A Novel Fuzzy Based Clustering Algorithm for Text Classification

A. Krishna Mohan[†] **V.V.Narasimha Rao**[††] **MHM Krishna Prasad**[†††]

[†]Department of Computer & Engineering J.N.T.U Kakinada
[††]Department of Computer Science & Engineering J.N.T.U. Kakinada
[†††]Department of Computer Science Science & Engineering Science & Engineering J.N.T.U Vijayanagaram

ABSTRACT
Due to the flourish of World Wide Web and the rapid development of the Internet technology, the increasing volume of digital textual data become more and more unmanageable, therefore the importance of text classification has gained significant attention. Text classification pose some specific challenges such as high dimensionality with each document (data point) having only a very small subset of them and representing multiple labels at the same time. Feature clustering is a powerful method to reduce the dimensionality of feature vectors for text classification. Many researchers worked on Feature Clustering for efficient text classification. Recently a Fuzzy based feature clustering was proposed in which Gaussian distribution is used for fuzzy membership function for clustering. But the problem of skewness may occur with this distribution. To overcome that we propose an efficient Fuzzy similarity based membership function for efficient clustering and with this proposed algorithm satisfactory results obtained.

Keywords
Dimensionality reduction, Skewness, feature extraction, fuzzy clustering, split normal distribution.

1. INTRODUCTION

Text classification is different from conventional classification approaches in construction of text documents. The dimensionality for text data is very large in comparison to other forms of data sets. Also each document may contain only a few of the features from the entire pool of feature set. Recently, text data processing approaches have attracted more and more attention. These approaches have to deal with a big difficulty of a large number of features involved. For example, two real-world data sets, 20 Newsgroups and Reuters21578 top-10, both have more than 15,000 features. Such high dimensionality is a severe obstacle for classification algorithms [9]. To alleviate this difficulty, feature reduction approaches are applied before document classification tasks are performed.

Two major approaches, feature selection and feature extraction, have been proposed for feature reduction. The feature selection methods select a subset of the original features and the classifier only uses the subset instead of all the original features to perform the text classification task. A well-known feature selection approach is based on Information Gain [12], which is an information-theoretic measure defined by the amount of reduced uncertainty given a piece of information. However, there are some problems associated with the feature selection based methods. Only a subset of the original words is used. Useful information that can be provided by the unused words may be ignored.

The feature extraction methods convert the representation of the original documents to a new representation based on a smaller set of synthesized features. Feature clustering [1, 4, 5, 3, 2, 10] is one of effective techniques for feature extraction. The idea of feature clustering is to group the words with a high degree of pair wise semantic relatedness into clusters and each word cluster is then treated as a single feature. In this way, the dimensionality of the features can be drastically reduced. The first feature extraction method based on feature clustering was suggested by Baker and McCallum [1] derived from the 'distributional clustering' idea of Pereira et al. [7]. An Information Bottleneck approach was proposed by Tishby et al. [2, 10] and showed that feature clustering approaches are more effective than feature selection ones. A Divisive Clustering (DC) method was proposed by Dhillon et al. [4], which is an information-theoretic feature clustering approach and more effective than other feature clustering methods. In these methods, each new feature is generated by combining a subset of the original words. A word is assigned to a subset if the similarity of the word to the subset is greater than those to other subsets, despite the distinction is very small. All the feature selection and extraction methods mentioned above require the number of new features be specified in advance by the user.

Later Jung-Yi Jiang, Ren-Jia Liou, and Shie-Jue Lee propose a fuzzy similarity-based self-constructing feature clustering algorithm [13], which is an incremental feature clustering approach to reduce the number of features for the text classification task. The words in the feature vector of a document set are represented as distributions, and processed one after another. Words that are similar to each. other are grouped into the same cluster. Each cluster is characterized by a membership function with statistical mean and deviation. If a word is not similar to any existing cluster, a new cluster is created for this word.

