Fuzzy Feature Clustering for Text Classification Using Sequence Classifier

Ambily Balaram

M-Tech Computer Science &Engineering
KMCT College of Engineering, Calicut, India

***Abstract*— Due to the rapid growth of Internet Technologies and the prosper of WWW, the volume of textual data is increasing more and more, thereby leading to the significance of text classification. Feature Clustering is a powerful method to reduce the dimensionality of feature vector for text classification. Text Classification is one of the important research issues in the field of text mining, where the documents are classified with supervised knowledge. This paper proposes a sequence classifier in a two stage approach combining Support Vector Machines (SVMs) and Conditional Random Fields (CRFs).It is (i) highly accurate, (ii) Scalable and (iii) Easy to use in Data mining approach. . The proposed model works efficiently and effectively with great performance and high - accuracy results**

***Keywords*— Conditional Random Fields, Support Vector Machine.**

I. INTRODUCTION

With the rapid growth of online information, text classification has become one of the key techniques for handling and organizing text data. The problem in the field of text classification is that of huge dimensionality of the data when documents are represented by a vector that indicates how often each word occurs in the document. For example, two real-world data sets, 20 Newsgroups and Reuters 21591 top-10, both have more than 15 thousand features. In this context, feature selection [1] and feature extraction [2] have been used for feature reduction.

Feature Selection involves choosing a subset of the feature for document representation. The best subset contains least number of dimensions that most contribute to accuracy and remaining unimportant features are discarded. The two main approaches to feature selection are filtering and Wrapping. Wrapping is more time consuming and sometimes infeasible to use.

Feature Extraction creates new feature from the functions of original feature transforming input data into a set of features. In general, feature extraction approaches are more effective than feature selection techniques, but are more computationally expensive [1].Feature Clustering is one of effective techniques for feature reduction in text classification.

Feature clustering is nothing but grouping of the words with a high degree of pair wise semantic relatedness into clusters and each word cluster contains the grouped features 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 proposed by Baker and McCallum[3] which was derived from the distributional clustering idea of Pereira et al. There are many feature clustering methods where each new feature is generated by combining a subset of the original words. But all those methods 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 [4], which is an incremental feature clustering approach to reduce the number of features for the text classification task.

In this paper, we propose a fuzzy feature clustering algorithm for text classification using a two stage approach using SVM/CRF[7].We can show that SVM/CRF can be used together to achieve high accuracy and high speed on a sequence classification task. Essentially, we first use SVMs to learn to predict the labels of individual input sequence data items. Then, we use a CRF to predict the sequence of all output labels, where the input to the CRF is the outputs of the SVMs applied to the inputs.

II. BACKGROUND AND RELATED WORK

To process documents, the bag-of-words model is usually used. Let di be a document and the set D= {d1,d2….dn } represent n documents. Let the word set W= {w1,w2…wm} be the feature set of the documents. Each document di, 1≤i≤n, can be represented as di=<di1,di2…dim}, where each dij denotes the number of occurrence of wj in document di. The feature reduction task is to find a new word W‘= {w‘1,w‘2…w‘m}, such that W and W‘ work equally well for all the desired properties with D. After feature reduction, each document di is converted to a new representation d‘i=<d‘i1,d‘i2…d‘ik} and the converted document set is D‘= {d‘1,d‘2….d‘n }. If k is very much smaller than m, computation cost can be drastically reduced.

A. *Feature Reduction*

The major characteristic or difficulty of text classification problem is the high dimensionality of the feature space. Therefore, it is highly desirable to reduce the native space without sacrificing the categorization accuracy.

In general, there are two ways of doing feature reduction, feature selection, and feature extraction. Feature selection methods include the removal of non-informative terms according to corpus statistics, and the construction of new features which combine lower level features into higher level orthogonal dimensions.

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.

