Volume 2, Issue 8, August 2012 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Clustering Algorithm for Text Classification Using Fuzzy Logic

[1]Sainani Arpitha, [2]Dr.P.Raja Prakash Rao
[1] M.TECH STUDENT OF TRR ENGINEERING COLLEGE, HYDERABAD, INDIA
[2] PROFESSOR OF TRR ENGINEERING COLLEGE, HYDERABAD, INDIA

Abstract: Feature clustering is a powerful method to reduce the dimensionality of feature vectors for text classification. We propose a fuzzy similarity-based self-constructing algorithm for feature clustering. The words in the feature vector of a document set are grouped into clusters, based on similarity test. 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. 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. 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. Experimental results show that our method can run faster and obtain better extracted features than other methods.

Indexing terms: fuzzy, training data, cluster, membership function

I. INTRODUCTION

In text classification, the dimensionality of the feature vector is usually huge. For example, 20 Newsgroups [1] and Reuters21578 top-10 [2], which are two real-world data sets, both have more than 15,000 features. Such high dimensionality can be a severe obstacle for classification algorithms [3], [4]. To alleviate this difficulty, feature reduction approaches are applied before document classification tasks are performed. Two major approaches, feature selection [6, [7], [8], [9], [10] and feature extraction [11], [12], [13], have been proposed for feature reduction. In general, feature extraction approaches are more effective than feature selection techniques, but are more computationally expensive [11], [12], [14]. Therefore, developing scalable and efficient feature extraction algorithms is highly demanded for dealing with high-dimensional document data sets. Classical feature extraction methods aim to convert the representation of the original high-dimensional data set into a lower-dimensional data set by a projecting process through algebraic transformations. For example, Principal Component Analysis [15], Linear Discriminant Analysis [16], Maximum Margin Criterion [12], and Orthogonal Centroid algorithm [17] perform the projection by linear transformations, while Locally Linear Embedding [18], ISOMAP [19], and Laplacian Eigenmaps [20] do feature extraction by nonlinear transformations. In practice, linear algorithms are in wider use due to their efficiency.

Several scalable online linear feature extraction algorithms [14], [21], [22], [23] have been proposed to improve the computational complexity. However, the complexity of these approaches is still high. Feature clustering [24], [25], [26], [27], [28], [29] is one of effective techniques for feature reduction in text classification. The idea of feature clustering is to group the original features into clusters with a high degree of pair wise semantic relatedness. Each cluster is treated as a single new feature, and, thus, feature dimensionality can be drastically reduced.

The first feature extraction method based on feature clustering was proposed by Baker and McCallum [24], which was derived from the "distributional clustering" idea of Pereira et al [30]. Al-Mubaid and Umair [31] used distributional clustering to generate an efficient representation of documents and applied a learning logic approach for training text classifiers. The Agglomerative Information Bottleneck approach was proposed by Tishby et al [25], [29]. The divisive information-theoretic feature clustering algorithm was proposed by Dhillon et al [27], which is an information-theoretic feature clustering approach, and is more effective than other feature clustering methods. In these feature clustering methods, each new feature is generated by combining a subset of the original words. However, difficulties are associated with these methods. A word is exactly assigned to a subset, i.e., hard-clustering, based on the similarity magnitudes between the word and the existing subsets, even if the differences among these magnitudes are small. Also, the mean and the variance of a cluster are not considered when similarity with respect to the cluster is computed. Furthermore, these methods require the number of new features be specified in advance by the user.

II. PROPOSED SYSTEM:

A fuzzy similarity-based self-constructing feature clustering algorithm, which is an incremental feature clustering approach to reduce the number of features for the text classification task is proposed. 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. 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. Three ways of weighting, hard, soft, and mixed, are introduced. By this algorithm, the derived membership functions match closely with and describe properly the real distribution of the training data. 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. Real world experiments on data sets show that our method can run faster and obtain better extracted features than other methods.

Feature Reduction

In general, there are two ways of doing feature reduction, feature selection, and feature extraction. By feature selection approaches, a new feature set $W'=\{w'_1,w_2',\ldots,w_k'\}$ is obtained, which is a subset of the original feature set W. Then W0 is used as inputs for classification tasks.

Information Gain (IG) is frequently employed in the feature selection approach [10]. It measures the reduced uncertainty by an information-theoretic measure and gives each word a weight.

Feature Clustering

Feature clustering is an efficient approach for feature reduction [25], [29], which groups all features into some clusters, where features in a cluster are similar to each other. The feature clustering methods proposed in [24], [25], [27], [29] 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.,

$W_t \cap W_q = 0$, where $1 \leq q, t \leq k$ 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 as follows:

$d'_{it} = \sum w_j \in w_t d_{ij}$ which is a linear sum of the feature values in w_t.