Manuscript received May 5, 2013
Manuscript revised May 20, 2013

Similarity between a word and a cluster is defined by considering both the mean and the variance of the cluster. When all the words have been fed in, a desired number of clusters are formed automatically. We then have one extracted feature for each cluster. The extracted feature corresponding to a cluster is a weighted combination of the words contained in the cluster.

In this paper, we adopt the self-constructing concept of - Yi Jiang, Ren-Jia Liou, and Shie-Jue [13] and propose an efficient fuzzy similarity-based self-constructing feature clustering approach using an efficient fuzzy membership function. Sometimes the data distribution may be skewed so the usage of the exponential approach to data distribution may give more weak results which is used by Jung-Yi Jiang, Ren-Jia Liou, and Shie-Jue Lee [13]. We are using efficient split Gaussian distribution function as fuzzy membership function. By this algorithm, the derived membership functions match closely with and describe properly the real distribution of the training data. Besides, the user need not specify the number of extracted features in advance, and trial-and-error for determining the appropriate number of extracted features can then be avoided. Experiments on real world data sets show that our method can run faster and obtain better extracted features than other methods.

The remainder of this paper is organized as follows: Section 2 gives a brief background about feature reduction and the disadvantages of Skewness. Section 3 presents the proposed fuzzy similarity-based feature clustering algorithm and the proposed split distributional membership function. .Experimental results are presented in Section 4. Finally, we conclude this work in Section 5.

2. Background and Related Work:

To process documents, the bag-of-words model [8] is usually used. Let d_i be a document and the set $D=\{ d_1,d_2,\ldots.d_n \}$ represent *n* documents. Let the word set $W=\{w_1,w_2,\ldots..w_m\}$ be the feature set of the documents. Each document *di*, $1\leq i\leq n$, can be represented as $di=<d_{i1},d_{i2},\ldots d_{im}\}$, where each d_{ij} denotes the number of occurrence of w_j in document d_i. The feature reduction task is to find a new word $W'=\{w'_1,w'_2,\ldots..w'_m\}$, such

that W and W' work equally well for all the desired properties with *D*. After feature reduction, each document d_i is converted to a new representation $d'_i=<d'_{i1},d'_{i2},\ldots d'_{ik}\}$ and the converted document set is $D'= \{d'_1,d'_2,\ldots.d'_n \}$. If *k* is very much smaller than *m*, computation cost can be drastically reduced.

2.1 Feature Reduction:

In text classification, the dimensionality of the feature vector is usually huge. Even more, there is the problem of *Curse of Dimensionality*, in which the large collection of features takes very much dimension in terms of execution time and storage requirements. This is considered as one of the problems of *Vector Space Model (VSM)* where all the features are represented as a vector of *n* - dimensional data. Here, *n* represents the total number of features of the document. This features set is huge and high dimensional.

There are two popular methods for feature reduction: *Feature Selection and Feature Extraction*. In feature selection methods, a subset of the original feature set is obtained to make the new feature set, which is further used for the text classification tasks with the use of Information Gain [5]. In feature extraction methods, the original feature set is converted into a different reduced feature set by a projecting process. So, the number of features is reduced and overall system performance is improved [6].

Feature extraction approaches are more effective than feature selection techniques but are more computationally expensive. Therefore, development of scalable and efficient feature extraction algorithms is highly demanded to deal with high-dimensional document feature sets. Both feature reduction approaches are applied before document classification tasks are performed.

