Fuzzy Association Rule Mining Algorithm to Generate Candidate Cluster: An Approach to Hierarchical Document Clustering

Ashish Jaiswal[1], Nitin Janwe[2]

**[1] Department of Computer Science and Engineering, Nagpur University, Rajiv Gandhi College of Engineering, Research and Technology,
Chandrapur, Maharashtra, India**

**[2] Department of Computer Science and Engineering, Nagpur University, Rajiv Gandhi College of Engineering, Research and Technology,
Chandrapur, Maharashtra, India**

Abstract

As text documents are largely increasing in the internet, the process of grouping similar documents for versatile applications have put the eye of researchers in this area. However most clustering methods suffer from challenges in dealing with problems of high dimensionality, scalability, accuracy and meaningful cluster labels. Hierarchical clustering is a solution on that. Proper clustering set generation plays important role in hierarchical clustering. This paper shows the process of generation of candidate cluster set of given document set. Testing has been done on some predefined data set.

Keywords: *Candidate Cluster, Document Clustering, Document Set, Hierarchical Document Clustering.*

1. Introduction

Document clustering is a process of automatic grouping of text documents into clusters in such a way that documents within a cluster have similarity in comparison to one another but are dissimilar to documents in other cluster. There are two methods to cluster the documents i.e. Hierarchical method and Partitioning method. Hierarchical method organizes the clusters into a tree whereas partitioning method partitions the set of documents into a number of clusters by moving documents from one cluster to another [1].

Hierarchical method can be further classified as Agglomerative and Divisive Hierarchical Clustering depending on whether the hierarchical decomposition is formed in a bottom- up or top down fashion. Steinbach showed that Unweighted Pair Group Method with Arithmetic Mean (UPGMA) is the most accurate one in Agglomerative category [2].

K- means and its variants are the most well known partitioning method that create a non hierarchical clustering consists of K clusters. Steinbach shows that Bisecting k means outperforms basic k means as well as agglomerative hierarchical clustering in terms of accuracy and efficiency [2].

Hierarchical clustering is often portrayed as the better quality clustering approach, but is limited because of its quadratic time complexity. In contrast, K means and its variants have a time complexity that is linear in the number of documents, but are thought to produce inferior clusters. Thus K means is used because of run time efficiency and agglomerative hierarchical clustering is used because of its quality. Sometimes K means and agglomerative hierarchical approaches are combined so as to get best of both worlds [2].

Today thousands of documents are stored and retrieved electronically. A typical search on World Wide Web can return thousands of documents. Clustering is used to categorize the document databases. Automatic grouping of the documents enables the user to have a clear and easy grasp of what kind of documents are retrieved, providing tremendous help for the user to locate right information. Thus, Many clustering algorithms have been proposed to cluster the documents but most of them still suffer from challenges in dealing with the problems of high dimensionality, scalability, accuracy, ease of browsing and meaningful cluster labels.

The rest of the paper is organized as follows: Section 2 represents the related work in the field; Section 3 represents the proposed approach for candidate cluster generation; Section 4 shows the experimental results based on proposed approach. Section 5 outlines the future direction and we conclude the paper in Section 6.

2. Related Work

The hierarchical Frequent Term based clustering (HFTC) proposed by Beil, Ester and Xu in 2002 attempts to address the special requirements in document clustering using the notion of frequent itemsets. HFTC greedily selects the next frequent itemset which represents the next cluster, minimizing the overlap of clusters in terms of shared documents. The clustering result depends on the order of the selected itemsets, which in turn depends on the greedy heuristics used. Experiments show that HFTC is not scalable [3].

Benjamin C. M Fung, Ke Wang, Martin Ester in 2003 proposed an algorithm to use the notion of frequent itemsets which comes from association rule mining for document clustering. Each cluster is identified by some words called frequent itemsets for the documents in the cluster. Frequent itemsets are also used to produce hierarchical topic tree structure for clusters. By focusing on frequent item the dimensionality of documents set is reduced. This method outperforms best in terms of both clustering accuracy and scalability [3].

