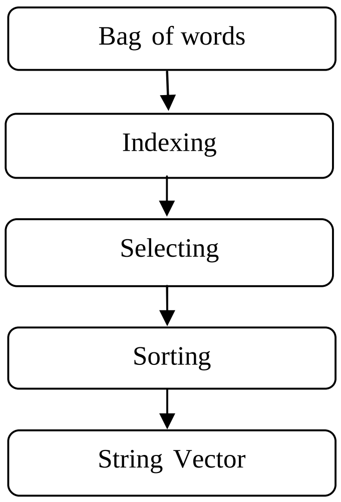


Fig.3.6 Process of mapping a bag of words into a string vector

Figure 3.6 illustrates the process of encoding a document into its string vector with the simple definition of features. In the first step a document is indexed into a list of words and their frequencies. Its detail process of the first step is illustrated in figure 3.4.

If the dimension of string vectors is set to d, highest frequent (repeated) words are selected from the list, in the second step. Continuously in the third step, the selected words are placed in the descending order of their number of frequencies. This ordered list of words becomes a string vector, which are given as the input to the classifier. A string vector is a finite ordered set of words. In other words, a string vector is defined as a vector whose elements are strings, instead of numerical values. Note that a string vector is different from a bag of words, although both of them are similar as each other in their appearance. A bag of words is an infinite unsorted set of words the number of words is variable and they are independent of their positions. But in string vectors, words are dependent on their positions as elements, since words correspond to their features.

The differences between bag of words, numerical vectors and string vectors are illustrated in Table 1.

	Bag of Words	Numerical Vector	String Vector
Elements	Word	Numerical value	String
Similarity measure	Number of shared word	Inner products Euclidean Distance	Semantic Similarity
Attributes	Words	Words	Property of words

Table 3.1 Comparison of bag of words, numerical vectors and string vectors

There are three advantages in representing documents into string vectors. The first advantage is to avoid the aforementioned two main problems completely: the huge dimensionality and the sparse distribution. The second advantage is that string vectors are characterized as more transparent representations of documents than numerical vectors it is easier to guess the content of documents only from their representations. The third advantage is that there is the potential possibility of tracing more easily, as to why documents are classified so. However, in order to use string vectors more freely, it is necessary to set foundations that are more mathematical.

3.3.2 Neural Based Classifier

This section describes the supervised neural network, this competitive self organizing neural networks belong to a class of recurrent networks, and-they are based on algorithms of unsupervised learning, using string vectors as its input vectors with respect to its architecture, initialization, training and classification.

The competitive self organizing neural network follows Perceptron in that synaptic weights are connected directly between the input layer and the output layer, and the defined weights are updated only when each training example is misclassified. However, note that neural based classifier is different from Perceptron in context of its detail process of learning and classification, which uses string vectors as its input vectors, instead of numerical vectors.

In the process of classifier training, the input is a series of the documents included in the organization given as sample documents like other approaches. In this approach, the features of a string vector should be defined manually before encoding sample documents. This process is applied to all the string vectors encoding the sample documents, and is iterated with the fixed number of category given as a parameter. After this iteration, the process of classifier training generates the optimized weights of the words in each learning node as its output.

The learning layer is a competitive layer, which is given as an additional layer to the input and the output layer instead of the hidden layer of back propagation with respect to its role. In this layer, the output neurons of a neural network compete among themselves to become active. Whereas in MLP several output neurons may be active simultaneously, but in this model only a single output neuron is active at any time.

In the competitive neural network, each example is classified by summing the winner optimized weights, whether it is a training or unseen example. In addition weights connected to itself from the input nodes as its categorical score. The competitive layer determines synaptic weights between the input and the output layer by referring to the tables owned by learning nodes. The learning of neural based classifier refers to the process of optimizing weights stored in the tables.

Figure 3.7 shows the diagram of self organizing neural based text classifier. Complete algorithm of self organized neural based classifier and competitive learning algorithm was mentioned in classifier training algorithm and learning algorithm, respectively.