2.2 Feature Clustering:

Feature clustering is an efficient approach for feature reduction [14], [15], which groups all features into some Clusters, where features in a cluster are similar to each Other. The feature clustering methods proposed in [14],[15],[16],[17] are "hard" clustering methods, where each word of the original features belongs to exactly one word cluster. Therefore each word contributes to the synthesis of only one new feature. Each new feature is obtained by summing up the words belonging to one cluster. Let D be the matrix consisting of all the original documents with m features and D' be the matrix consisting of the converted documents with new k features. The new feature set $W'=\{w'_1,w'_2,\ldots..w'_k\}$ corresponds to a partition $\{W_1,W_2,\ldots..W_k \}$ of the original feature set W, i.e., $Wt \cap Wq=Ø$, where $1 \leq q; t \leq k$ and $t \neq q$. Note that a cluster corresponds to an element in the partition. Then, the t^{th} feature value of the converted document d'_i is calculated

$$d'_{it} = \sum_{wj \in Wt} d_{ij}$$

as follows which is a linear sum of the feature values in W_t. The divisive information-theoretic feature clustering

(DC) algorithm, proposed by Dhillon et al. [17] calculates the distributions of words over classes, $P(C/w_j)$, $1 \le j \le m$, where $C = \{ c_1; c_2; \ldots\ldots; c_p \}$, and uses Kullback-Leibler divergence to measure the dissimilarity between two distributions.
The distribution of a cluster Wt is calculated as follows:

$$P(C|W_t) = \sum_{w_j \in W_t} \frac{P(W_j)}{\sum_{wj \in Wt} P(W_j)} P(C|W_j)$$

The goal of DC is to minimize the following objective Function:

$$\sum_{t=1}^{k} \sum_{w_j \in W_t} P(w_j) KL\,(P(C|w_j), P(C|W_t)) \quad (1)$$

which takes the sum over all the k clusters, where k is Specified by the user in advance.

Later Jung-Yi Jiang, Ren-Jia Liou, and Shie-Jue Lee propose a fuzzy similarity-based self-constructing feature clustering algorithm with fuzzy membership function as follows.

$$\mu_{G_j}(x_i) = \prod_{q=1}^{p} \exp\left[-\left(\frac{x_{iq} - m_{jq}}{\sigma_{jq}}\right)^2\right] \quad (2)$$

With mean and deviations are

$$m_i = \frac{\sum_{j=1}^{q} x_{ji}}{|G|}$$

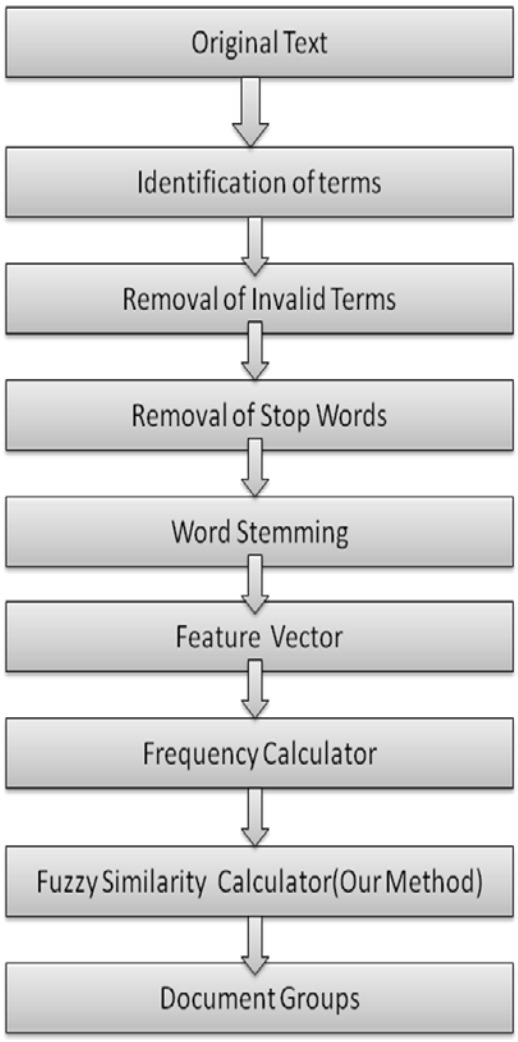
$$\sigma_i = \sqrt{\frac{\sum_{j=1}^{q} (x_{ji} - m_{ji})^2}{|G|}}$$

