Investigate the Performance of Document Clustering Approach Based on Association Rules Mining

Noha Negm
Math. And Computer Science Dept.
Faculty of Science, Menoufia University
Shebin El-Kom, EGYPT

Passent Elkafrawy
Math. And Computer Science Dept.
Faculty of Science, Menoufia University
Shebin El-Kom, EGYPT

Mohamed Amin
Math. And Computer Science Dept.
Faculty of Science, Menoufia University
Shebin El-Kom, EGYPT

Abdel Badeeh M. Salem
Computer Science Dept. Faculty of Computers and Information, Ain Shams University
Cairo, EGYPT

Abstract—**The challenges of the standard clustering methods and the weaknesses of Apriori algorithm in frequent termset clustering formulate the goal of our research. Based on Association Rules mining, an efficient approach for Web Document Clustering (ARWDC) has been devised. An efficient Multi-Tire Hashing Frequent Termsets algorithm (MTHFT) has been used to improve the efficiency of mining association rules by targeting improvement in mining of frequent termset. Then, the documents are initially partitioned based on association rules. Since a document usually contains more than one frequent termset, the same document may appear in multiple initial partitions, i.e., initial partitions are overlapping. After making partitions disjoint, the documents are grouped within the partition using descriptive keywords, the resultant clusters are obtained effectively. In this paper, we have presented an extensive analysis of the ARWDC approach for different sizes of *Reuter's* datasets. Furthermore the performance of our approach is evaluated with the help of evaluation measures such as, *Precision*, *Recall* and *F-measure* compared to the existing clustering algorithms like Bisecting K-means and FIHC. The experimental results show that the efficiency, scalability and accuracy of the ARWDC approach has been improved significantly for *Reuters* datasets.**

Keywords—Web Document Clustering; Knowledge Discovery; Association Rules Mining; Frequent termsets; Apriori algorithm; Text Documents; Text Mining; Data Mining

I. INTRODUCTION

The internet has become the largest data repository, facing the problem of information overload. The existence of an abundance of information, in combination with the dynamic and heterogeneous nature of the Web, makes information retrieval a tedious process for the average user. Search engines, Meta-Search engines and Web Directories have been developed in order to help the users quickly and easily satisfy their information need. The Search engine performs exact matching between the query terms and the keywords that characterize each web page and presents the results to the user. These results are long lists of URLs, which are very hard to search. Furthermore, users without domain expertise are not familiar with the appropriate terminology thus not submitting the right query terms, leading to the retrieval of more irrelevant pages. This has led to the need for the development of new techniques to assist users effectively navigate, trace and organize the available web documents, with the ultimate goal of finding those best matching their needs. Document Clustering is one of the techniques that can play an important role towards the achievement of this objective.

Document clustering has become an increasingly important task in analyzing huge numbers of documents distributed among various sites. Furthermore organizing them into different groups called as clusters, where the documents in each cluster share some common properties according to defined similarity measure. The fast and high-quality document clustering algorithms play an important role in helping users to effectively navigate, summarize, and organize the information.

Document clustering has been studied intensively because of its wide applicability in areas such as Web Mining, Search Engines, Information Retrieval, and Topological Analysis. Document Clustering is different than document classification. In document classification, the classes (and their properties) are known a priori, and documents are assigned to these classes; whereas, in document clustering, the number, properties, or membership (composition) of classes is not known in advance. Thus, classification is an example of supervised machine learning and clustering that of unsupervised machine learning [1]. This distinction is illustrated in figure (1). Document Clustering can produce either disjoint (hard clustering) or overlapping (soft clustering) partitions. In an overlapping partition, it is possible for a document to appear in multiple clusters whereas in disjoint clustering, each document appears in exactly one cluster [2].

Document clustering algorithms may be divided into two groups: Hierarchical algorithms produce a hierarchy of clusters, while Partitioning algorithms give a flat partition of the set.

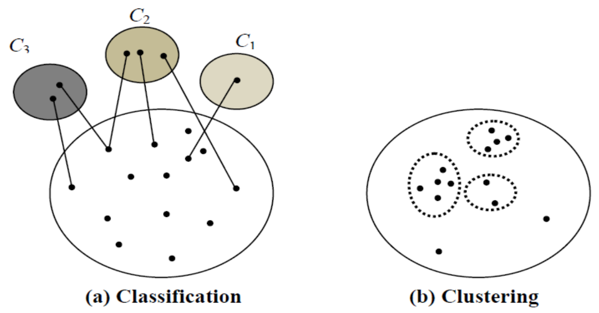

Fig. 1. In (a), three classes are known a priori, and documents are assigned to each of them. In (b), an unknown number of groupings must be inferred from the data based on a similarity criterion [1].

Although standard clustering techniques such as k-means can be applied to document clustering, they usually do not satisfy the special requirements for clustering documents: high dimensionality, high volume of data, ease for browsing, and meaningful cluster labels. In addition, many existing document clustering algorithms require the user to specify the number of clusters as an input parameter [3]-[8]. Incorrect estimation of the value always leads to poor clustering accuracy. Furthermore, many clustering algorithms are not robust enough to handle different types of document sets in a real-world environment. In some document sets, cluster sizes may vary from few to thousands of documents. This variation tremendously reduces the resulting clustering accuracy for some of the state-of-the art algorithms.

The challenges of hierarchical clustering and the weaknesses of the standard clustering methods formulate the need for an accurate, efficient, and scalable clustering method that addresses the special challenges of document clustering. Frequent itemset-based clustering method is shown to be a promising method for high dimensionality clustering in recent literature. It reduces the dimension of a vector space by using only frequent itemsets for clustering. Frequent itemsets form the basis of association rule mining [9]. Exploiting the property of frequent itemsets (each subset of a frequent itemset is also frequent) and using data structures supporting the support counting, the set of all frequent itemsets can be efficiently determined even for large databases. Recent studies on frequent termsets in text mining fall into two categories. One is to use Association Rules to conduct text categorization [10,11] and the other one is to use frequent itemsets for text clustering [12]-[26].

In our prior research [27], we have presented an efficient Association Rules-based Web Document Clustering approach (ARWDC). The main idea of the association rule-based clustering stage is based on a simple observation: the documents under the same topic should share a set of common keywords. Some minimum fraction of documents in the document set must contain these common keywords, and they correspond to the notion of frequent termsets which form the basis of the initial clusters. An essential property of frequent termset is its representation of words that commonly occur together in documents.