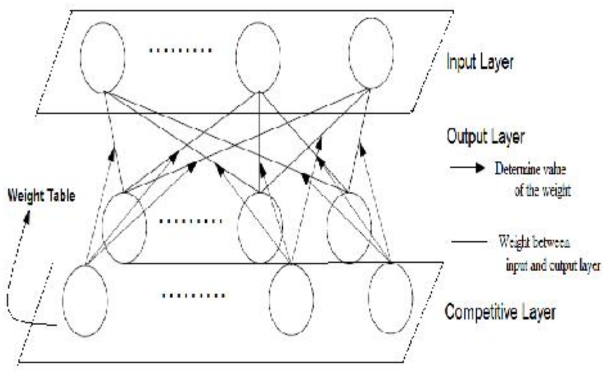

Input Layer
Output Layer
Determine value of the weight
Weight between input and output layer
Competitive Layer
Weight Table

Fig 3.7 Architecture of self organizing neural based text classifier

The output layer generates the degree of membership of the input vector encoding a document to each category.

The conditions for designing the architecture of neural based text categorizer are given as follows.

• The number of input nodes should be identical to the dimension of the string vectors encoding documents

• The number of competitive nodes should be identical to the number of given categories.

• The number of output nodes should be identical to the number of given categories.

The nodes in the competitive layer "compete" with each other, and the node that has the largest output becomes the "winning" neuron. The winning neuron is set to true and all other neurons are set to false.

The learning of neural based classifier follows its initialization. An input string vector is denoted by x =[t1, t2,..... td] where ti, 1≤ i ≤ d , is a word given as an element of the input string vector, x, and d is the dimension of the given string vector, x. A set of the given predefined categories is denoted by C=[c1, c2... c|c|]. The weigh, wji denote the weight connected between an input node, i and an output node corresponding to the category, cj 1≤ j ≤ |C |. The value of the term weight, wji, is defined, using equation (2),

tablej (ti) if there is the word in the table
Wji =
---- (5)
0 otherwise

where tablej denotes the table owned by the learning node corresponding to the category, cj and tablec(tj) means the weights of the string, ti, stored in the table, tablej. The weight, wji, means the membership of the word, ti, in the category, cj. Therefore, if there is the word, ti, in the table, tablej, the weight, wji, is fetched from the table, tablej. Otherwise, the weight, wji becomes zero.

The value of each output node is computed using the equation (7) with the current weights. The output node corresponding to the target category is denoted by ot , and the output node corresponding to the classified category is denoted by oc. The rule for updating weights is expressed by equation(6) ,

weightt (wji) = weightt (wji) + η weightt (wji)
if ot $\neq$ oc ---- (6)
weightc (wji) = weightc (wji) - η weightc (wji)

where weightt (wji) indicates the weight of the word, wji , in the target category, given as an element of the string vector, and weightc (wji) indicates its weight in the classified category. Equation (6) indicates that the weights of the given input vector are adjusted by increasing its weights in its target category and reducing those in its misclassified category.

We compute the value of the output node, oj , the output node corresponding to the category, cj , using equation

oj= $\sum_{j=1}^{|C|} \sum_{i=1}^{d}$ wji ---- (7)

The value of oj means the membership of the given input vector, x in the category, cj . Since values of output nodes are combined by linear combination of weights illustrated in equation (7), the neural network is similar as Perceptron. This is the first property shared with Perceptron. The category corresponding to the output node which generate its maximum categorical score (winner category) is decided as the category of the given example. Therefore, the output of this process is one of the predefined categories, assuming that the competitive neural network is applied to text categorization without the decomposition.