4. OUR METHOD:

There are some difficulties with the clustering-based feature extraction methods described in the previous section. Firstly, they have to be given the parameter k indicating the desired number of clusters to which all the patterns have to be assigned. Secondly, the computation time depends on the number of iterations, which may be expensively high. Thirdly The existing methods uses the guassian distribution as membership function in which the distribution may be skewed which may give more weak results.
We propose an approach to deal with these difficulties. We develop an incremental word clustering procedure which uses a pre-specified threshold to determine the number of clusters automatically. Each word contains a similarity degree, between 0 and 1, to each cluster. Based on these degrees, a word with a larger degree will contribute a bigger weight than another one with a smaller degree to form a new feature corresponding to the cluster.
In our method we are mainly focusing on the problem of Skewness occurring in the distribution.To reduce the Skewness, We are proposing the Split Gaussians distribution function which reduces the skewness in the distribution.

3.1 Preprocessing Steps:

Initially each sentence is pre-processed and finds out the feature vector.The Preprocessing of the document can be done by removing the invalid terms, removal of stop word and the process of word stemming as shown in figure.Later each feature's frequency is calculated by the frequency calculator which is applied to our Fuzzy similarity method .Finally, the conversion of the *Feature Vector into* the *Reduced Feature Vector.*



3.2 Skewed Distribution:

In mathematics, the term "skew" can refer either to statistical skew (also called "skewness") or to geometrical skew. In statistics, skew is a measure of the extent to which a data distribution is asymmetrical. In the realm of probability and statistics, skewness or skew is a measure of the extent to which a data distribution is distorted from a symmetrical normal distribution. The distortion is in one direction, either toward higher values or lower values. In the former case, the distribution is positively skewed; in the latter, it is negatively skewed.

3.2.1 Drawbacks:

Researchers use statistics to try to determine the true values of a specific phenomenon. For example, researchers might want to know the average ages of people in certain age groups. They would ideally like to know the exact average, but various factors, such as skewed distributions can lead to less accurate results. The actual average of the distribution falls somewhere between the median and mean of the distribution.

Distributions are graphical depictions of statistical probabilities. Statistical distributions look like mountains. The lines on the far left and right side of the graph are called tails.

Distributions are skewed when one tail is longer than the other. When the skew is positive, the longer tail is on the side of the graph with positive numbers, with the opposite true for negative skews. An example of a skewed distribution could involve the unemployment rate. Some people may have never held a job because they stayed in prison most of their lives, skewing the unemployment rate up. Skewed distributions are part of research techniques that still can give accurate data, especially when the distributions are not skewed too far

Extremely skewed distributions can lead to misleading statistics, since the skewed distribution can drive an average up or down. For example, the average income of a particular society may be low, but a handful of people might have high earnings, which skew the average income of the society upward. Badly skewed data can lead to incorrect results, a disadvantage of skewed distributions

3.3 Feature Frequency Calculator:

Suppose, we are given a document set D of n documents $d_1, d_2; \ldots ; d_n$, together with the feature vector W of m words $w_1; w_2; \ldots ; w_m$ and p classes $c_1, c_2; \ldots ; c_p$, as specified in Section 2. We construct one word pattern for each word in W. For word w_i, its word pattern x_i is defined, similarly as in [16], by

$$x_i = < x_{i1}, x_{i2}, \ldots . x_{ip} >$$

$$= < P(c_1|w_1), P(c_2|w_1) \ldots . P(c_p|w_1) >$$

Where

$$P(C_j|W_i) = \frac{\sum_{q=1}^{n} d_{qi} \times \delta_{qj}}{\sum_{q=1}^{n} d_{qi}} \quad (3)$$

for $1 \le j \le p$. Note that d_{qi} indicates the number of occurrences of w_i in document d_q, as described in Section 2. Also, δ_{qj} is defined as

$$\delta_{qj} = \begin{cases} 1, & \text{if document } \mathbf{d}_q \text{ belongs to class } c_j, \\ 0, & \text{otherwise.} \end{cases}$$

Our goal is to group the words in W into clusters, based on these word patterns. A cluster contains a certain number of word patterns, and is characterized by the product of

p special Gaussian functions. In this paper the following definition of split Gaussian l fuzzy membership function is used.We are taking the Split Gaussian function to reduces the skewness in the data distribution.