To illustrate that this property is important for clustering, we consider two frequent terms, “apple” and “window”. The documents that contain the word “apple” may discuss about fruits or farming. While the documents that contain the word “window” may discuss about renovation. However, if we found association rules between both words occur together in many documents, then we may identify another topic that discusses about operating systems or computers. By precisely identifying these hidden topics as the first step and then clustering documents based on them, we can improve the accuracy of the clustering solution.

The Apriori algorithm remains the most commonly used algorithm in the mining process [9]. The Apriori achieves good reduction on the size of candidate set but still suffers from generating huge numbers of candidates and taking many scans of large databases for frequency checking. Our MTHFT algorithm proposed in [28] for efficient mining of association rules from documents instead of Apriori algorithm. Since by using MTHFT algorithm, the scanning cost and computational cost is improved moreover the performance is considerably increased furthermore increase up the clustering process.

In this paper, we have presented an extensive analysis of the ARWDC approach for different sizes of Reuters datasets. Furthermore the performance of the approach is compared with the existing two clustering algorithms like Bisecting K-means and FIHC and evaluated with the help of evaluation measures such as, Precision, Recall and F-measure.

The organization of the paper is as follows. The concise review of related researches is presented in Section 2. The ARWDC approach based on association rules is described in Section 3. The extensive analysis of the ARWDC approach using different sizes of Reuters datasets moreover the comparison with other clustering algorithms are given in section 4. The conclusion is summed up in Section 5 and the future work in Section 6.

II. Review Of Litruture

In data mining literature, there are limited researches for clustering the data based on association rules mining. Whereas all researches for clustering web documents based on frequent termsets are conducted in web mining field. A review of researches and the work that has been done are presented in this section.

Association Rules Mining is considered the basis of data mining research [9], [29]. The first method of integrating association rules and clustering techniques in an undirected hypergraph is presented in [30]. The frequent itemsets were modeled as hyperedges and a min-cut hypergraph partitioning algorithm was used to cluster items. There has been some theoretical work relating hypergraphs with association rules [31]. Directed hypergraphs [32],[33] extend directed graphs and have been used to model many-to-one, one-to-many and many-to-many relationships in theoretical computer science and operations research.

The method for clustering of data in a high dimensional space based on a hypergraph model is proposed in [34]. In a hypergraph model, each data item represented as a vertex and related data items connected with weighted hyperedges. A hyperedge represented a relationship (affinity) among subsets of data and the weight of the hyperedge reflected the strength of this affinity. A hypergraph partitioning algorithm used to find a partitioning of the vertices such that the corresponding data items in each partition were highly related and the weight of the hyperedges cut by the partitioning minimized. The method is linearly scalable with respect to the number of dimensions of data and items, provided the support threshold used in generating the association rules is sufficiently high. it suffers from the fact that right parameters are necessary to find good clusters.

An algorithm to mine association rules from medical data based on digit sequence and clustering is presented in [35]. The entire database divided into partitions of equal size, each partition called cluster. Each cluster considered one at a time by loading the first cluster into memory and calculating frequent itemsets. Then the second cluster considered similarly and calculating frequent itemsets. This approach reduced main memory requirement since it considered only a small cluster at a time and it is scalable and efficient.

The first criterion for clustering transactions using frequent itemsets, instead of using a distance function is presented in [25]. In principle, this method can also be applied to document clustering by treating a document as a transaction; however, the method does not create a hierarchy for browsing. The novelty of this approach is that it exploits frequent itemsets (by applying Apriori algorithm) for defining a cluster, organizing the cluster hierarchy, and reducing the dimensionality of document sets.

The two clustering algorithms, FTC and HFTC, are proposed in [12]. The basic motivation of FTC is to produce document clusters with overlaps as few as possible. FTC works in a bottom-up fashion. As HFTC greedily picks up the next frequent itemset to minimize the overlapping of the documents that contain both the itemset and some remaining itemsets. The clustering result depends on the order of picking up itemsets, which in turn depends on the greedy heuristic used. The weakness of the HFTC algorithm is that it is not scalable for large document collections.

To measure the cohesiveness of a cluster directly using frequent itemsets, the FIHC algorithm is proposed in [14]. Two kinds of frequent item are defined in FIHC: global frequent item and cluster frequent item. However, FIHC has three disadvantages in practical application: first, it cannot solve cluster conflict when assigning documents to clusters. Second, after a document has been assigned to a cluster, the cluster frequent items were changed and FIHC does not consider this change in afterward overlapping measure. Third, in FIHC, frequent itemsets is used merely in constructing initial clusters.

Frequent Term Set-based Clustering (FTSC) algorithm is introduced in [15]. FTSC algorithm used the frequent feature terms as candidate set and does not cluster document vectors with high dimensions directly. The results of the clustering texts by FTSC algorithm cannot reflect the overlap of text classes. But FTSC and the improvement FTSHC algorithms are comparatively more efficient than K-Means algorithm in the clustering performance.

The document clustering algorithm on the basis of frequent termsets is proposed in [22]. Initially, documents were denoted as per the Vector Space Model and every term is sorted in accordance with their relative frequency. Then frequent term sets can be mined using frequent-pattern growth (FP growth). Lastly, documents were clustered on the basis of these frequent term sets. The approach was efficient for very large databases, and gave a clear explanation of the determined clusters by their frequent term sets. The efficiency and suitability of the proposed algorithm has been demonstrated with the aid of experimental results.

To the best of our knowledge, all previous researchers depend on the frequent termsets for clustering web documents. While we do not know of any research that exploits association rules in web document clustering.

III. Association Rules based Clustering Approach

An effectual approach for clustering a web documents with the aid of association rules is discussed in this section[27]. The ARWDC approach as shown in figure (3) consists of the following major stages:

- Offline Collecting of Documents
- Document Preprocessing
- Association Rules Mining
- Document Clustering
- Post Processing

A. *Offline Collecting of Documents stage*

The first step in the ARWDC approach is collecting and analyzing the documents (i.e. the relevant documents). The process of selecting documents in the ARWDC approach is done offline that means the documents are previously downloaded. The largest Reuters datasets is an example for offline documents [36]. The Reuters-21578 collection is distributed in 22 files. Each of the first 21 files (reut2-000.sgm through reut2-020.sgm) contain 1000 documents, while the last (reut2- 021.sgm) contains 578 documents. Documents were marked up with SGML tags. There are 5 categories Exchanges, Organizations, People, Places and Topics in the Reuters dataset and each category has again sub categories in total 672 sub categories. We have collected the TOPIC category sets to form the dataset. The TOPICS category set contains 135 categories. From these documents we collect the valid text data of each category by extracting the text which is in between <BODY> ,</BODY> and placed in a text document and named it according to the topic.