Algorithm -I

Input : A Series of Predefined Documents, Number of Predefined Categories

Output : Optimized weights in each corresponding learning node with its feature string vector

```
    //Classifier Training and Testing //
Step1:  Encode series of documents into string vectors
Step 2: Design the architecture of competitive self organized text
        classifier
Step3:  Loop through the list of encoded training and testing
        document
    3.1:In competitive layer initialize weights in each learning
        node with its string vector within its corresponding
        category.
Step 4: Loop through the list of encoded testing document
    4.1: Compute the values of winner nodes of the encoded
         document with the current weight using equation (7)
    4.2: Classify each training vector into the corresponding
         category with its highest value
    4.3: If the winner node classify the documents correctly go
         to step 5, else
        4.3.1: Update table weights using the equation (6)
Step 5: Output calculated weights
Step6: End
```

Algorithm -II

Input : A Series of unseen document and the optimized weights in each learning node

Output: Selected Winner Category

```
    //Winner Category Selection //
Step 1: Encode series of documents into string vectors
Step 2: Compute the value of noun terms using equation (5)
Step 3: Compute the output value of nodes in the encoded
        document using the equation (7)
Step 4: Classify the unseen string vector into the category
        corresponding to the output node (winner node) with
        its highest value
Step 5: End
```

4. EXPRERIMENTATION & RESULTS

4.1 Introduction

The proposed methodology is experimented with manually copied content to the text file from multiple website related to the three categories that are Computer Science, Sports and Medicine, which is an unstructured data in the form of text represent the number of document for each category.

To implement our system, in the construction of categorical data, 100 articles are used to each category. Therefore, 300 articles are used totally related to three predefined category as on previously mentioned.

4.2 Implementation in C#.Net

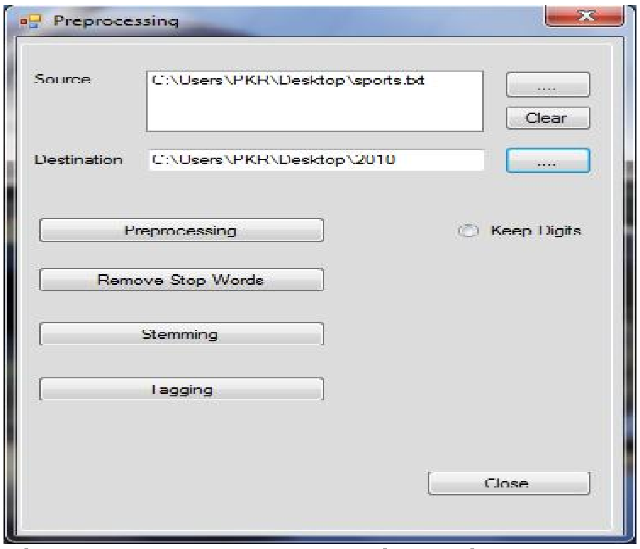
C# was developed to bring rapid development to C++ without sacrificing the power and control of C and C++. C# supports all the features of object oriented language such as encapsulation, inheritance and polymorphism . It treats everything as an object and there are no global functions, variables and constants. It enables you to create a variety of applications and components.

Sql Server:

Microsoft SQL Server is a relational database management system developed by Microsoft Inc. As a database, it is a software product whose primary function is to store and retrieve data as requested by other software applications, be it those on the same computer or those running on another computer across a network (including the Internet).

4.3 Test Bet TB1

In this experiment use manually copied content to the text file from multiple website related to the three previously mentioned categories, which is an unstructured data that are converted to structured data using preprocessing tool shown in figure 4.1 in the form of text represent the number of document for each category.