III .OUR METHOD (PREPROCESSING)

There are some issues pertinent to most of the existing feature clustering methods. First, the parameter k, indicating the desired number of extracted features, has to be specified in advance. This gives a burden to the user, since trial-and-error has to be done until the appropriate number of extracted features is found. Second, when calculating similarities, the variance of the underlying cluster is not considered. Intuitively, the distribution of the data in a cluster is an important factor in the calculation of similarity. Third, all words in a cluster have the same degree of contribution to the resulting extracted feature. Sometimes, it may be better if more similar words are allowed to have bigger degrees of contribution. Our feature clustering algorithm is proposed to deal with these issues. Suppose, we are given a document set D of n documents d_1, d_2, . . . , d_n, together with the feature vector W of m words w_1, w_2, . . . , w_m and p classes c_1, c_2, . . . , c_p, as specified. We construct one word pattern for each word in W. For word w_i, its word pattern x_i is defined, similarly as in [27], by

$$X_i = < X_{i1}, X_{i2}, \ldots\ldots\ldots\ldots X_{in} >$$
$$= < P(C1/Wi), P(C2/Wi \ldots\ldots\ldots\ldots\ldots, P(Cn/Wi >$$

for $i \leq j \leq p$.

Note that d_{qi} indicates the number of occurrences of w_i in document d_q

Also, ∂_{qj} is defined as either 1 or 0

Therefore, we have m word patterns in total.

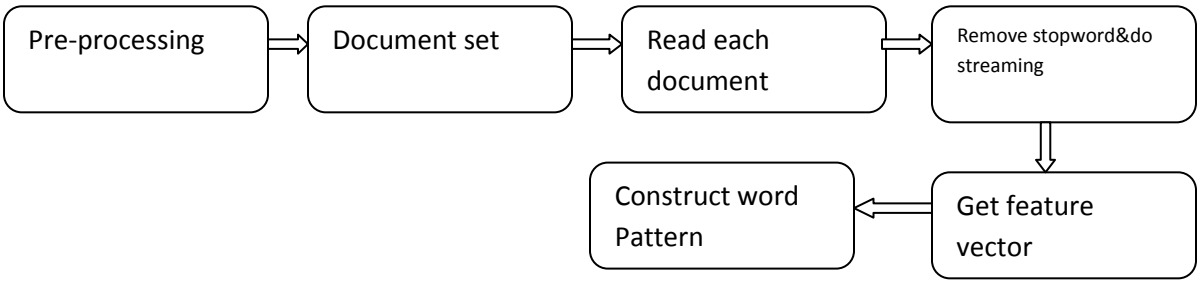

Fig 1: Flowdiagram1

Self-Constructing Clustering

Our clustering algorithm is an incremental, self-constructing learning approach. Word patterns are considered one by one. The user does not need to have any idea about the number of clusters in advance. No clusters exist at the beginning, and clusters can be created if necessary. For each word pattern, the similarity of this word pattern to each existing cluster is calculated to decide whether it is combined into an existing cluster or a new cluster is created. Once a new cluster is created, the corresponding membership function should be initialized. On the contrary, when the word pattern is combined into an existing cluster, the membership function of that cluster should be updated accordingly.

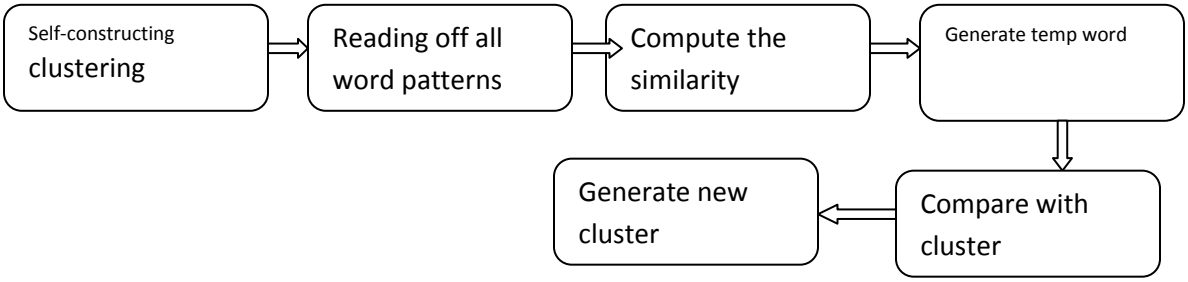

Fig 2: flowdiagram2

Let k be the number of currently existing clusters. The clusters are G_1, G_2, . . .,G_k, respectively. Each cluster G_j has mean m_j=mj1,mj2,…..mjn and deviation σ_j= < σ_{j1} σ_{j2} ……j σm_j >. Let S_j be the size of cluster G_j. Initially, we have k =0. So, no clusters exist at the beginning. For each word pattern

$$X_i = < X_{i1}, X_{i2}, \ldots\ldots\ldots\ldots X_{in} >$$

we calculate the similarity of x_i to each existing clusters, i.e.,

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

Fig 3 :basic main formula from , [40]

for $1 \leq j \leq k$. We say that xi passes the similarity test on cluster G_j if μGJ(Xi) >ρ where p, 0≤p≤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 above function is 2 [34], [35]. 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.