B. *Document Preprocessing stage*

Preprocessing stage is a very important step since it can affect the result of a clustering algorithm. So it is necessary to pre-process the data sensibly. Preprocessing have the several steps that take a text document as input and output as a set of tokens to be used in feature vector. It begins after collecting the documents that need to be clustered. The ARWDC approach employs several pre-processing steps including stop words removal, stemming on the document set and indexing documents by applying TF*ID:

- Stop words removal: In this process, the documents are filtered by removing the stop-words from documents content and reduce noise. Stop-words are words that from non-linguistic view do not carry information such as (a, an, the, this, that, I, you, she, he, again, almost, before, after). One major property of stop-words is that they are extremely common words.
- Stemming: Removes the prefixes and suffixes in the words and produces the root word known as the stem. Typically, the stemming process will be performed so that the words are transformed into their root form [37]. A good stemmer should be able to convert different syntactic forms of a word into its normalized form, reduce the number of index terms, save memory and storage and may increase the performance of clustering algorithms to some extent; meanwhile it should try stemming. Porter Stemmer [38] is a widely applied method to stem documents. It is compact, simple and relatively accurate. It does not require to create a suffix list before applied. In this paper, we apply Porter Stemmer in our pre-processing .
- Indexing documents: the indexing process has done on the filtered and stemmed documents. The documents indexed automatically by labelling each document by a set of the most important words with their frequencies. The techniques for automated production of indexes associated with documents usually rely on frequency-based weighting schema. The weighting schema is used to index documents and to select the most important words in all document collections. The purpose of weighting schema is to reduce the relative importance of high frequency terms while giving a higher weight value for words that distinguish the documents in a collection. The weighting scheme TF-IDF (Term Frequency, Inverse Document Frequency) is used to assign higher weights to distinguished terms in a document, and it is the most widely used. Weighting scheme is defined as [39]:

$$w(i,j) = tfidf\,(d_i, t_j) = \begin{cases} Nd_i, t_j * \log_2 \dfrac{|C|}{Nt_j} & \text{if } Nd_i, t_j \geq 1 \\ 0 & \text{if } Nd_i, t_j = 0 \end{cases} \quad (1)$$

where w(i,j) ≥ 0, Nd_i, t_j denotes the number the term t_j occurs in the document d_i (term frequency factor), Nt_j denotes the number of documents in collection C in which t_j occurs at least once (document frequency of the term t_j) and |C| denotes the number of the documents in collection C. The first clause applies for words occurring in the document, whereas for words that do not appear (Nd_i, t_j =0), we set w (i,j)=0. The weighting scheme includes the intuitive presumption that is: the more often a term occurs in a document, the more representative of the content of the document (term frequency). Moreover the more documents the term occurs in, the less discriminating it is (inverse document frequency). Once a weighting scheme has been selected, automated indexing can be performed by simply selecting the words that satisfy the given weight constraints for each document. The major advantage of an automated indexing procedure is that it reduces the cost of the indexing step. For each document, we store all words, with their frequencies and their calculated weighing values. Next, the words that have zero weighted value were eliminated automatically and select only the words that satisfy the given weighting threshold. Finally, the words (the number of words that satisfy the threshold weight value) taken as the final set of words to be used in the Association Rule Mining stage. This is the criteria of using the weight constraints.

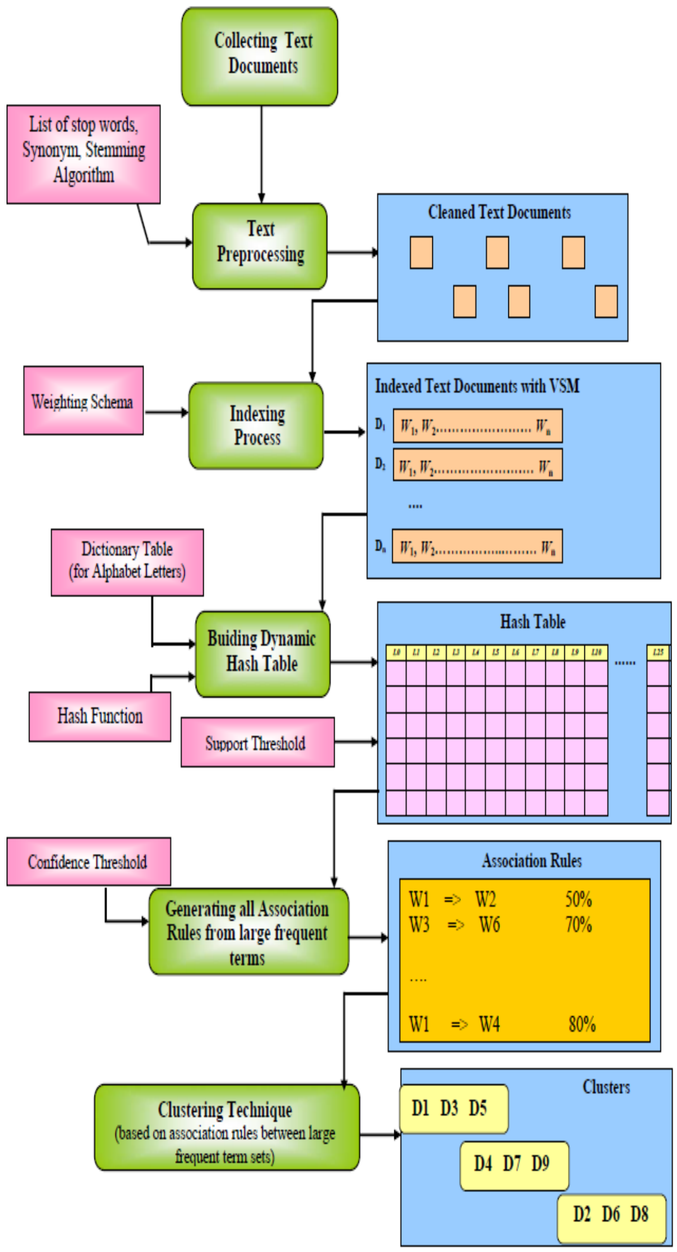
Collecting Text Documents
List of stop words, Synonym, Stemming Algorithm
Text Preprocessing
Cleaned Text Documents
Weighting Schema
Indexing Process
Indexed Text Documents with VSM
D1 W1, W2....................... Wn
D2 W1, W2....................... Wn
....
Dn W1, W2....................... Wn
Dictionary Table (for Alphabet Letters)
Buiding Dynamic Hash Table
Hash Function
Support Threshold
Hash Table
Confidence Threshold
Generating all Association Rules from large frequent terms
Association Rules
W1 => W2 50%
W3 => W6 70%
....
W1 => W4 80%
Clustering Technique (based on association rules between large frequent term sets)
Clusters
D1 D3 D5
D4 D7 D9
D2 D6 D8

Fig. 2. ARWDC approach.