Chun- Ling Chen, Frank S.C. Tseng, Tyne Liang in 2010 proposed an effective fuzzy frequent itemset based hierarchical clustering approach which uses fuzzy frequent itemsets discovered by fuzzy association rule mining to improve the clustering accuracy of FIHC. Algorithm works in three stages. In the first stage the key terms will be retrieved from the document set for removing noise, and each document is pre-processed into the designated representation for the following mining process. In the second stage, a fuzzy association rule mining algorithm is employed to discover a set of highly relevant fuzzy frequent itemsets, which contains key terms to be regarded as the labels of candidate clusters. In the final stage, the documents will be clustered into a hierarchical cluster tree based on these candidate clusters. The obtained hierarchical cluster tree with meaningful cluster descriptions can offer users a more flexible ability in document management. [4]

Rakesh Agrawal, Tomasz Imielinski, Arun Swami [5] in 1993 present an efficient algorithm that generates all significant association rules between items in the database. The algorithm incorporates buffer management and novel estimation and pruning techniques. The algorithm uses a carefully tuned estimation procedure to determine what itemsets should be measured in a pass. The algorithm makes multiple passes over the database. In each pass, the support for certain itemsets is measured. These itemsets are called as candidate itemsets. This procedure strikes a balance between the number of passes over the data and the number of itemsets that are measured in a pass. If large numbers of itemsets are measured in a pass and many of them turn out to be small then there is measurement effort wastage. Conversely if small numbers of itemsets are measured and many of them turn out to be large then there are unnecessary passes.
Another feature of this algorithm is that it uses pruning techniques to avoid measuring certain itemsets, while guaranteeing completeness. These are the itemsets that the algorithm can prove will not turn out to be large. There are two pruning techniques. The first one is “remaining tuple optimization” and the second is “pruning function optimization”. These pruning functions can prune out itemsets as they are generated.
The algorithm incorporates buffer management to handle the fact that all the itemsets that need to be measured in a pass may not fit in memory, even after pruning. When memory fills up, certain itemsets are deleted and measured in the next pass in such a way that the completeness is maintained.
The algorithm exhibited excellent performance on the sales data. The estimation procedure exhibited high accuracy and the pruning techniques were able to prune out a very large fraction of itemsets without measuring them.

Rakesh Agrawal, Ramakrishnan Shikant [6] in 1994 considered the problem of discovering association rule between items in large database of sales transaction. The authors presented two algorithms, Apriori and AprioriTid for discovering all significant association rules between items in large databases of transactions. Experimental results show that both the algorithms always outperform the other known algorithms. Empirical evaluations shows that these algorithms outperforms the known algorithms by factors ranging from three for small problems to more than an order of magnitude for large problems. The best features of two algorithms can be combined into a hybrid algorithm called AprioriHybrid. Scale-up experiments showed that AprioriHybrid scales linearly with the number of transactions. The execution time decreases a little as the number of items in the database increases. As the average transaction size increases while keeping the database size constant, the execution time increases only gradually. Experiments demonstrate the feasibility of using AprioriHybrid in real applications involving very large databases.

Yuepeng Cheng, Tong Li and Song Zhu [7] in 2010 proposed a document clustering technique based on term clustering and association rules. In this technique words are extracted from document collection and then term are constructed according to Average Mutual Information (AMI) between terms. Document Vector Space Model is represented by term clustering and then association rules are used to mine document clustering. This technique is compared with the k means and k-medoids to test the

effectiveness, and then Average Difference Degree (ADD) and Average Similar Degree (ASD) of document clustering and cost of time of clustering process of these three methods are tested. Experimental results show that performance and clustering quality of this technique are improved than other mentioned methods of clustering process.

Arnaud Ribert, Abdel Ennaji, Yves Lecourtier [8] in 1999 proposed an algorithm to treat time incremental data by a hierarchical clustering. This method proceeds by updating the hierarchical representation of the data instead of re-computing the whole tree when new patterns have to be taken into account. Experimental results shows that the algorithm allows to progressively perform hierarchical clustering of big sets of data, which contains seven times more elements as compared to classical algorithm. The incremental algorithm described is a first stage towards the use of hierarchical clustering in industrial applications which is an alternative to partitioning clustering methods.