B. *Feature Clustering*

Feature clustering is an efficient approach for feature reduction [3], which groups all features into some clusters, where features in a cluster are similar to each Other. In hard feature clustering methods, 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…w‘k} corresponds to a partition {W1, W2,…..Wk } of the original feature set W, i.e., Wt ∩ Wq=Ø, where 1 ≤ q; t ≤k and t ≠q. Note that a cluster corresponds to an element in the partition. Then, the tth feature value of the converted document d‘i is calculated

$$dit'=dijwj\in Wt$$

as follows which is a linear sum of the feature values in Wt. The divisive information-theoretic feature clustering (DC) algorithm, proposed by Dillon et calculates the distributions of words over classes, P(C/wj), 1 ≤ j ≤ m, where C ={ c1; c2; . . . . . ; cp }, and uses Kullback-Leibler divergence to measure the dissimilarity between two distributions. The distribution of a cluster Wt is calculated as follows:

$$P(C|Wt) = \sum_{wj\in Wt} \frac{P(wj)}{\sum wj \in WtP(wj)} P(C|wj)$$

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

$$\sum_{t=1}^{k} \sum_{wj\in Wt} P(wj)KL(P(C|wj), P(C|Wt))$$

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

III. BASIC IDEA OF CONDITIONAL RANDOM FIELD(CRF)

Conditional Random Fields are a framework for building probabilistic models to segment and label sequence data. A key advantage of CRFs is their great flexibility to include a wide variety of arbitrary, non-independent features of the input.

Given a dataset of input and output sequences (X, Y), the training objective for a CRF model is to choose parameters W (also called weights) that maximize the conditional log likelihood log P(Y |X;W), which is

$$\sum_{(\bar{x}_i,\bar{y}_i)\in(X,Y)} \log \frac{\exp \sum_{z=1}^{d} w_z F_z(\bar{x}_i,\bar{y}_i)}{\sum_{\bar{y}'} \exp \sum_{z=1}^{d} w_z F_z(\bar{x}_i,\bar{y}')}$$

Here there are d different fixed feature-functions denoted Fz for z = 1, . . . , d. There is one trainable parameter wz for each Fz. Each feature-function Fz is actually a sum over output sequence positions of a lower-level feature-function fz. That is, each high-level feature-function Fz has the form

$$Fz\ (\bar{x}i, \bar{y}i) = \sum_{j} fz\ (xij, yij-1, yij)$$

where j ranges over the elements of yi and yi0 is a special token to represent the beginning of a sequence. Although the lower-level functions fz contain real-values, all the z functions we use are binary, i.e. they have value 0 or 1. Each fz function can depend on any or all of the input sequence, and/or on up to two adjacent labels in the output sequence ¯yi. The reason why only at most two adjacent output labels can be used is that making predictions efficiently with a trained CRF model depends on the Viterbi algorithm to compute

$$argmax \sum_{z=1}^{d} wz\ Fz(\bar{x}i, \bar{y}i)$$

and this algorithm cannot handle lower-level feature-functions that involve more than two adjacent elements of ¯yi. The alternative CRFs that we consider use various combinations of the following six types of feature-function, which are all special cases of the general form above.

Feature-functions of the first type have the form

$$Fz^{(1)}(\bar{x}i, \bar{y}i) = \sum_{j} Fz^{(1)}(Xij, yij)$$

There are c · v · p functions of this type, because there are c possible values for yij , v attributes of xij , and p possible values for each attribute.

Feature-functions of the second type have the form

$$Fz^{(2)}\ (\bar{x}i, \bar{y}i) = \sum_{j} Fz^{(2)}(Xij, yij-1, yij)$$

The number of functions of this type is c2vp.

We represent these feature types as follows:

$$Fz^{(3)}\ (\bar{x}i, \bar{y}i) = \sum_{j}\ Fz^{(3)}(yij) \quad \text{and}$$

$$Fz^{(4)}\ (\bar{x}i, \bar{y}i) = \sum_{j} Fz^{(4)}(yij-1, yij)$$

There are c features of the former type, and c2 of the latter type.