C. Association Rules Mining Stage

Association rules can be used to solve the problem of finding clusters of similar items. For instance, in market-basket type data, a practical application of association rules is to identify clusters of similar items based on the customer sales information. This helps to understand patterns in sales of items and to group items based on customer interests. Association rule mining is to find out association rules that satisfy the predefined minimum support and confidence from a given database. The problem is usually decomposed into two sub-problems: 1) One is to find those itemsets whose occurrences exceed a predefined threshold in the database; those itemsets are called frequent or large itemsets, 2) The second problem is to generate association rules from those large itemsets with the constraints of minimal confidence.

Apriori algorithm considered to be the basic for all developed algorithms to solve the first problem. However there are two drawbacks of the Apriori algorithm. One is the complex candidate generation process that uses most of the time, space and memory. Another drawback is the multiple scan of the database. Although the drawbacks of the Apriori algorithm, it still use for generating the frequent termsets that used in the document clustering. In order to speed up the mining process as well as to address the scalability with different documents regardless of their sizes, we used our algorithm [28] called Multi-Tire Hashing Frequent Termsets algorithm (MTHFT) in figure (4) to generate all strong association rules. It is basically different from all the previous algorithms since it overcomes the drawbacks of Apriori algorithm by employing the power of data structure called Multi-Tire Hash Table. Moreover it uses new methodology for generating frequent termsets by building the hash table during the scanning of documents only one time consequently, the number of scanning on documents decreased.

Once the frequent termsets from documents have been generated, it is straightforward to generate all strong association rules from them (where strong association rules satisfy both minimum support and minimum confidence). This can be done using the following equation for confidence, where the conditional probability is expressed in terms of termsets support count [40] :

$$Confidence\ (A \rightarrow B) = p(B \backslash A) = \frac{support_{count}(A \cup B)}{support_{count}(A)} \qquad (2)$$

where $port_{count}\ (A \cup B)$ is the number of documents containing the termsets $(A \cup B)$, and $support_{count}\ (A)$ is the number of documents containing the termset A.

1) The advantages of MTHFT Algorithm: The MTHFT algorithm has many advantages summarized as follows:

- Provides facilities to avoid unnecessary scans to the documents, which minimize the I/O. Where the scanning process occurs on the hash table instead of whole documents compared to Apriori algorithm
- The easy manipulations on hash data structure and directly computing frequent termsets are the added advantages of this algorithm, moreover the fast access and search of data with efficiency.
- MTHFT shows better performance in terms of time taken to generate frequent termsets when compared to Apriori algorithm. Furthermore, it permits the end user to change the threshold support and confidence factor without re-scanning the original documents since the algorithm saves the hash table into secondary storage media.
- The main advantage of this algorithm is that, it is scalable with all types of documents regardless of their sizes.
- Depending on the multi-tire technique in building the primary bucket, each bucket can store only a single element then we cannot associate more than one term with a single bucket, which is a problem in the case of collisions.

MTHFT Algorithm:
T_m: Set of all termsets for each document d
C_m: Candidate termsets for each document d
I_k : Frequent termsets of size k.
AR_k: Association Rules of size k

Input: All Text documents.
Process logic: Building Multi-Tire Hash Table and Finding the frequent termsets.
Output: Generating all strong Association Rules.

```
for each document $d_m \in D$ do begin
    $T_m = \{ t_i : t_i \in d_m ,\ 1 \le i \le n \}$
        for each term $t_i \in T_m$ do
            $h(t_i) = t_i \bmod N;$
            $t_i.count{+}{+};$
                // insert each term in hash table
        end
        $C_k$ = all combinations of $t_i \in d_m$
        $C_m = subset(C_k , d_m);$
            for each candidate $c_j \in C_m$ do
                $h(c_j) = c_j \bmod N;$
                $c_j.count{+}{+};$
            // insert each candidate in hash table
            end
end
  for given s= minsup in hash table do
      $I_1 = \{t \mid t.count \ge minsup \}$
      $I_k = \{c \mid c.count \ge minsup, k \ge 2\}$
  end
      for given c= minconf in $I_k$ do
          $AR_k = \{ I_i \rightarrow I_j \mid confidence \ge minconf, k \ge 2\}$
   end
```

Fig. 3. The MTHFT algorithm.

D. Documents Clustering Stage

Document clustering algorithm based on association rules considered a keyword-based algorithm which picks up the core rules between words with specific criteria and groups the documents based on these keywords. This approach includes five main steps:

- Picking out all strong Association Rules
- Constructing initial partitions

- Merging Similar Partitions
- Making Partition Disjoint
- Clustering Documents

1) Picking out all Strong Association Rules: The Multi-Tire Hashing Frequent Termsets algorithm is used in the previous step to find out all strong association rules furthermore to speeding up the mining process. It have ability to determine large frequent termsets at different minimum support threshold values without redoing the mining process again. Therefore, we can generating different sets of association rules between different frequent termsets in the clustering process easily. We start with a set of association rules R_s generated between the set of 2-large frequent termsets s since $R_k = I_i \rightarrow I_j$.

$$R_s = \{ R_1, R_2, R_3, \ldots\ldots\ldots\ldots\ldots\ldots, R_k\} \quad (3)$$

2) Constructing Initial Partitions: initially, we sort the set of all strong association rules R_s in descending order in accordance with their confidence level as in (4):

$$Conf(R_1) > Conf(R_2) > \ldots\ldots\ldots\ldots\ldots\ldots Conf(R_k) \quad (4)$$

An initial partition P_1 is constructed for first association rule in R_s. Afterward, all the documents containing both termsets that constructed the rules are included in the same cluster. Next, we take the second association rules whose confidence is less than the previous one to form a new partition P_2. This partition is formed by the same way of the partition P_1. This procedure is repeated until every association rules moved into partition P_i since

$$P_i = < R_i\ ,\ doc\ [\ R_i] > \quad (5)$$

Since a document usually contains more than one frequent termset, the same document may appear in multiple initial partitions, i.e., initial partitions are overlapping. The purpose of initial partitions is to ensure the property that all the documents in a cluster contain all the terms in the association rules that defines the partition. These rules can be considered as the mandatory identifiers for every document in the partition. We use these association rules as the partition label to identify the partition . The main purpose of presenting the partition label is to facilitate browsing for the user.