Preprocessing

Source C:\Users\PKR\Desktop\sports.txt

Destination C:\Users\PKR\Desktop\2010

Keep Digits

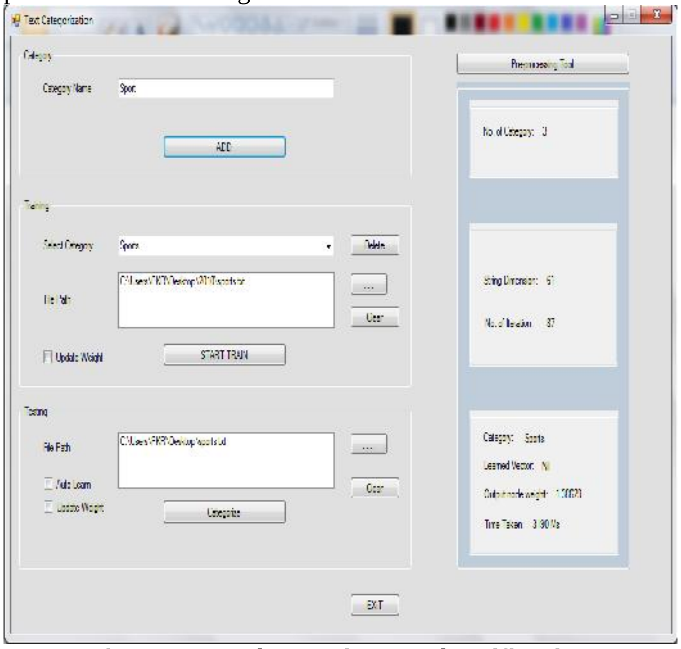
Fig.4.1 Document Preprocessing and Feature Extraction

Test Bet TB1 contains 300 text files related to three predefined categories. Therefore 300 articles are used totally shown in table 4.1. In this research 40% data set is used testing and 60% for training:

Cate gory	Number of training document	Number of testing document	Total
Computer Science	60	40	100
Sports	60	40	100
Medicine	60	40	100

Table 4.1. Training set and Testing set.

This process is applied to all the preprocessed documents and iterated with the fixed number of category given as a parameter shown in figure 4.4.


Text Categorization

Category

Category Name Sport

Training

Select Category Sports

File Path C:\Users\PKR\Desktop\2010\sports.txt

Update Weight

Testing

File Path C:\Users\PKR\Desktop\sports.txt

Auto Learn

Update Weight

No of Category: 3

String Dimension: 61

No. of Iteration 87

Category: Sports

Learned Vector: N

Output node weight: 1.00020

Time Taken 3.90 Ms

Fig.4.4 Neural Based Text Classification

After this iteration, the classifier classify the given unseen documents to under appropriate category based on its output weight. The classifier output on sample testing document set shown in figure 4.5.

Fig.4.5 A result of categorized unseen test document set

5.PERFORMANCE EVALUATION

This section is concerned with the empirical validation of the performance of proposed method in several experiments. To evaluate the overall performance, the system is implemented using C#.net with Sql Server 2008 as a data base. The evaluation process includes two steps. First, the documents are passed into POS tagger to identify the noun phrases as a feature candidate, after this feature reduction the output documents are given as a input to the neural based text classifier to categorize the documents based on its content.

5.1 Dimensionality Reduction

Table 5.1 shows the comparison result of existing method and proposed method of document representation to extract valuable feature candidates from raw document. The result shows the percentage of words rejected as a invalid feature candidates from total number of words in the given document.

Methods	Computer science	Sports	Medicine
Stemming and Stop word	71.04	68.67	64.89
Noun Phrase	78.08	75.78	73.40

Table 5.1 Comparison result of feature candidate Extraction

An important issue of text categorization is how to measure the performance of the classifiers. Many measures have been used, each of which has been designed to evaluate some aspect of the categorization performance of a system [19]. In the experiments, three approaches, SVM, Back Propagation, and neural based classifier are evaluated as the approaches to text categorization.

5.2. Micro and Macro Averaging

For evaluating performance average across categories, there are two conventional methods, namely macro-averaging and micro-averaging.

use a,b,c,d to donate the number of different document, as listed below,

a - is the number of document in the positive category that have that string.

b - is the number of document in the positive category that do not have that string.

c - is the number of documents in the negative category that have that string.

d - is the number of document in the negative category that do not have that string.

Macro-averaged performance scores are determined by first computing the performance measures per category and then averaging these to compute the global means.

Micro-average performance scores are determined by first computing the totals of a, b, c, and d for all categories and then use these totals to compute the performance measures. Performance scores are determined by first computing the totals of a, b, c, and d for all categories and then use these totals to compute the performance measures.

