Preceding Document Clustering by Graph Mining Based Maximal Frequent Termsets Preservation

Syed Shah and Mohammad Amjad
Department of Computer Engineering, Jamia Millia Islamia, India

Abstract: *This paper presents an approach to cluster documents. It introduces a novel graph mining based algorithm to find frequent termsets present in a document set. The document set is initially mapped onto a bipartite graph. Based on the results of our algorithm, the document set is modified to reduce its dimensionality. Then, Bisecting K-means algorithm is executed over the modified document set to obtain a set of very meaningful clusters. It has been shown that the proposed approach, Clustering preceded by Graph Mining based Maximal Frequent Termsets Preservation (CGFTP), produces better quality clusters than produced by some classical document clustering algorithm(s). It has also been shown that the produced clusters are easily interpretable. The quality of clusters has been measured in terms of their F-measure.*

Keywords: *Bipartite graph, graph mining, frequent termsets mining, bisecting K-means.*

Received June 18, 2016; accepted June 29, 2017

1. Introduction

Document clustering (or text clustering) is the application of cluster analysis to textual documents. It has applications in text mining, automatic document organization, topic extraction and fast information retrieval or filtering. The applications may be online or offline. Originally document clustering was used to improve the precision in information retrieval systems [10, 16] and for finding the nearest neighbors of a document [2]. Later it was found to be useful in browsing text documents (e.g., news articles) [19] and also for organizing the results of a web user's query onto a search engine [22]. It has also been used to generate hierarchical clusters of documents [9].

Two well known document clustering techniques are K-means and agglomerative hierarchical clustering. K-means is faster than agglomerative hierarchical clustering (hierarchical clustering has a quadratic time complexity in contrast to a variant of K-means that has a linear time complexity) [18]. But both these algorithms do not address the fundamental problem of document clustering that of very high dimensionality.

Later a new approach was introduced where a clustering algorithm was not applied to the input document corpus but instead to its refined form that constitutes not all terms but only those that are frequent. Thus two fields of data mining-frequent termsets mining and clustering - were treated as two phases of document clustering, one after another. Based on this new approach many algorithms were proposed (see section 1.1.).

1.1. Related Work

Morzy *et al*. [15] introduces a hierarchical clustering algorithm that uses sequential patterns found in the database to generate both the clustering model and data clusters, [13] in one approach Clustering based on Frequent Word Sequences (CFWS) takes into consideration the sequence of frequent words where {cricket, bat} and {bat, cricket} are treated as two different patterns and in another approach Clustering based on Frequent Word Meaning Sequences (CFWMS) takes into consideration frequent word meaning sequences where not only sequence but also the contextual meaning of each word has been considered, [11] applies frequent-itemset based clustering to web search results, [6] proposes document clustering based on maximal frequent sequences where only maximal word sequences have been considered, [1] starts with an empty set, it continues selecting one more element (one cluster description) from the set of remaining frequent itemsets until the entire document collection is contained in the cover of the set of all chosen frequent itemsets, [4] takes into consideration both global frequent items and cluster frequent items, [23] finds frequent itemsets and then uses minimum spanning tree algorithm to construct clusters, [21] groups web transactions using a hierarchical pattern-based clustering approach, [12] uses Apriori for finding frequent itemsets and then uses the mined frequent itemsets to obtain partitions (with no overlapping) and after that groups documents within a partition using derived keywords, some other researches propose use of Wikipedia as external knowledge source thus taking into consideration the semantic relationships between words [8], for dynamic data [17] proposed evolutionary clustering that was again based on frequent itemsets, to decrease the number of patterns and time complexity [14] proposed a pattern-based hierarchical document clustering that mines for local patterns and builds a cluster hierarchy

without mining for globally significant patterns, [5] performs pattern-based clustering on numerical datasets using unsupervised decision trees (it extracts, from a collection of trees generated through a new induction procedure, a small subset of patterns useful for clustering), to improve accuracy of clustering [3] proposes usage of fuzzy association rule mining algorithm, [20] uses a measure h-confidence (minimum of all confidence values in an itemset) to consider only those frequent itemsets that pass a certain h-confidence threshold and [4] applies hierarchical clustering to global frequent itemsets thus constructing a cluster tree followed by child pruning and sibling merging,. Algorithms that take into consideration word meanings or the word sequences or propose use of external knowledge source(s) for finding out semantic relationships between words produce slightly better quality clusters but the trade-off between quality and time is too much. The evolutionary approaches suffer from creating low quality clusters but since the input data stream is dynamic this problem seems unavoidable. Algorithms that mine for local patterns instead of global do save time but produce low quality clusters.

This paper proposes an algorithm for document clustering on the same lines (performing frequent termset mining followed by clustering) on a document corpus. A novel algorithm has been introduced to perform frequent termsets mining. The results of this phase are then provided as an input to the most time efficient clustering algorithm-Bisecting K-means.

The rest of this paper is organized as follows. Section 2 briefly defines some of the terms related to our work. In section 3, we introduce our novel algorithm for frequent termsets mining. In section 4, clustering of the results of section 3 has been discussed. An experimental evaluation on real text data was conducted, and section 5 reports its major results. Section 6 summarizes the paper and outlines some interesting directions for future research.