Feature Extraction

Word patterns have been grouped into clusters, and words in the feature vector W are also clustered accordingly. For one cluster, we have one extracted feature. Since we have k clusters, we have k extracted features. The elements of T are derived based on the obtained clusters, and feature extraction will be done. We propose three weighting approaches: hard, soft, and mixed. In the hard-weighting approach, each word is only allowed to belong to a cluster, and so it only contributes to a new extracted feature. In the soft-weighting approach, each word is allowed to contribute to all new extracted features, with the degrees depending on the values of the membership functions. The mixed-weighting approach is a combination of the hard-weighting approach and the soft-weighting approach.

Text Classification

Given a set D of training documents, text classification can be done as follows: Get the training document set and specify the similarity threshold ρ. Assume that k clusters are obtained for the words in the feature vector W. Then find the weighting matrix T and convert D to D`. Using D` 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. SVM is a kernel method, which finds the maximum margin hyperplane 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 hyperplane, but the misclassified patterns should be penalized. **Weka** (Waikato Environment for Knowledge Analysis) is a popular suite of machine learning software written in Java, developed at the University of Waikato, New Zealand. Weka is a collection of machine learning algorithms for data mining tasks. Weka contains tools for data pre-processing, classification, regression, clustering, association rules, and visualization. It is also well-suited for developing new machine learning schemes. Using weak we classify the text.

III CONCLUSIONS

We have presented a fuzzy self-constructing feature clustering (FFC) algorithm, which is an incremental clustering approach to reduce the dimensionality of the features in text classification. Features 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. 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. 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 three real-world data sets have demonstrated that our method can run faster and obtain better extracted features than other methods.

REFERENCES

[1] Http://people.csail.mit.edu/jrennie/ 20Newsgroups/, 2010.
[2] Http://kdd.ics.uci.edu/databases/ reuters21578/reuters21578. html. 2010.
[3] H. Kim, P. Howland, and H. Park, "Dimension Reduction in Text Classification with Support Vector Machines," J. Machine Learning Research, vol. 6, pp. 37-53, 2005.
[4] F. Sebastiani, "Machine Learning in Automated Text Categorization," ACM Computing Surveys, vol. 34, no. 1, pp. 1-47, 2002.
[5] B.Y. Ricardo and R.N. Berthier, Modern Information Retrieval. Addison Wesley Longman, 1999.
[6] A.L. Blum and P. Langley, "Selection of Relevant Features and Examples in Machine Learning," Aritficial Intelligence, vol. 97, nos. 1/2, pp. 245-271, 1997.
[7] E.F. Combarro, E. Montanes, I. Dı´az, J. Ranilla, and R. Mones, "Introducing a Family of Linear Measures for Feature Selection in Text Categorization," IEEE Trans. Knowledge and Data Eng., vol. 17, no. 9, pp. 1223-1232, Sept. 2005.
[8] K. Daphne and M. Sahami, "Toward Optimal Feature Selection," Proc. 13th Int'l Conf. Machine Learning, pp. 284-292, 1996.
[9] R. Kohavi and G. John, "Wrappers for Feature Subset Selection," Aritficial Intelligence, vol. 97, no. 1-2, pp. 273-324, 1997.
[10] 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.