Xiaoke Su, Yang Lan. Renxia Wan and Yuming Qin [9] in 2009 proposed a fast incremental hierarchical clustering algorithm which is found to be feasible and effective. The existed incremental clustering algorithm does not take the memory constraint into account and it is difficult to obtain a satisfy result when it is used for large-scale data sets. A fast clustering algorithm is presented by changing the radius threshold value dynamically. The clustering result is no longer spherical shape. At the same time an inter-cluster dissimilarity measure is put forward which is capable of handling the categorical data. Theoretical analysis and experimental results show the algorithm can not only overcome the impact of the inadequate of the memory when clustering the large scale data set, but also accurately reflect the characteristics of the data set. Both of these indicate the effectiveness of the algorithm. Clustering with the fixed final clusters number will show a reliable rationality, and can be used for ultra-large-scale data set, particularly for the data stream environment.

E. Gothai, Dr. P. Balasubramanie [10] in 2010 proposed a new method of clustering called Multilevel Clustering which is a combination of supervised and an unsupervised technique for forming the cluster. It is an incremental stream of hierarchical clustering which improves the efficiency, reduces the time consumption and accuracy of text categorization algorithm by forming an exact sub clustering. The authors presented a survey of existing techniques used for duplicate entries in database records and proposed Multilevel Clustering and proved that it is better than existing algorithm. The algorithm works in four phases and requires very small amount of memory.

S. Murali Krishna, S. Durga Bhavani [11] in 2010 presented an extensive analysis of frequent itemset based text clustering approach for different real life datasets and the performance of the frequent itemset based text clustering approach is evaluated with the help of evaluation measures such as, Precision, Recall and F-measure. The text clustering approach consists of text preprocessing, Mining of frequent itemsets which uses Apriori algorithm, Partitioning text documents based on frequent itemsets and clustering the text documents within the partition. The experimental result shows that the efficiency of the frequent itemset based text clustering approach has been improved significantly for different real life datasets. The approach effectively groups the documents into cluster and provides better precision.

3. Proposed Approach

3.1 Term Document Matrix

The Term Document Matrix is the way of representation of documents. each document d is represented by the TF vector, dtf = (tf1, tf2,…., tfn), where tfi is the frequency of the ith term in the document. A collection of d documents described by t terms can be represented as t × d matrix A, referred as term document matrix. To get the representation of all documents in a document set D, a document pre processing algorithm is used. The input to this algorithm is:

- a set of Documents D = {d_1,d_2,d_3,….d_i,….d_n}.
- A stop word list: the stop words are frequent words that carry no information and meaningless when used as a search term (i.e. pronouns, prepositions, conjunctions etc). These words occur too frequently in a document and are usually ignored by the system when searching is done. Stop words are eliminated using a list of stop words. If a word in the document matches a word in the stop list, then the word will not be included as part of clustering. An advantage of using stop words removal is that it reduces the number of terms that identifies the document.
- The minimum threshold value p of term frequency × inverse document frequency. It is used for the measure of importance of term t_j within document d_i.
- The minimum threshold value q of term frequency × document frequency. It is evaluated by dividing the term frequency by its document frequency and
- The minimum threshold value r of product of term frequency × inverse document frequency and term frequency × document frequency

The output of the algorithm is

- The key term set of document set D, K_D
- The representation of all the documents in the form of term document matrix

The method of working of algorithm is:

- Extract all the terms set (i.e. set of terms appeared in document set D)
- Remove all the stop words from the term set
- Apply Word stemming algorithm which reduces the word to its stem or root form. For example words "association", "associating" are all reduced to the stem "associate". i.e prefixes and suffixes of each word are removed.
- Now for each document which belongs to the document set and for each term which belong to the term set, evaluate its term frequency × inverse document frequency, term frequency × document frequency and (term frequency)^2 weights.
- Now take the term set if they are greater than the threshold values of p, q and r. the term set which is obtained is called Key term set.
- Now for each document which belongs to document set D and each term which belong to key term set, count its frequency in document d_i to obtain the document with its term and its frequency.

This stage describes the required transformation process of documents to obtain desired representation of documents. As there are thousands of words in the document set, the purpose of this stage is to reduce the dimensionality of high clustering accuracy. To solve the problem of dimensionality we have find the terms that are significant and important to represent the content of each document. Hence we must remove the terms that are not meaningful and discriminative to increase the clustering accuracy and maintain the computing cost small.