Approaches to Text Categorization	Micro Average	Macro Average
SVM	0.60	0.55
NNBP	0.80	0.85
Our Method	0.75	0.85

Table 5.2 Results of Micro and Macro average

The results of this experiment are presented in table 5.2. Even if the macro-averaged proposed neural network is not better than NNBP in the task, both are comparable to each other with respect to the performance of text categorization.

Even though NNBP that it requires very much time for training as the payment for the best performance. In addition, the NNBP is not practical in dynamic environments where NNBP must be trained again, very frequently. Hence, the proposed method is more recommendable than NNBP with respect to both the learning time and the performance.

5.3. Precision and Recall Rate

Precision and recall are widely used for evaluation measures in text categorization [20]. Precision is a measure of the accuracy provided that a specific class has been predicted. Recall is a measure of the ability of a prediction model to select instances of a certain class from a data set. It is commonly also called sensitivity, and corresponds to the true positive rate. So we need to define the recall and precision rate with the parameters that defined in previous as:

$$Precision = \frac{a}{a+b}$$

$$Recall = \frac{a}{a+c}$$

Approaches to Text Categorization	Precision	Recall
SVM	0.60	0.4
NNBP	0.55	0.4
Our Method	0.85	0.8

Table 5.3 Results of Precision and Recall.

Table 5.3 shows the complete recall and precision rate on given document collection. In SVM experiment, precision and recall are low in some categories. And NNBP takes more training time than neural based classifier model.

From the above table shown the experimental results of the proposed methodology and the existing system among the obtained results the proposed methodology is pictorial better recognition rate by sampling with the existing methodology using neural network. The pictorial representation of the proposed methods by comparing with the existing methodology presented in below

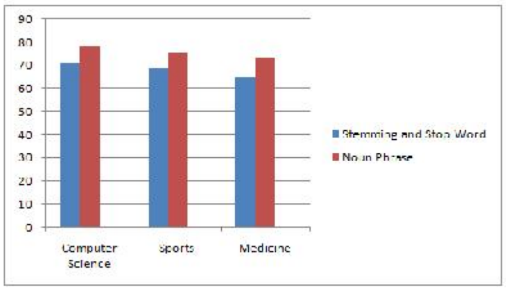


Fig 5.1 Result of feature candidate Extraction on TB1

From the above show the graphical representation of the result which are different in the early table 5.1 result of proposed method and existing method of valuable feature candidate selection for efficient classification to obtain reducing training and testing iteration in terms of document classification on TB1.

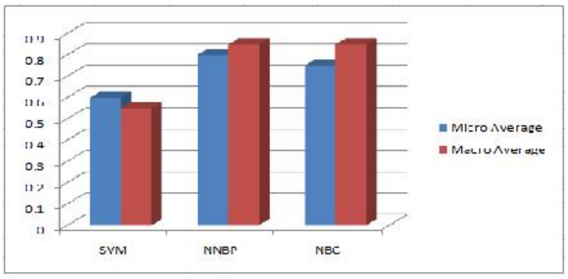


Fig 5.2 Comparison Results of Micro and Macro average on TB1

From the above show the graphical representation of the result which are different in the early table 5.2, the result shows the performance of neural based classifier is comparable with NNBP in terms of Micro and Macro averaging.

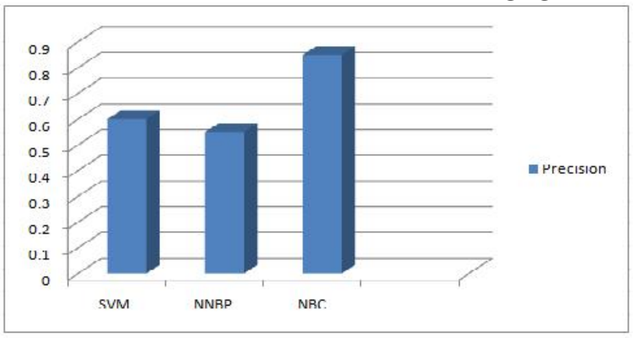


Fig 5.3 Comparison Results of Precision on TB1

From the above show the graphical representation of the result which are different in the early table 5.3, the result shows the performance of neural based classifier in terms Precision on TB1. The method can over perform the traditional method with classify precision rate of 0.8.