IV. OUR METHOD

There are some issues pertinent to most of the existing feature clustering methods. Firstly, they have to be given the value of k indicating the required number of clusters in advance to which all the patterns have to be assigned. Secondly, the computation time depends on the number of iterations, which may be expensively high. Our feature clustering algorithm deals with these issues.

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, 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 a two stage sequence classifier for text classification. In our approach, first SVMs are trained to predict the label of each input sequence element; this is a standard multiclass supervised learning task. Second, one CRF is trained to predict the output sequence of labels using as its input the outputs from the previously trained SVMs. The intuition is that both learning approaches are somewhat orthogonal in their advantages, so a combination of them can yield better results.

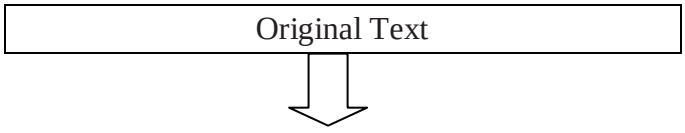
A. *Pre-processing Steps*

The basic phases in text categorization include pre-processing features, extracting relevant features against the features in a database, and finally categorizing a set of documents into predefined categories. Among these, pre-processing is the most important subtask of text classification.

The importance of pre-processing is emphasized by the fact that the quantity of training data grows exponentially with the dimension of the input space. It has already been proven that the time spent on pre-processing can take from 50% up to 80% of the entire classification process, which clearly proves the importance of pre-processing in text classification process.

The pre-processing phase of the study converts the original textual data in a data-mining-ready structure, where the most significant text-features that serve to differentiate between text-categories are identified. It is the process of incorporating a new document into an information retrieval system. An effective pre-processor represents the document efficiently in terms of both space (for storing the document) and time (for processing retrieval requests) requirements and maintain good retrieval performance (precision and recall). This phase is the most critical and complex process that leads to the representation of each document by a select set of index terms. The main objective of pre processing is to obtain the key features or key terms from online news text documents and to enhance the relevancy between word and document and the relevancy between word and category.

The pre processing procedure is shown in fig 1. 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. The goal behind pre processing is to represent each document as a feature vector, that is, to separate the text into individual words.