3.2 Extraction of Candidate Cluster

Effective candidate cluster generation is very important in hierarchical document clustering. We have used membership functions to convert the matrix obtained above into a fuzzy set. Those membership functions are as follows.

$$L_{ij}(n_{ij}) = \begin{cases} 0, & n_{ij} = 0 \\ 1+ n_{ij}/x_1, & 0 < n_{ij} < x_1 \\ 2, & n_{ij} = x_1 \quad , \quad x_1 = \min(n_{ij}), \; x_2 = \text{avg}(n_{ij}) \\ 1 + x_2 - n_{ij} / x_2 - x_1, & x_1 < n_{ij} < x_2 \\ 1, & n_{ij} \geq x_2 \end{cases}$$

$$M_{ij}(n_{ij}) = \begin{cases} 0, & n_{ij} = 0 \\ 1, & n_{ij} = 1 \\ 1 + n_{ij} - x_1 / x_2 - x_1, & x_1 < n_{ij} < x_2 \\ 2, & n_{ij} = x_2 \quad , \quad x_1 = \min(n_{ij}), \; x_2 = \text{avg}(n_{ij}), \; x_3 = \max(n_{ij}) \\ 1 + x_3 - n_{ij} / x_3 - x_2, & x_2 < n_{ij} < x_3 \\ 1, & n_{ij} = x_2 \end{cases}$$

$$H_{ij}(n_{ij}) = \begin{cases} 0, & n_{ij} = 0 \\ 1, & n_{ij} \leq x_1 \\ 1+ n_{ij} / x_2 - x_1, & x_1 < n_{ij} < x_2 \, , \quad x_1 = \text{avg}(n_{ij}), \; x_2 = \max(n_{ij}) \\ 2, & n_{ij} = x_2 \end{cases}$$

Where N_{ij} is a set and equals to $\{L_{ij}(n_{ij})/t_j$. Low, $M_{ij}(n_{ij})/t_j$. Mid, $H_{ij}(f_{ij})/t_j$. High$\}$. The notation t_j . z is called fuzzy region of t_j where z can be Low Mid or High. Min (n_{ij}) is the minimum frequency of terms in D, Max (n_{ij}) is the maximum frequency of terms in D and avg (n_{ij}) is the average frequency of terms in D.

Candidate cluster is identified for each key terms which follows certain rules. It includes the set of documents those contains equivalent key term. The candidate cluster set of document is the set of candidate clusters. The objective of this stage is to take a document set D, a set of predefined membership functions, the minimum support value θ and the minimum confidence value λ as input and the output is a set of candidate clusters. Algorithm discovers fuzzy frequent itemset that has an associated fuzzy count value which is regarded as the degree of importance that the itemset contributes to the document set. Following steps generates the candidate cluster.

Input:

- A document set D
- Set of membership functions
- The minimum support value θ
- The minimum confidence value λ

Method:

Step1. Calculate fuzzy value by using membership function

Step2. Calculate scalar cardinality of three fuzzy results.

Step3. Find the region of each key term with maximum count.

Step4. Find fuzzy frequent 1- itemset L_1 which has a support value greater than θ.

Step5. Generate candidate set C_2 contains a set of documents for each key term.

Step6. Find fuzzy frequent-2 itemsets L_2 which is the pair of key terms.

6.1 Calculate minimum value by comparing the fuzzy value of the pair.

6.2 Calculate scalar cardinality in D as count.

6.3 Put those regions which has larger value than θ.

Step7. Generate C_3 from L_2

Step8. Only those pairs in step 6 is to be considered which has a confidence value greater than λ.

4. Experimental Work

We have performed above algorithm on three data sets, Classic, Classic 30 and Tr11. We have find the total number of clusters generated in L1 and L2. The said algorithm has been implemented in Matlab 7.9. Following table shows the statistics of test dataset.

Table 1: Experimental Results

Dataset	***Number of Documents***	***Number of key terms***	***Clusters from L1***	***Clusters from L2***
Classic 30	30	1073	291	40957
Classic	7094	41684	49658	9683310
Tr11	414	6424	2898	565110

5. Future Work

Our future work focus on the construction of hierarchical tree based on these generated candidate clusters.