3) Merging Similar Partitions: in this step, all partitions that contain the similar documents are merged into one partition. The benefit of this step is reducing the number of resulted partitions.

4) Making partitions Disjoint: in this step, we remove the overlapping of partitions since there are some documents belong to one or more initial partitions. we assign a document to the "Optimal" initial partition so that each document belongs to exactly one partition. This step also guarantees that every document in the partition still contains the mandatory identifiers. We propose the Weighted Score ($P_i \leftarrow doc_j$) in equation (6) to measure the optimal initial partition P_i for a document doc_j.

$$\left(\boldsymbol{P_i \leftarrow doc_{\,j}}\right) = \boldsymbol{\textstyle\sum_k w_k * m_i/n_w} \quad (\boldsymbol{6})$$

where $\sum_k w_k$ represents the sum of weighted values of all words constructed the association rules from $doc_{\,j}$, m_i represents the number of documents in the initial partition P_i, and n_w represents the number of words that construct the partition P_i from $doc_{\,j}$. The weighted values of words w_k are defined by the standard inverse document frequency (TF-IDF) in the indexing process in section (III.B). The Weighted Score measure used the weighed values of frequent termsets instead of the number of occurrences of the terms in a document. Since the weighted values are an important piece of information based on the intuitive presumption of the weighting schema that is: the more often a term occurs in a document, the more representative of the content of the document (term frequency). Moreover the more documents the term occurs in, the less discriminating it is (inverse document frequency). To make partitions non-overlapping, we assign each doc_j to the initial partition P_i of the highest $score_i$. After this assignment, if there are more than one P_i that maximizes the Weighted Score $\left(P_i \leftarrow doc_{\,j}\right)$, we will choose the one that has the most number of words in the partition label. After this step, each document belongs to exactly one partition.

Example: Consider we have eleven documents to do clustering process. They are manually selected from different four topics (*Economy, Computer Science, Sports*, and *Avain Bird Flue*). Each document is indexed by a set of weighted words. After the mining process, we generated a set of strong association rules from 2-large frequent termsets equalls to 226 rule with 50% minimum confidence. The initial partitions of this example are constructed equals to 131 partition. After merging partitions based on the the similar documents we have 15 partition as shown in Table 1.

From the table, we observed that there are more than one document belongs to more than one partition *for example*, D_7 belongs to (P_1, P_3 and P_{15}) and D_5 belongs to (P_{10} , and P_{11}) and so on. To remove the overlapping between partitions and find the optimal partition for a document $doc_{\,j}$, we need to calculate its scores against each initial partition that contains the document as follows: to find the optimal partition for D_7 so that we begin to calcuate its scors against each initial partition (P_1 , P_3 and P_{15})

Weighted Score $(P_1 \leftarrow D_7)$

$$= (2.45+1.87+2.45+4.91+2.45+4.91) * 2 / 6$$

$$\boxed{= 6.34}$$

Weighted Score $(P_3 \leftarrow D_7)$

$$= (2.45+2.45+2.45+4.91+2.45+4.91) * 1 / 6$$

$$= 3.27$$

Weighted Score $(P_{15} \leftarrow D_7)$

$$= (2.45+1.87) * 2 / 2 = 4.32$$

TABLE I. INITIAL PARTITIONS

Initial Partitions	Text Documents
P_1	D_8,**D_7**
P_2	D_9,D_{10},D_{11}
P_3	**D_7**
P_4	D_9,D_{11}
P_5	D_8
P_6	D_1
P_7	D_1,D_2
P_8	D_6
P_9	D_4
P_{10}	D_5
P_{11}	D_4,D_5
P_{12}	D_6,D_8
P_{13}	D_1,D_2,D_3
P_{14}	D_1,D_3
P_{15}	D_6,**D_7**

From the above calculation, D_7 will assign to P_1 which has the highest score. After repeating the above computation for each document, each document belongs to exactly one partition as shown in Table 2.

5) *Clustering Documents:* after removing the overlapping and put each document in its optimal partition, we begin to clustering documents based on the partition labels. In this step, we don't require to pre-specified number of clusters as previous standard clustering algorithms. we have a set of non-overlapping partitions P_i and each partition has a number of documents D_j. We first identify the association rules that construcr each partition. The set of all words that construct all association rule in P_i called the labeling Words Ld $[W_i]$. Moreover every document in the partition must contain all the words in the partition label. We use the partition label to identify the partition.

TABLE II. DISJOINT PARTITIONS

Initial Partitions	Text Documents
P_1	D_8,D_7
P_2	D_9,D_{10},D_{11}
P_3	D_4,D_5
P_4	D_6
P_5	D_1,D_2,D_3

We observed that the partition labeling words based on association rules are more informative than other based on frequent termsets in [28]. However the number of association rules always greater than the number of frequent termsets, the rules carry out more information and identify hidden knowledge from documents help us to improve the accuracy of the clustering process.

The definition of the similarity measure plays an important role in obtaining effective and meaningful clusters. For each document D_j in partition P_i, to compute its similarity measure we must obtain the Derived keywords Vd $[W_i]$ from taking into account the difference words between the top weighted frequent words for each document with the labeling words. Subsequently the total support of each derived word is computed within the partition. The set of words satisfying the partition threshold (the percentage of the documents in partition P_i that contains the termset) are formed as Descriptive Words Pw $[C_i]$ of the partition P_i. Afterward, we compute the similarity of each document in the partitions with respect to the descriptive words. The similarity between two documents S_m is computed as in [41]. Based on the similarity measure, a new cluster is formed from the partitions i.e. each cluster will contain all partitions that have the similar similarity measures.

E. *Post processing*

For different applications there are different ways to do post processing. One common post processing is to select a suitable threshold to generate the final cluster result. After document clustering we get a basic cluster map in which the clusters are organized like a tree or in a flat way. Thereby some post processing algorithms may be applied to find out the correct clusters relation.

IV. EXPERIMENTAL RESULTS AND PERFORMANCE EVALUATION

Our experiments have been performed on a personal computer with a 2.50 GHz CPU and 6.00 GB RAM and we chose the programming language C#.net for the implementation because it allows fast and flexible development. The largest dataset, Reuters, is chosen to exam the efficiency and scalability of the ARWDC approach. To evaluate the effectiveness of the ARWDC approach, this section presents the result comparisons with some of the popular hierarchical document clustering algorithms like Bisecting K-means and FIHC for clustering web documents. The rest of this section first explains the evaluation measures, and finally presents and analyzes the experiment results.