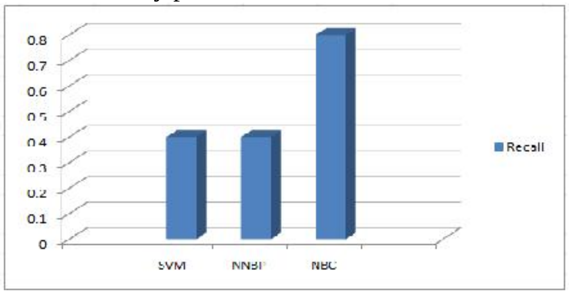


Fig 5.3 Comparison Results of Recall on TB1

From the above show the graphical representation of the result which are different in the early table 5.3, the result shows the performance of neural based classifier in terms Recall on TB1. This picture shows the robustness and quality of text categorization by the competitive self organizing neural based classifier. The method can over perform the traditional method with classify recall rate of 0.75.

6.CONCLUSION

This research presented work on the effective neural network algorithm for text classification, 150 text document belonging to 3 categories i.e. computer science, sports, and medicine have been encoded by POS tagger for to extract noun phrase. In this method of document representation used a full inverted index as the basis for the operation on string vectors, instead of a restricted sized similarity matrix. As well as which is better than performing traditional approach to represent the document by minimizing document preprocessing time and feature dimensionality. Furthermore, competitive self organizing neural based classifier, which receives string vectors as its input vector. It provides the potential easiness for tracing why each document is classified under the category. Neural based classifier worked significantly better than two mentioned traditional approached. Finally, this work conclude that it is more recommendable to encode documents into string vectors by extracting noun phrase than into numerical vectors.

REFERENCES

[1] M.-L.Antonie and O.R.Za¨ıane, “Text document categorization by term association”, In Proc. of the IEEE 2002 “International Conference on Data Mining”, pp.19–26, Maebashi City, Japan, 2002.

[2] K.Androutsopoulos, K.V.Koutsias, Chandrinos, C.D. Spyropoulos, “An Experimental Comparison of Naïve Bayes and Keyword-based Anti-spam Filtering with Personal Email Message”, Proceedings of 23rd ACM SIGIR, pp.160-167, 2000.

[3] V.I. Frants, J. Shapiro, V.G. Voiskunskii, ―Automated Information Retrieval: Theory and Methods‖, Academic Press, 1997.

[4] F.Sebastiani, “Machine Learning in Automated Text Categorization”, ACM Computing Survey, Vol.34, No.1, pp.1-47, 2002.

[5] Martinez-Arroya, M(2006). “Learning on optimal Naïve Bayes Classifier”,p(1236-1239)

[6] Sapan Kevych,N(24-38).”Time Series prediction using support vector machines A survey” ,p(24-38)

[7] Nitin Mathuriya, Ashish Bansal ”Applicability of Backpropagation neural network for recruitment data mining vol-1. Issue 3,2012.

[8] Zhihang chen, Chengwe Ni and Yil. “Neural network approaches for text document categorization” p(1050-1060)

[9] Miguel E.Ruiz, Padmini Srinivasan “Automatic Text Categorization using neural networks “Advance in classification research vol III.

[10] V.Srividhya, Anitha “Evaluating Preprocessing Techniques in Text Categorization” Proc. International journal of computer science and application Issue .2010

[11] K.Barker, N. Cornacchia, Using Noun Phrase Heads to Extract Document Keyphrases. In H. Hamilton, Q. Yang (eds.): Canadian AI 2000. Lecture Notes in Artificial Intelligence, 2000, Vol. 1822, Springer-Verlag, Berlin Heidelberg, 40 – 52.