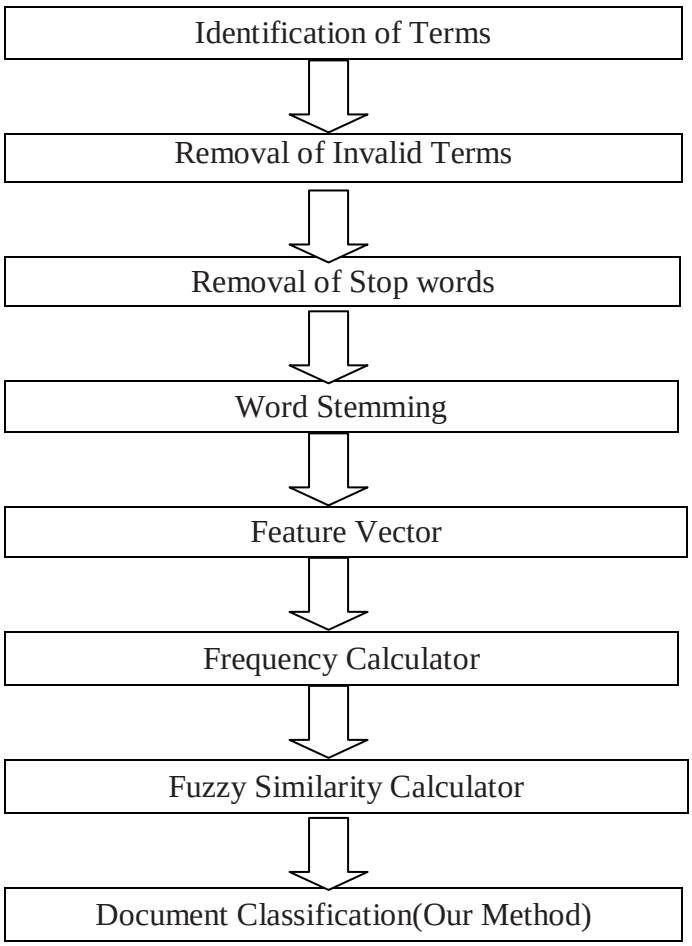


Fig 1 Pre-processing Procedure

B. *Calculating Frequency*

Suppose, we are given a document set D of n documents d1, 2; . . . ; dn, together with the feature vector W of m words w1; w2; . . . ; wm and p classes c1, c2; . . . ; cp, as specified in Section 2. We construct one word pattern for each word in W. For word wi, its word pattern xi is defined, by

$$xi = \langle xi1, xi2, \dots \dots xip \rangle$$

$$= \langle P(c1|w1), P(c2|w2) \dots \dots P(cp|w1) \rangle$$

$$where \quad P(Cj|Wi) = \frac{\sum_{q=1}^{n} dqi \times \delta qj}{\sum_{q=1}^{n} dqi}$$

for 1≤j≤p. dqi indicates the number of occurrences of wi in document dq and δqj is defined as

$$\delta_{qj}=\begin{cases}1, if\ document\ d_q\ belongs\ to\ class\ c_j\\ 0, \quad otherwise\end{cases}$$

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.

C. *Fuzzy Feature Clustering Algorithm*

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. For each word pattern, the similarity of this word pattern to each existing cluster is calculated using Gaussian function , to decide whether it is combined into an existing cluster or a new cluster is created. 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.

D. *Feature Extraction*

The Feature extraction can be done by using the following matrix equation.

D‘=DT

Where D is input document set such as [d1,d2,..dn] T .D‘ is the reduced document set [d1‘,d2‘..dn‘]T .

$$T=\begin{bmatrix}t11 & \dots & t1k\\ \vdots & \ddots & \vdots\\ tm1 & \dots & tmk\end{bmatrix}$$

$$with\ di=[di1,di2\ \dots..dim]$$

$$d'i=[d'i1,d'i2\ \dots.d'ik]for\ 1\le i\le n$$

Here we have to find the T such that k is far smaller than m to achieve the feature reduction. The elements of T are derived based on the obtained clusters and feature extraction will be done. There are three weighting approaches hard, soft and mixed. In the hard weighting approach, only each word belongs to a single cluster only. In soft weighting approach each word may belong to different clusters. The mixed weighting approach is a combination of both soft weighting and hard weighting approaches.

The whole clustering algorithm can be summarized as below.

Initialization:

$Dimension\ dm=2$

$Threshold\ \rho$

$Initial\ Deviation\ \sigma 0$

Input:

1. $Word\ Pattern\ \bar{w}=\langle x1,x2,\dots..xm\rangle$
2. $p\ classes\ \{c1,c2\ \dots\ \dots.cp\}$

Output:

$Cluster\ of\ words\ \{G1,G2\ \dots..Gk\}$

Procedure:

Step1.For each word pattern ,load word pattern xi, 1≤i≤m, calculate

$$tempA=\mu Gj(xi)=\prod_{q=1}^{dm}exp\left(-\frac{(xiq-mjq)}{\sigma jq}^{2}\right)$$

Step2. If (tempA<ρ) then

A new cluster Gh, h=k+1 is created.

mh=xi, σh=σ0

else

Step 3.Let Gt be the cluster by which xt passes the similarity test by

$$t=\arg\max_{1<\sigma<k}(\mu Ga(xi))$$

add xt to the cluster Gt and update mean and deviation using the equation

$$mtj=\frac{St\times mtj+xij}{St+1}$$

$$\sigma tj=\sqrt{A-B}+\sigma 0\ Where$$

$$A=\frac{(st-1)(\sigma tj-\sigma 0)^{2}+St\times mtj^{2}+xtj^{2}}{St}$$

$$B=\frac{(St+1)}{St}\left(\frac{St\times mtj_{xij}}{St+1}^{2}\right)$$

$$St=St+1$$

Step 4.Return k created Clusters.