A. *Evaluation Methods*

The F-measure, as the commonly used external measurement, is used to evaluate the accuracy of our clustering algorithms. F-measure is an aggregation of Precision and Recall concept of information retrieval. Recall is the ratio of the number of relevant documents retrieved for a query to the total number of relevant documents in the entire collection as in (7):

$$Recall\,(K_i\,,C_j) = \frac{n_{ij}}{|K_i|} \quad (7)$$

Precision is the ratio of the number of relevant documents to the total number of documents retrieved for a query as in (8):

$$Precision\,(K_i\,,C_j) = \frac{n_{ij}}{|C_j|} \quad (8)$$

while F-measure for cluster C_j and class K_i is calculated as in (9):

$$F\,(K_i\,,C_j) = \frac{2 * Recall\,(K_i\,,C_j) * Precision\,(K_i\,,C_j)}{Recall\,(K_i\,,C_j) + Precision\,(K_i\,,C_j)} \quad (9)$$

where n_{ij} is the number of members of class K_i in cluster C_j. $|C_j|$ is the number of members of cluster $C_j$and $|K_i|$ is the number of members of class K_i .

The weighted sum of all maximum F-measures for all natural classes is used to measure the quality of a clustering result *C*. This measure is called the *overall F-measure* of *C*, denoted $F(C)$ is calculated as in (10):

$$F(C) = \sum_{K_i \in K} \frac{|K_i|}{|D|} max_{C_j \in C}\{F(K_i, C_j)\} \qquad (10)$$

where *K* denotes all natural classes; *C* denotes all clusters at all levels; $|K_i|$ denotes the number of documents in natural class K_i; and $|D|$ denotes the total number of documents in the dataset. The range of $F(C)$ is [0,1]. A large $F(C)$ value indicates a higher accuracy of clustering.

B. *Experimental Results*

In this section, we evaluate the performance of the ARWDC approach in terms of the efficiency, accuracy and scalability compared to Bisecting K-means and FIHC algorithms. We chose Bisecting k-means because it has been reported to produce a better clustering result consistently compared to k-means and agglomerative hierarchical clustering algorithms. FIHC is also chosen because it uses frequent word sets. For a fair comparison, we did not implement Bisecting k-means and FIHC algorithms by ourselves. We downloaded the CLUTO toolkit [42] to perform Bisecting k-means, and obtained FIHC [43] from their author.

- ***Performance Investigations on Accuracy***

The F-measure represents the clustering accuracy. Table 3 shows the F-measure values for all three algorithms with different user specified numbers of clusters. Since ARWDC and HFTC do not take the number of clusters as an input parameter, we use the same minimum support 15% in Reuters dataset to ensure fair comparison.

From table (3), The highlighted results show that our ARWDC approach is better than Bisecting k-means and FIHC algorithms for specified Reuters data set. Furthermore the final average results indicate that the ARWDC outperforms all other algorithms in accuracy for most number of clusters.

Fig. 4 shows the comparison between all the three clustering approaches based on the overall F-measure values with different numbers of clusters. It illustrates that the ARWDC has the higher F-measure values than all competitive algorithms because it uses a better model for text documents. Higher F-measure shows the higher accuracy.

- ***Performance Investigations on Efficiency and Scalability***

The largest dataset, Reuters, is chosen to exam the efficiency and scalability of our approach. Many experiments were conducted to exam the efficiency of ARWDC approach.

TABLE III. F-MEASURE COMPARISON OF CLUSTERING ALGORITHMS

Datasets	# of	Overall F-measure		
	Clusters	Bisecting k-means	FIHC	ARWDC
Reuters 21578	3	0.34	0.53	**0.57**
	15	0.38	0.45	**0.56**
	30	0.38	0.43	**0.53**
	60	0.27	0.38	**0.59**
	average	0.41	0.44	**0.55**

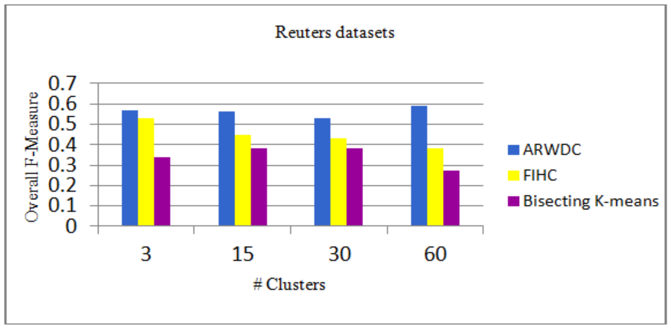

Fig. 4. Overall F-measure results comparison with Reuters dataset.

Figure 5 compares the runtime of ARWDC with bisecting k-means and FIHC algorithms on different sizes of documents of Reuters. The minimum support is set to 15% to ensure that the accuracy of all produced clustering are approximately the same. The number of documents is taken as X-axis and the time taken to find the clusters is taken as Y-axis. ARWDC approach runs approximately twice faster than the others. This is returned to the effect of using MTHFT algorithm for mining association rules. Since the execution time is decreased to mine association rules as support decreased in compared to Apriori algorithm. We conclude that ARWDC is more efficient than other approaches.

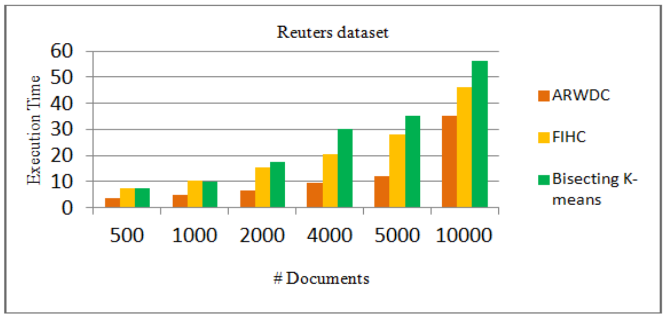

Fig. 5. Efficiency comparison of ARWDC with FIHC and Bisecting *K*-means on different sizes of Reuters at minsup=15%.

A large dataset from Reuters are created for examining the scalability of ARWDC approach. We duplicated the files in Reuters until we get 20000 documents. Figure 6 illustrates that our algorithm runs approximately twice faster than bisecting k-means and FIHC in this scaled up document set.

Figure 7 and 8 illustrate the runtimes with respect to the number of documents for different stages of AREDC approach and FIHC algorithm. Figure 7 shows that the MTHFT and clustering are not time-consuming stages since MTHFT algorithm improved the mining process and speed up the clustering stage. It demonstrates that ARWDC is a very scalable method.