6. Conclusion

In this paper we have proposed the technique of the generation of the Candidate clusters. Fuzzy association rule mining algorithm has been used to generate those clusters. Membership functions are used to convert term document matrix into fuzzy set. This work will be useful in hierarchical clustering which clusters the documents on the basis of generated candidate cluster set.

References

[1] Benjamin C. M. Fung, Ke Wang, and Martin Ester, Simon Frazer University, Canada, "Hierarchical Document Clustering"

[2] M. Steinbach, G. Karypis, Vipin Kumar, "A Comparison of Document Clustering Techniques"

[3] B. Fung, K.Wang and M. Ester, " Hierarchical Document Clustering using frequent itemsets", In Proc. SIAM International Conference on Data Mining, 2003, pp. 59-70

[4] Chun- Ling Chen, Frank S.C. Tseng, Tyne Liang, " Mining Fuzzy Frequent itemset sets for Hierarchical Document Clustering" International Journal of Information Processing and Management 46(2010) 193-211

[5] Rakesh Agrawal, Tomasz Imielinski, Arun Swami; "Mining Association Rules between Sets of Items in Large Database", Proceedings of the 1993 ACM SIGMOD Conference Washington DC, USA, May 1993.

[6] Rakesh Agrawal, Ramakrishna Srikant " Fast Algorithm for mining Association Rules", Proceedings of the 20th VLDB Conference Santiago, Chile, 1994

[7] Yuepeng Cheng, Tong Li, Song Zhu, " Document Clustering Technique based on Term Clustering and Association Rules", 978-1-4244-6977-2/10 © 2010 IEEE

[8] Arnaud Ribert, Abdel Ennaji, Yves Lecourtier, "An Incremental Hierarchical Clustering" Vision Interface '99, Trios-Rivieres, Canada, 19-21 May

[9] X.Su, Y. Lan, R.. Wan, Y. Qin, " A Fast Incremental Clustering Algorithm", Proceedings of the 2009 International Symposium on Information Processing (ISIP'09), Huangsham, P.R.China, August 21-23, 2009 pp 175-178

[10] E. Gothai, Dr. P. Balasubramanie, " Performance Evaluation of Hierarchical Clustering Algorithms" Proceedings of the International Conference on Communication and Computational intelligence-2010. Kongu Engineering College, Perundura, Erode, T.N., India 27-29 December 2010 pp 457- 460

[11] S. Murali Krishan, S. Durga Bhavani, " Performance Evaluation of an Efficient Frequent Items sets based text clustering approach", pp 60-68 Global Journal of Computer Science and Technology, Vol. 10 Issue 11 (Ver. 1.0) October 2010.

[12] Yu Steck, M. Lobur, Faisal M.E. Sardieh, M. Dombrova, V, Artsibasov, "Development and Study of Clustering Algorithms for Large sets of Data", CADSM'2011, 23-25 February, 2011, Polyana-Svalyava (Zakarpattya), UKRAINE, pp. 202-204

[13] V. Mary, Amala Bai, Dr. D. Manimegalai, " An Analysis of Document Clustering Algorithms", ICCCCT-10, 978-1-4244-7770-8/10/ © 2010 IEEE

Ashish Jaiswal received his BE Degree in Computer Engineering from Nagpur University, Nagpur (M.S), India in 2006. He is currently pursuing post graduate in Computer Science and

Engineering from Rajiv Gandhi College of Engineering, Research and Technology, Chandrapur (M.S), India. He is majoring in Computer Science and is familiar with Data Mining. His research area includes Data Mining and Mobile Computing.

Prof. Nitin J. Janwe received his BE Degree in Computer Technology from Shri Guru Gobind Singhji College of Engg. & Technology, Nanded in 1991. He completed his post graduation in Computer Science and Engineering from Nagpur University, Nagpur (M.S), India in 2007. He is head of Computer Science and Engineering Department at Rajiv Gandhi College of Engineering, Research and Technology, Chandrapur, Maharashtra, India. He is majoring in Computer Science and is familiar with Image and Video Processing. His research area includes Image and Video Processing, Machine Vision & Learning, Computer Graphics and Operating Systems.