The split normal distribution arises from merging two opposite halves of two probability density functions (PDFs) of normal distributions in their common mode.

The PDF of the split normal distribution is given by

$$f(x;\mu,\sigma_1,\sigma_2) = A\exp(-\frac{(x-\mu)^2}{2\sigma_1^2}) \quad \text{if } x < \mu$$

$$f(x;\mu,\sigma_1,\sigma_2) = A\exp(-\frac{(x-\mu)^2}{2\sigma_2^2}) \quad \text{otherwise}$$

(4)

Where μ is the mode and σ is the standard deviation.The fuzzy similarity of a word pattern x = <x_1; x_2; . . . ; x_n> to cluster G is defined by the following membership function:

$$\mu_G(x) = \prod_{i=1}^{p} f(x_i,\mu_i,\sigma_1,\sigma_2) \qquad (5)$$

Where $\mu \in \Re$ — mode (location, real) $\sigma_1 > 0$— left-hand-side standard deviation (scale, real) $\sigma_2 > 0$— right-hand-side standard deviation (scale, real)

$$A = 2/\pi(\sigma_1 + \sigma_2)^{-1}.$$

Estimation of parameters:

John[2] proposes to estimate the parameters using maximum likelihood method. He shows that the likelihood function can be expressed in an intensive form, in which the scale parameters σ_1 and σ_2 are a function of the location parameter μ. The likelihood in its intensive form is:

$$L(\mu) = -\left[\sum_{x_i:x_i<\mu}(x_i-\mu)^2\right]^{1/3} - \left[\sum_{x_i:x_i>\mu}(x_i-\mu)^2\right]^{1/3}$$

and has to be maximized numerically with respect to a single parameter μ only.

Given the maximum likelihood estimator $\hat{\mu}$ the other parameters take values:

$$\hat{\sigma}_1^2 = \frac{-L(\mu)}{N}\left[\sum_{x_i:x_i<\mu}(x_i-\mu)^2\right]^{2/3},$$

$$\hat{\sigma}_2^2 = \frac{-L(\mu)}{N}\left[\sum_{x_i:x_i>\mu}(x_i-\mu)^2\right]^{2/3},$$

(6)

Where **N** is the number of observations which is equal to the length of the cluster..

3.2 Self Constructing Clustering:

In the self-constructing feature clustering algorithm, clusters are generated, with none at the beginning, incrementally from the training data set based on fuzzy similarity. One feature pattern is considered in each time. The fuzzy similarity between the input feature and each existing feature cluster is calculated. If the input feature is similar enough to none of the existing clusters, a new cluster for the feature is created and the corresponding membership functions should be initialized. Otherwise, the input feature is combined to the existing cluster to which it is most similar, and the corresponding membership functions of that cluster should be updated

Let k be the number of currently existing clusters. The clusters are G1, G2; . . .;Gk, respectively. Each cluster Gj has mode μ_j = <μj1; μj2; . . .; μjp> and deviation $\sigma_{j=}$ < $\sigma_{j1,}$ $\sigma_{1j2.....}$ σ_{jp} >. Let Sj be the size of cluster Gj. Initially, we have k = 0. So, no clusters exist at the beginning. For each word pattern xi = <x_{i1}; x_{i2}; . . . ; x_{ip}>; 1≤i ≤ m, we calculate, according to (4,5), the similarity of xi to each existing clusters, i.e.,

$$f(x;\mu,\sigma_1,\sigma_2) = A\exp(-\frac{(x-\mu)^2}{2\sigma_1^2}) \quad \text{if } x < \mu$$

$$f(x;\mu,\sigma_1,\sigma_2) = A\exp(-\frac{(x-\mu)^2}{2\sigma_2^2}) \quad \text{otherwise}$$

$$\mu_G(x) = \prod_{i=1}^{n} f(x_i, \mu_i, \sigma 1, \sigma 2)$$

For 1 ≤ j ≤k. We say that xi passes the similarity test on cluster Gj if

$$\mu_{G_j}(\mathbf{x}_i) \geq \rho, \qquad (7)$$

Where ρ, 0≤ρ≤1, is a predefined threshold.