[11] D.D. Lewis, "Feature Selection and Feature Extraction for Text Categorization," Proc. Workshop Speech and Natural Language, pp. 212-217, 1992.
[12] H. Li, T. Jiang, and K. Zang, "Efficient and Robust Feature Extraction by Maximum Margin Criterion," T. Sebastian, S. Lawrence, and S. Bernhard eds. Advances in Neural Information Processing System, pp. 97-104, Springer, 2004.
[13] E. Oja, Subspace Methods of Pattern Recognition. Research Studies Press, 1983.
[14] J. Yan, B. Zhang, N. Liu, S. Yan, Q. Cheng, W. Fan, Q. Yang, W. Xi, and Z. Chen, "Effective and Efficient Dimensionality Reduction for Large-Scale and Streaming Data Preprocessing," IEEE Trans. Knowledge and Data Eng., vol. 18, no. 3, pp. 320-333, Mar. 2006.
[15] I.T. Jolliffe, Principal Component Analysis. Springer-Verlag, 1986.
[16] A.M. Martinez and A.C. Kak, "PCA versus LDA," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 23, no. 2 pp. 228-233, Feb. 2001.
[17] H. Park, M. Jeon, and J. Rosen, "Lower Dimensional Representation of Text Data Based on Centroids and Least Squares," BIT
Numerical Math, vol. 43, pp. 427-448, 2003.
[18] S.T. Roweis and L.K. Saul, "Nonlinear Dimensionality Reduction by Locally Linear Embedding," Science, vol. 290, pp. 2323-2326, 2000.
[19] J.B. Tenenbaum, V. de Silva, and J.C. Langford, "A Global Geometric Framework for Nonlinear Dimensionality Reduction," Science, vol. 290, pp. 2319-2323, 2000.
[20] M. Belkin and P. Niyogi, "Laplacian Eigenmaps and Spectral Techniques for Embedding and Clustering," Advances in Neural Information Processing Systems, vol. 14, pp. 585-591, The MIT Press 2002.
[21] K. Hiraoka, K. Hidai, M. Hamahira, H. Mizoguchi, T. Mishima, and S. Yoshizawa, "Successive Learning of Linear Discriminant Analysis: Sanger-Type Algorithm," Proc. IEEE CS Int'l Conf. Pattern Recognition, pp. 2664-2667, 2000.
[22] J. Weng, Y. Zhang, and W.S. Hwang, "Candid Covariance-Free Incremental Principal Component Analysis," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 25, no. 8, pp. 1034-1040, Aug. 2003.
[23] J. Yan, B.Y. Zhang, S.C. Yan, Z. Chen, W.G. Fan, Q. Yang, W.Y. Ma, and Q.S. Cheng, "IMMC: Incremental Maximum Margin Criterion," Proc. 10th ACM SIGKDD, pp. 725-730, 2004.
[24] L.D. Baker and A. McCallum, "Distributional Clustering of Words for Text Classification," Proc. ACM SIGIR, pp. 96-103, 1998.
[25] 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.
[26] M.C. Dalmau and O.W.M. Flo´ rez, "Experimental Results of the Signal Processing Approach to Distributional Clustering of Terms on Reuters-21578 Collection," Proc. 29th European Conf. IR Research, pp. 678-681, 2007.
[27] 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.
[28] D. Ienco and R. Meo, "Exploration and Reduction of the Feature Space by Hierarchical Clustering," Proc. SIAM Conf. Data Mining, pp. 577-587, 2008.
[29] N. Slonim and N. Tishby, "The Power of Word Clusters for Text Classification," Proc. 23rd European Colloquium on Information Retrieval Research (ECIR), 2001.
[30] F. Pereira, N. Tishby, and L. Lee, "Distributional Clustering of English Words," Proc. 31st Ann. Meeting of ACL, pp. 183-190, 1993.
[31] H. Al-Mubaid and S.A. Umair, "A New Text Categorization Technique Using Distributional Clustering and Learning Logic," IEEE Trans. Knowledge and Data Eng., vol. 18, no. 9, pp. 1156-1165, Sept. 2006.
[32] G. Salton and M.J. McGill, Introduction to Modern Retrieval. McGraw-Hill Book Company, 1983.
[33] T. Joachims, "A Probabilistic Analysis of the Rocchio Algorithm with TFIDF for Text Categorization," Proc. 14th Int'l Conf. Machine Learning, pp. 143-151, 1997.
[34] J. Yen and R. Langari, Fuzzy Logic-Intelligence, Control, and Information. Prentice-Hall, 1999.
[35] J.S. Wang and C.S.G. Lee, "Self-Adaptive Neurofuzzy Inference Systems for Classification Applications," IEEE Trans. Fuzzy Systems, vol. 10, no. 6, pp. 790-802, Dec. 2002.
[36] T. Joachims, "Text Categorization with Support Vector Machine: Learning with Many Relevant Features," Technical Report LS-8-23, Univ. of Dortmund, 1998.
[37] C. Cortes and V. Vapnik, "Support-Vector Network," Machine Learning, vol. 20, no. 3, pp. 273-297, 1995.
[38] B. Scho¨lkopf and A.J. Smola, Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond. MIT Press, 2001.
[39] J. Shawe-Taylor and N. Cristianini, Kernel Methods for Pattern Analysis. Cambridge Univ. Press, 2004.
[40] D.D. Lewis, Y. Yang, T. Rose, and F. Li, "RCV1: A New Benchmark Collection for Text Categorization Research," J. Machine Learning Research, vol. 5, pp. 361-397, http://www.jmlr.org/papers/volume5/lewis04a/le wis04a.pdf, 2004.