E. *Two Stage SVM/CRF Text Classification*

Given a set D of training documents, text classification can be done as follows: We specify the similarity threshold , 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 D0. Using D0 as training data, a classifier based on two stage SVM/CRF is built.

In our approach, first SVMs are trained to predict the label of each input sequence element; this is a standard multiclass supervised learning task. Second, one CRF is trained to predict the output sequence of labels using as its input the outputs from the previously trained SVMs. During SVM training, the goal is to learn each class based on each sequence element (i.e. data item or data point) and its label in the training set, by maximizing the separation between data points with labels in the same class and other data points. Many studies have shown that SVMs tend to obtain superior results, compared to other classifiers, for predicting individual labels. This advantage of SVMs stems from their ability to use high-dimensional feature spaces.

Given a data point in the test set, the output of the trained SVMs is a vector of scores. In the second stage of our approach, this vector is used as the input attributes for a CRF classifier Traditionally, a feature-function for a CRF is based on one or more data points, and one label or two adjacent labels. Our proposed new type of feature-function is based on a prediction vector of scores for a data point, instead of directly on the attributes of the data point. Essentially, the two-stage approach uses SVMs as a feature induction method, in order to allow a CRF to learn a better overall classifier.

For multiclass classification SVM can be used in either one-against-all or one-against 0ne fashion. With the one-against all technique, each class is trained separately against the union of all other classes. Applying the trained SVMs on a test data point (xij,yij) yields a vector of prediction scores (g1,g2…….gc)ij. where c is the number of classes. With one-against one technique, each class is trained separately against each other class. Applying the trained SVMs to the test data point yields a vector of prediction scores (g1,g2…….gb)ij where b=c(c-1)/2. There are two additional advantages of using this approach as part of the two-stage SVM/CRF method:it yields faster SVM training, and it increases the bandwidth of information passed to the CRF. Although one-against-one training is conducted (c − 1)/2 times, each time only the data points in two classes are involved. SVM training time is typically super linear in the number of training examples, so learning more classifiers each with a smaller training set is a net win. This improvement in running time is proportional to the number of alternative labels is considerable. The increase in communication bandwidth between the SVMs and the CRF can potentially improve the accuracy achievable by the CRF. However, the larger number of inputs for the CRF tends to increase its training time.

Let X be a set of input sequences and let Y be the corresponding set of sequences of labels. The data (X, Y) consist of samples ($\bar{}$xi, $\bar{}$yi) for i = 1, ..., n. Each sample ($\bar{}$xi, $\bar{}$yi) consists of L(i) data points and their labels. A label yij can belong to one of c different classes, and each input data point xij can have p dimensions with each dimension having one of v values.

Our contribution is to introduce features for the two-stage approach that depend on the data point xij only indirectly, through prediction scores gz(xij) assigned by SVM classifiers.

We formalize this idea as follows:

$$F_z^{(5)}(\bar{x}_i, \bar{y}_i) = \sum_j f_z^{(5)}(g_z(\mathbf{x}_{ij}), y_{ij})$$

and

$$F_z^{(6)}(\bar{x}_i, \bar{y}_i) = \sum_j f_z^{(6)}(g_z(\mathbf{x}_{ij}), y_{ij-1}, y_{ij})$$

where gz(xij) is one element of the score vector produced by the multiclass SVM classifier applied to xij . Real-valued SVM scores are discretized, in order to allow the fz(5) and fz(6) feature-functions to be binary. Each different integer value, for each of the binary SVM classifiers, then gives rise to a

different binary feature-function. Given a real-valued score gz(xij), the integer value that is used as input to the feature-function is

$$g'z(Xij) = \lceil 10.gz(xij) \rceil$$

We maximize a regularized version of the conditional log likelihood,as is customary with CRF as

$$J(X,Y) = \log P(Y|X;W) + \log P(W)$$

The objective function is maximized by gradient descent.