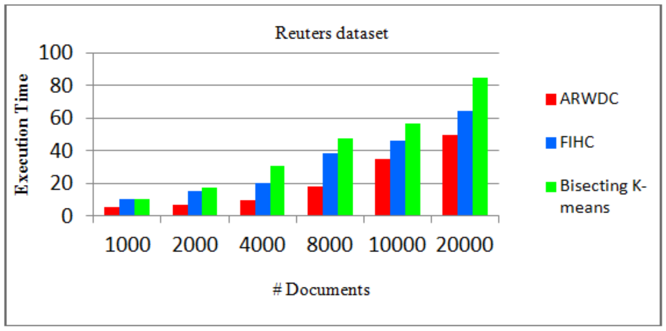

Fig. 6. Scalability comparison of ARWDC, FIHC and Bisecting K-means with scale up document set.

Figure 8 also shows that the Apriori and the clustering are the most time-consuming stages in FIHC, while the runtimes of MTHFT and clustering stages are comparatively short. Since the efficiency of the Apriori is very sensitive to the input parameter minimum support. Consequently, the runtime of FIHC is inversely related to this parameter. In other words, runtime increases as minimum support decreases.

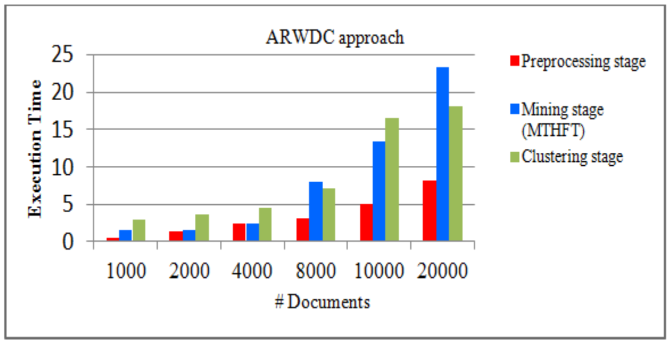

Fig. 7. Scalability comparison of ARWDC approach on different sizes of Reuters for all different stages.

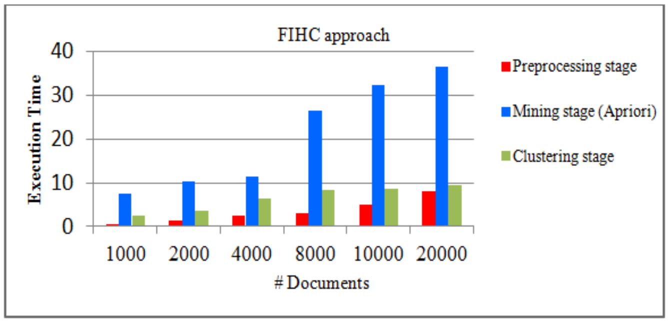

Fig. 8. Scalability comparison of FIHC algorithm on different sizes of Reuters for all different stages.

In conclusion, the major advantages of our ARWDC approach are as follows:

- By generating the strong association rules with specific criteria , the dimensionality of a document is drastically reduced. This is a key factor for the efficiency and scalability of ARWDC approach.
- Experimental results show that ARWDC outperforms the well-known clustering algorithms in terms of accuracy. It is robust and consistent even when it is applied to large and complicated document sets.
- Many existing clustering algorithms require the user to specify the desired number of clusters as an input parameter. ARWDC treats it only as an optional input parameter. Close to optimal clustering quality can be achieved even when this value is unknown.
- Easy to browse with more informative and meaningful partition labels since each partition has a set of association rules which a user may utilize for browsing.
- Since a real world document set may contain a few hundred thousand of documents, experiments show that our approach is significantly more efficient and scalable than all of the tested competitors.

II. CONCLUSION

In this paper, we have conducted an extensive analysis of association rules-based web document clustering ARWDC approach. The largest dataset, Reuters, is chosen to exam the efficiency and scalability of our algorithm. The experimental results show that at different sizes of Reuters datasets, the ARWDC approach improved scalability. Furthermore when compared with other clustering algorithms like Bisecting K-means and FIHC, the accuracy and efficiency are improved. Moreover, ARWDC approach associated a meaningful label to each final cluster. Then the user can easily find out what the cluster is about since the label can provide an adequate description of the cluster based on Association Rules. However, it is time-consuming to determine the labels after the clustering process is finished. From all experiments, we conclude that ARWDC approach has favorable quality in clustering documents using Association Rules.

III. FUTURE WORK

The importance of document clustering will continue to grow along with the massive volumes of web documents. With the standardization of XML as an information exchange language over the web, documents formatted in XML have become quite popular. Moreover, most of the clustering algorithms of MEDLINE abstracts are based on pre-defined categories. In future, we intend to apply ARWDC approach for automatically clustering the MEDLINE abstracts formatted in XML to help biomedical researchers in quickly finding relevant and important articles related to their research field without need to predefine categories.

References

[1] N. Andrews, and E. Fox, “Recent developments in document clustering,” Technical Report TR-07-35, Computer Science, Virginia Tech. April 2007.

[2] S. Sharma, and V. Gupta, “Recent developments in text clustering techniques,” in Proc. of Int. J. of Computer Applications, vol. 37, pp. 14-19, 2012.

[3] K. Jain, N. Murty, and J. Flynn, “Data clustering: a review,” in Proc. of Int. Conf. on ACM Computing Surveys, vol. 31, pp. 264-323, 1999.

[4] M. Steinbach, G. Karypis, and V. Kumar, “A comparison of document clustering techniques,” KDD Workshop on Text Mining, 2000. Avaiable online : http://glaros.dtc.umn.edu/gkhome/node/157

[5] P. Berkhin, (2004)“Survey of clustering data mining techniques,” [Online]. Available: http://www.accrue.com/products/rp_cluster_review pdf

[6] R. Xu, “Survey of clustering algorithms,” in Proc. of Int. Conf. of IEEE Transactions on Neural Networks, vol. 15, pp. 634-678, 2005.

[7] F. Benjamin, W. Ke, and E. Martin, “Hierarchical document clustering,” Simon Fraser University, Canada, pp. 555-559, 2005.

[8] J. Ashish, and J. Nitin, “Hierarchical document clustering: A review,” in Proc. of 2nd National. Conf. on Information and Communication Technology, 2011, Proceedings published in Int. J. of Computer Applications.

[9] R. Agrawal, T. Imielinski, and A. Swami, “Mining association rules between sets of items in large databases,” in Proc. of Int. Conf. on Management of Data, vol. 22, pp. 207-216, Washington 1993.

[10] O. Zaiane and M. Antonie, “Classifying text documents by association terms with text categories,” in Proc. of Int. Conf. of Australasian Database, vol. 24, pp. 215-222, 2002.