[12] Kamal Sarkar, Mita Nasipuri and Suranjan Ghose "A New Approach to Keyphrase Extraction Using Neural Networks" International Journal of Computer Science Issues, Vol. 7, Issue 2, No 3, March 2010.

[13] Massand, B., Linoff, G., Waltz, D. (1992). Classifying News Stories using Memory based Reasoning, In: the Proceedings of 15th ACM International Conference on Research and Development in Information Retrieval, p. 59-65.

[14] Mladenic, D., Grobelink, M. (1999). Feature Selection for unbalanced class distribution and Naïve Bayes. In: the Proceedings of International Conference on Machine Learning, p. 256-267.

[15] Joachims, T. (1998). Text Categorization with Support Vector Machines: Learning with many Relevant Features, In: the Proceedings of 10th European Conference on Machine Learning, p. 143-151.

[16] Wiener, E.D. (1995). "A Neural Network Approach to Topic Spotting in Text", The Thesis of Master of University of Colorado.

[17] Ruiz, M.E., Srinivasan,P. (2002). "Hierarchical Text Categorization Using Neural Networks", Information Retrieval, 5 (1) 87-118.

[18] Taeho Jo. (2010). "Representation of Texts into String Vectors for Text Categorization", Journal of Computing Science and Engineering, Vol. 4, No. 2, June 2010, Pages 110-127.

[19] Yang, Y. (1999). "An evaluation of statistical approaches to text categorization, Information Retrieval", 1 (1-2) 67-88.

[20] Y.B.Wu, Q.Li, "Document keyphrases as subject metadata: incorporating document key concepts in search results", Journal of Information Retrieval, 2008, Volume 11, Number 3, 229-249

[21] H. Liu, MontyLingua: "An end-to-end natural language processor with common sense", 2004, retrieved in 2005 from web.media.mit.edu/~hugo/montylingua.

[22] D. Isa, L. Lee, "Text Document Preprocessing with the Bayes Formula for Classification Using the Support Vector Machine", IEEE Transactions on Knowledge and Data Engineering, Vol.20, No.9 pp.1264-1272, 2008.

[23] K. Toutanova, F. Chen, K. Popat, and T. Hofmann, "Text classification in a hierarchical mixture model for small training sets", Proc. ACM Conf. on Information and Knowledge Management (CIKM), pp.105-113, 2001.

[24] Y. HaCohen-Kerner, Automatic Extraction of Keywords from Abstracts, In V. Palade, R. J. Howlett, L. C. Jain (eds.): KES 2003. Lecture Notes in Artificial Intelligence, 2003, Vol. 2773,Springer-Verlag, Berlin Heidelberg, 843 – 849.

[25] A. Hulth, J. Karlgren, A. Jonsson, H. Boström, Automatic Keyword Extraction Using Domain Knowledge, In A. Gelbukh (ed.): CICLing 2001. Lecture Notes in Computer Science, 2001, Vol. 2004, Springer-Verlag, Berlin Heidelberg, 472 – 482.

[26] Y. Matsuo, Y. Ohsawa, M. Ishizuka, KeyWorld: Extracting Keywords from a Document as a Small World, In K. P. Jantke, A. shinohara (eds.): DS 2001. Lecture Notes in Computer Science, 2001, Vol. 2226, Springer-Verlag, Berlin Heidelberg, 271– 281.

[27] J. Wang, H. Peng, J.-S. Hu, Automatic Keyphrases Extraction from Document Using Neural Network., ICMLC 2005, 633-641

[28] E. Frank, G. Paynter, I. H. Witten, C. Gutwin, C. Nevill-Manning, Domain-specific keyphrase extraction. In proceeding of the sixteenth international joint conference on artificial intelligence, 1999, San Mateo, CA.

[29] I. H. Witten, G.W. Paynter, E. Frank et al, KEA: Practical Automatic Keyphrase Extraction, In E. A. Fox, N. Rowe (eds.): Proceedings of Digital Libraries'99: The Fourth ACM Conference on Digital Libraries. 1999, ACM Press, Berkeley, CA , 254 – 255.