If the user intends to have larger clusters, then he/she can give a smaller threshold. Otherwise, a bigger threshold can be given. As the threshold increases, the number of clusters also increases. Note that, as usual, the power in (4) its value has an effect on the number of clusters obtained.

A larger value will make the boundaries of the Gaussian function sharper, and more clusters will be obtained for a

given threshold. On the contrary, a smaller value will make the boundaries of the Gaussian function Smoother and fewer clusters will be obtained instead.

Two cases may occur. First, there are no existing fuzzy clusters on which xi has passed the similarity test. For this
case, we assume that xi is not similar enough to any existing cluster and a new cluster Gh, h = k + 1, is created with

$$\mu h = xi; \sigma h = \sigma 0; \quad (8)$$

Where $\sigma 0 = <\sigma 0; \ldots; \sigma 0>$ is a user-defined constant vector. Note that the new cluster Gh contains only one member, the word pattern xi, at this point. Estimating the deviation of a cluster by (4, 5) is impossible, or inaccurate, if the cluster contains few members. In particular, the deviation of a new cluster is 0, since it contains only one member. We cannot use zero deviation in the calculation of fuzzy similarities.Therefore; we initialize the deviation of a newly created cluster by σ0, as indicated in (17). Of course, the number of clusters is increased by 1 and the size of cluster Gh, Sh, should be initialized, i.e.,

$$k = k + 1; Sh = 1: \quad (9)$$

Second, if there are existing clusters on which xi has passed the similarity test, let cluster Gt be the cluster with the largest membership degree, i.e.,

$$t = \arg \max_{1 \le j \le k} \{\mu G_j(x_i)\} \quad (10)$$

In this case, we regard xi to be most similar to cluster Gt, and the mode and standard deviation will be updated of that cluster.

3.2 Algorithm

Initialization:
# of original word patterns: m
# of classes: p
Threshold: ρ
Initial deviation: σ0
Initial # of clusters: k = 0
Input:
xi = <xi1; xi2; . . . ; xip>, $1 \le i \le m$
Output:
Clusters G1, G2; . . .;Gk
Procedure Self-Constructing-Clustering-Algorithm
For each word pattern xi, $1 \le i \le m$
Temp_ W= { Gj | $\mu_G(x_i) \ge \rho$ $1 \le j \le k$};
if (temp_ W==ϕ)
A new cluster Gh, h = k +1, is created by
(8)-(9);
else let Gt € temp_W be the cluster to which xi is closest by (10);
Incorporate xi into Gt by updating its σ values
end if;
end for;
return with the created k clusters;
end procedure

3.3 Text Classification:

Given a set D of training documents, text classification can be done as follows: We specify the similarity threshold _ for (16), and apply our clustering algorithm. Assume that k clusters are obtained for the words in the feature vector W. Then we find the weighting matrix T and convert D to D_0 by (25). Using D_0 as training data, a classifier based on Support vector machines (SVM) is built. Note that any classifying technique other than SVM can be applied.Joachims [36] showed that SVM is better than other methods for text categorization.

A **support vector machine** (**SVM**) is a concept in statistics and computer science for a set of related supervised learning methods that analyze data and recognize patterns, used for classification and regression analysis. The standard SVM takes a set of input data and predicts, for each given input, which of two possible classes forms the input, making the SVM a non-probabilistic binary linear classifier. Given a set of training examples, each marked as belonging to one of two categories, an SVM training algorithm builds a model that assigns new examples into one category or the other. An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall on.