The gradient $(\partial/\partial w_z^{(l)})J(X,Y)$ is

$$\sum_{(\bar{x},\bar{y})\in(X,Y)} F_z^{(l)}(\bar{x},\bar{y}) - \sum_{\bar{y}'} p(\bar{y}'|\bar{x};w_z^{(l)})F_z^{(l)}(\bar{x},\bar{y}') - \frac{2w_z^{(l)}}{\sigma}$$

The gradient, for each weight and for each training example (¯x, ¯y), is essentially the difference between the feature-function value for (¯x, ¯y) and the average value of the feature-function averaging over each ¯y′ with probability given by the current model p(¯y′|¯x;w).

V. PERFORMANCE CRITERIA

To compare classification effectiveness of each method, Jung-Yi Jiang, Ren-Jia Liou, and Shie-Jue Lee adopted the performance measure of micro averaged precision(MicroP), micro averaged recall(MicroR),micro averaged F1 (MicroF1)and microaveraged accuracy(MicroAcc).The experimental results shows that classifier provides an accuracy of about 72.48%. Our hypothesis is that the two-stage combined SVM/CRF method performs more mathematically and computationally complex methods. In previous papers, accuracy is measured as the average error per character. Also as the average over words of the average error per character in each word. Both definitions of accuracy yield very similar results.

$$AccPerChar = I/N \sum_{(i,j)} I(\widehat{yij} = yij)$$

where N is the total number of characters in the test set, yij is the true value of the jth character of the ith word in the test set, and ˆyij is the predicted value of this character .

$$AccPerWord = 1/M \sum_{i=1}^{M} \left[1/L(i) \sum_{j=1}^{L(i)} I(\widehat{yij} = yij) \right]$$

where M is the total number of words and L(i) is the total number of characters in the ith word.The two-stage approach proposed here is much faster and is expected to provide an accuracy of about 89.24%.The performance of two stage SVM/CRF method is good and its accuracy is comparable when using features based on the vector of scores and on adjacent label.

VI. CONCLUSIONS

In this paper, we propose a fuzzy feature clustering algorithm for text classification using a two stage approach using SVM/CRF. It achieves high accuracy because of the maximum-margin nature of SVMs, and because CRFs can model correlations between neighboring output labels. The two stage is scalable because the input for training each SVM is only a small subset of the entire training data and CRF uses only a limited number of features, namely the outputs of the SVMs trained in the first stage.

ACKNOWLEDGMENT

The author would like to thank everyone.

REFERENCES

[1] [1] Y. Yang and J.O. Pedersen, “A Comparative Study on Feature Selection in Text Categorization,” Proc. 14th Int’l Conf. Machine Learning, pp. 412-420, 1997.

[2] [2] D.D. Lewis, “Feature Selection and Feature Extraction for Text Categorization,” Proc. Workshop Speech and Natural Language, pp. 212-217, 1992.

[3] [3] L.D. Baker and A.McCallum, “Distributional Clustering of Words for Text Classification,” Proc. ACM SIGIR, pp. 96-103, 1998.

[4] [4] Jung-Yi Jiang, Ren-Jia Liou, and Shie-Jue Lee, Member, IEEE A Fuzzy Self-Constructing Feature Clustering Algorithm for Text Classification

[5] [5] T. Joachims, “Text Categorization with Support Vector Machine: Learning with Many Relevant Features,” Technical Report LS-8- 23, Univ. of Dortmund, 1998.

[6] [6] Y. Yang and X. Liu, “A Re-Examination of Text Categorization Methods,” Proc. ACM SIGIR, pp. 42-49, 1999.

[7] [7] John Lafferty, Andrew McCallum, Fernando Pereira “Conditional Random Fields: Probabilistic Models For Segmenting And Labeling Sequence Data”