[11] B. Liu, W. Hsu and Y. Ma, “Integrating classification and association rule mining,” in Proc. of Int. Conf. of ACM SIGKDD on Knowledge Discovery and Data Mining, pp. 80-86, 1998.

[12] M. Beil, and X. Xu, “Frequent term-based text clustering,” in Proc. of Int. Conf. on Knowledge Discovery and Data Mining, pp. 436- 442, 2002.

[13] O. Zamir and O. Etzioni, “Web document clustering: A feasability demonstration,” in Proc. of Int. Conf. of ACM SIGIR, pp. 46-54, 1998.

[14] M. Hassan and K. John, “High quality, efficient hierarchical document clustering using closed interesting itemsets,” in Proc. of Int. IEEE Conf. of on Data Mining, pp. 991-996, 2006.

[15] L. Xiangwei, H. Pilian, A study on text clustering algorithms based on frequent term sets, Springer-Verlag Berlin Heidelberg, 2005.

[16] Y.J. Li, S.M. Chung, J.D. Holt, “Text document clustering based on frequent word meaning sequences,” in Proc. of Int. Conf. of Data & Knowledge Engineering, vol. 64, pp. 381–404, 2008.

[17] H. Edith, A.G. Rene, J.A. Carrasco-Ochoa, and J.F. Martinez-Trinidad, “Document clustering based on maximal frequent sequence,” in Proc. of Int. Conf. of FinTal, vol. 4139, pp. 257-267, LNAI 2006.

[18] Z. Chong, L. Yansheng, Z. Lei and H. Rong, FICW: Frequent itemset based text clustering with window constraint, Wuhan University journal of natural sciences, vol. 11, pp. 1345-1351, 2006.

[19] L. Wang, L. Tian, Y. Jia and W. Han, “A Hybrid algorithm for web document clustering based on frequent term sets and k-means,” Lecture Notes in Computer Science, Springer Berlin, vol. 4537, pp. 198-203, 2010.

[20] Z. Su, W. Song, M. Lin, and J. Li, “Web text clustering for personalized e-Learning based on maximal frequent itemsets,” in Proc. of Int. Conf. on Computer Science and Software Engineering, vol. 06, pp. 452-455, 2008.

[21] Y. Wang, Y. Jia and S. Yang, “Short documents clustering in very large text databases,” Lecture Notes in Computer Science, Springer Berlin, vol. 4256, pp. 38-93, 2006.

[22] W. Liu and X. Zheng, “Documents clustering based on frequent term sets,” in Proc. of Int. Conf. on Intelligent Systems and Control, 2005.

[23] H. Anaya, A. Pons and R. Berlanga, “A Document clustering algorithm for discovering and describing topics,” Pattern Recognition Letters, vol. 31, pp. 502-510, April 2010.

[24] R. Kiran, S. Ravi, and p. Vikram, “Frequent itemset based hierarchical document clustering ung Wikipedia as external knowledge,” Springer-Verlag Berlin Heidelberg, 2010.

[25] B. Fung, K. Wang, and M. Ester, “Hierarchical document clustering using frequent itemsets,” in Proc. of Int. Conf. on Data Mining, vol. 30, pp. 59-70, 2003.

[26] R. Baghel and Dr. R. Dhir, A Frequent concept based document clustering algorithm, in Proc. of Int. J. of Computer Applications, vol. 4, pp. 0975 – 8887, 2010.

[27] N. Negm, P. Elkafrawy, and A. Salem, “An Efficient hash-based association rule mining approach for document clustering,” in Proc. of Int. 16th WSEAS Conf. on COMPUTERS, July 14-17, pp. 376-381, 2012, Kos Island, Greece.

[28] N. Negm, P. Elkafrawy, M. Amin, and A. Salem, “Clustering web documents based on efficient multi-tire hashing algorithm for mining frequent termsets,” in Proc. of Int. J. of Advanced Computer Science and Applications, vol. 2, pp. 6-14, 2013.

[29] R. Agrawal and R. Srikant, “Fast algorithms for mining association rules,” in Proc. of Int. Conf. of Very Large Data Bases, VLDB, pp. 487–499. Morgan Kaufmann, 12–15, 1994.

[30] E. Han, G. Karypis, V. Kumar, and B. Mobasher, “Clustering based on association rule hypergraphs,” in Proc. of SIGMOD Workshop Research Issues on Data Mining and Knowledge Discovery(DMKD ’97), 1997.

[31] D. Gunopulos, H. Mannila, R. Khardon, and H. Toivonen, “Data mining, hypergraph transversals, and machine learning (extended abstract),” in Proc. of Int. Conf. on PODS, pp. 209–216, 1997.

[32] G. Italiano, G. Ausiello, and U. Nanni, “Dynamic maintenance of directed hypergraphs,” in Proc. of Int. Conf. on Theoretical Computer Science, 72(2-3), pp.97–117, 1990.

[33] G. Gallo, G. Longo, and S. Pallottino, “Directed hypergraphs and applications,” in Proc. of Int. Conf. on Discrete Applied Mathematics, 42(2), pp.177–201, 1993.

[34] E.Han, G. Karypis, V. Kumar, and B. Mobasher, “ Hypergraph based clustering in high dimensional data sets :A Summary of Results,” Copyright 1997, IEEE.

[35] M. Jabbar, P. Chandra, and B. Deekshatulu, Cluster based association rule mining for heart attack prediction, in Proc. of J. of Theoretical and Applied Information Technology, 31st October 2011, vol. 32, no. 2, pp.196-201, 2011.

[36] http://kdd.ics.uci.edu/databases/reuters21578/reuters21578.html

[37] J. Jayabharathy, S. Kanmani, and A. Ayeshaa Parveen, A survey od document clustering algorithm with topic discovery, in Proc. of J. of Computing, February 2011, vol. 3, no. 2.

[38] http://tartarus.org/martin/PorterStemmer/

[39] M. Berry, Survey of text mining: clustering, classification, and retrieval,” Springer-Verlag New York, Inc., 2004.

[40] B. Liu, Web Data Mining. Exploring Hyperlinks, Contents, and Usage Data, 2nd ed, Springer-Verlag Berlin Heidelberg, 2011.

[41] S. Krishna, and S. Bhavani, An ffficient approach for text clustering based on frequent itemsets, in Proc. of European J. of Scientific Research, vol.42, no.3, pp.399-410, 2010.

[42] http://glaros.dtc.umn.edu/gkhome/views/cluto

[43] http://ddm.cs.sfu.ca/dmsoft/Clustering/fihc_index.html