SVM is a kernel method, which finds the maximum margin hyper plane in feature space separating the images of the training patterns into two groups [37], [38], [39].

To make the method more flexible and robust, some patterns need not be correctly classified by the hyper plane, but the misclassified patterns should be penalized. Therefore, slack variables $_i$ are introduced to account for misclassifications. The objective function and constraints of the classification problem can be formulated as:

$$\min_{w,b} \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{l}\xi_i$$

$$s.t. y_i\left(\mathbf{w}^T\phi(\mathbf{x}_i) + b\right) \ge 1 - \xi_i, \xi_i \ge 0, i = 1, 2, \ldots, l,$$

where l is the number of training patterns, C is a parameter, which gives a tradeoff between maximum margin and classification error, and yi, being +1 or -1, is

the target label of pattern xi. Note that o:X->F is a mapping from the input space to the feature space F, where patterns are more easily separated, and $\mathbf{w}^T\phi(\mathbf{x_i})$ + b=0 is the hyper plane to be derived with w, and b being weight vector and offset, respectively.

An SVM described above can only separate apart two classes, yi = +1 and yi =-1. We follow the idea in [36] to construct an SVM-based classifier.

For p classes, we create p SVMs, one SVM for each class. For the SVM of class cv, 1≤ v≤ p, the training patterns of class cv are treated as having yi =+1, and the training patterns of the other classes are treated as having yi =-1. The classifier is then the aggregation of these SVMs. Now we are ready for classifying unknown documents. Suppose, d is an unknown document. We first convert d to d' by

$$\mathrm{d}' = \mathrm{dT}.$$

Then we feed d0 to the classifier. We get p values, one from each SVM. Then d belongs to those classes with 1, appearing at the outputs of their corresponding SVMs. For example, consider a case of three classes' c1, c2, and c3. If the three SVMs output 1, -1, and 1, respectively, then the predicted classes will be c1 and c3 for this document. If the three SVMs output -1, 1, and 1, respectively, the predicted classes will be c2 and c3

4. Experimental Result:

To compare classification effectiveness of each method, we adopt the performance measures in terms of microaveraged precision (MicroP), micro averaged recall (MicroR), microaveraged F1 (MicroF1), and micro averaged accuracy (MicroAcc)] defined as follows:

$$MicroP = \frac{\Sigma_{i=1}^{p} TP_i}{\Sigma_{i=1}^{p}(TP_i + FP_i)},$$

$$MicroR = \frac{\Sigma_{i=1}^{p} TP_i}{\Sigma_{i=1}^{p}(TP_i + FN_i)},$$

$$MicroF1 = \frac{2MicroP \times MicroR}{MicroP + MicroR},$$

$$MicroAcc = \frac{\Sigma_{i=1}^{p}(TP_i + TN_i)}{\Sigma_{i=1}^{p}(TP_i + TN_i + FP_i + FN_i)},$$

where p is the number of classes. TP_i (true positives wrt c_i) is the number of c_i test documents that are correctly classified to c_i. TN_i (true negatives wrt c_i) is the number of non-c_i test documents that are classified to non-c_i. FP_i (false positives wrt c_i) is the number of non-c_i test documents that are incorrectly classified to c_i. FN_i (false negatives wrt c_i) is the number of c_i test documents that are classified to non-c_i. The experimental results show that our method works well & shows better performance than the previous methods.

5. Conclusion:

In this paper we have presented an efficient Fuzzy clustering algorithm for text classification which uses a different fuzzy membership function that solves the problem of skewness.The skewness of data in the distribution gives more weak results which effects the resulting clusters which may affects the performance of the text classification. In this paper we are proposing the split Gaussian distribution function as fuzzy membership function that works well for all types of data.

6. Acknowledgments:

Our thanks to the experts who have contributed towards development of this paper.

REFERENCES:

[1] L. D. Baker and A. McCallum. "Distributional clustering of words for text classification," 21st Annual International ACM SIGIR, pages 96–103, 1998.

[2] R. Bekkerman, R. El-Yaniv, N. Tishby, and Y. Winter. "Distributional word clusters vs. words for text categorization," Journal of Machine Learning Research, 3:1183–1208, March 2003.

[3] M. C. Dalmau and O. W. M. Fl´o'rez. "Experimental results of the signal processing approach to distributional clustering of terms on reuters-21578 collection," 29th European Conference on IR Research, pages 678–681, 2007.

[4] I. S. Dhillon, S. Mallela, and R. Kumar. "A divisive infromation-theoretic feature clustering algorithm for text classification," Journal of Machine Learning Research, 3:1265–1287, March 2003.

[5] D. Ienco and R. Meo. "Exploration and reduction of the feature space by hierarchical clustering," 2008 SIAM Conference on Data Mining, pages 577–587, 2008.

[6] S. J. Lee and C. S. Ouyang. "A neuro-fuzzy system modeling with self-constructing rule generation and hybrid svd-based learning," IEEE Transactions on Fuzzy Systems, 11(3):341– 353, June 2003.

[7] F. Pereira, N. Tishby, and L. Lee. "Distributional clustering of English words," 31st Annual Meeting of ACL, pages 183– 190, 1993. [8] G. Salton and M. J. McGill. Introduction to Modern Retrieva,. McGraw-Hill Book Company, 1983.

[8] G. Salton and M. J. McGill. Introduction to Modern Retrieva,. McGraw-Hill Book Company,1983.

[9] F. Sebastiani. "Machine learning in automated text categorization," ACM Computing Surveys, 34(1):1–47, March 2002.

[10] N. Slonim and N. Tishby. "The power of word clusters for text classification," 23rd European Colloquium on Information Retrieval Research (ECIR), 2001.

[11] L. X.Wang. A Couse in Fuzzy Systems and Control. Prentic-Hall International, Inc., 1997.

[12] Y. Yang and J. O. Pedersen. "A comparative study on feature selection in text categorization,

[13] Jung-Yi Jiang, Ren-Jia Liou, and Shie-Jue Lee, Member, IEEE A Fuzzy Self-Constructing Feature Clustering Algorithm for Text Classification
[14] R. Bekkerman, R. El-Yaniv, N. Tishby, and Y. Winter, "Distributional Word Clusters versus Words for Text Categorization," J. Machine Learning Research, vol. 3, pp. 1183-1208, 2003
[15] N. Slonim and N. Tishby, "The Power of Word Clusters for Text Classification," Proc. 23rd European
[16] L.D. Baker and A. McCallum, "Distributional Clustering of Words for Text Classification," Proc. ACM SIGIR, pp. 96-103, 1998.
[17] I.S. Dhillon, S. Mallela, and R. Kumar, "A Divisive Infomation- Theoretic Feature Clustering Algorithm for Text Classification," J. Machine Learning Research, vol. 3, pp. 1265-1287, 2003
[18] C.C. Chang and C.J. Lin, "Libsvm: A Library for Support Vector Machines," http://www.csie.ntu.edu.tw/~cjlin/libsvm. 2001.

Krishna Mohan received M.Tech Computer Science&Engg from Andhra University in1997.He is currently working as Associate professor in CSE department, JNTU Kakinada. He has 12 years of Industrial experience in the IT Industry with various MNC's like TCS,Vangaurd,Hard Ford Insurance , E-Trade ,Fugitse-ICIT.His main research interests includes machine learning ,data mining & Information retrieval, web technologies,J2EE,Web services.

V.V.Narasimha Rao is currently pursuing his M.Tech Computer Science &Engineering from J.N.T.U Kakinada. His main research interests are Information retrieval, machine learning, and data mining and mobile computing

MHM Krishna Prasad is currently working as Associate professor in JNTU Vijayanagaram.He obtained doctorate in 2011.His main research interests include machine learning, data mining & Web programming