2. Basic Preliminaries

We quickly review some standard definitions related to our work. Let D={d1, d2, d3,…} represent the set of documents called as document set or document corpus and let T={t1, t2, t3,…) represent the set of all the terms that occur in all the documents of D i.e., $t_i \in D \; \forall$ i. A termset TS $\subseteq$ T is a term or a set of terms and the support of a termset TS, denoted by supp (TS), is the fraction of documents containing TS. A termset whose support is equal to or greater than a user-specified minimum is a frequent termset i.e., if supp (TS) $\geq$ min_sup, then TS is a frequent termset.

We represent the document set D, the term set T and the relationships between them using a bipartite graph. A bipartite graph, also known as a bigraph, is a graph whose vertices can be divided into two disjoint independent sets U and V such that every edge connects a vertex in U to one in V. In our paper, D represents one such set and T another and the graph is denoted by G=(D, T, E) where E={e1, e2, e3,…} is the set of edges of G, each edge ek acknowledging presence of term ti in document dj (ti is represented by one type of vertex in in G and dj by another type of vertex).

3. Graph Mining Based Maximal Frequent Termsets Preservation

The most natural way to represent two sets that are interrelated to each other but the elements within a set are independent of each other is a bipartite graph. We stick to this nature of a document corpus. After performing the following preprocessing operations on our document corpus, we map the preprocessed corpus onto a bipartite graph (G_B):

1. Strip extra whitespaces.
2. Convert all alphabets to lower-case.
3. Remove all punctuation symbols (except intra-word dashes).
4. Remove all numbers.
5. Remove stop-words.
6. Perform stemming on each word of each document.

A summarized view of the various stages of our proposed approach is shown in Figure 1.

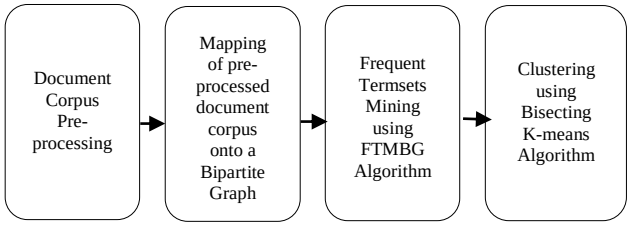

Figure 1. Stages of our proposed approach.

Next method shows the pseudocode of our algorithm-Frequent Termsets Mining using Bipartite Graphs (FTMBG). This algorithm operates on G_B and consists of three steps. In Step 1, FTMBG finds all frequent 1-termsets, in Step 2 it finds all frequent 2-termsets and in Step 3 it finds all frequent 3 or greater than 3 term-sets.[1]

Algorithm 1: FTMBG Algorithm

Input: G_B: Bipartite graph of preprocessed document set.
min_sup: minimum support threshold.
max_sup: maximum support threshold.
Output: FT: Frequent termsets.
Variables: nodocs: Number of type-1 (document) vertices of G_B.
noterms: Number of type-2 (term)

[1]The implementation has been done in R language and all the results were produced using this tool.

vertices of G_B left after Step I.
G_C, G_{temp}: Graph objects of temporary usage.
α: min_sup x nodocs.
β: max_sup x nodocs.

Method

Step 1: Frequent 1-termsets discovery
1. *Delete all type-2 vertices of G_B whose degree is below α or above β.*
2. *Preserve all remaining type-2 vertices' names.*
3. *Create a copy of G_B in G_C.*

Step 2: Frequent 2-termsets discovery
1. *for k in nodocs+1 to nodocs+noterms-1*
   find all the neighbors of k^{th} vertex.
   for j in k+1 to noterms
   $count_{k,j}$ = 0
   for i in 1 to degree of k^{th} vertex
   if an edge exists between j^{th} vertex of G_B and i_{th} neighbor of k, increment $count_{k,j}$
   endfor
   if $count_{k,j}$ >= α
   Preserve k^{th} and j^{th} vertices' names as one unit.
   Add this unit to G_B as a new type-2 vertex.
   Add edges between this new vertex and each of the type-1 vertices that have an edge with both k^{th} and j^{th} vertices.
   endif
   endfor
   endfor

Step3: Frequent 3 or greater than 3 termsets discovery
1. *Create a copy of G_C in G_{temp}.*
2. *for k in nodocs+1 to nodocs+noterms-1*
   find all the neighbors of k^{th} vertex.
   for j in noterms+k to VertexCount(G_B)
   $count_{k,j}$ = 0
   for i in 1 to degree of k^{th} vertex
   if an edge exists between j^{th} vertex of G_B and i_{th} neighbor of k, increment $count_{k,j}$
   endfor
   if $count_{k,j}$ >= α
   Preserve k^{th} and j^{th} vertices' names (in sorted order) as one unit (avoiding redundancy).
   Add this unit to G_{temp} as a new type-2 vertex.
   Add edges between this new vertex and each of the type-1 vertices that have an edge with both k^{th} and j^{th} vertices in G_B.

For an example, we consider a small text document set of three documents {d1, d2, d3}. Preprocess it and then map it onto a bipartite graph G_B (Figure 2). Term-vertex has been marked grey and document-vertex as orange. An edge between a term-vertex and adocument-vertex represents the presence of the term in the document.

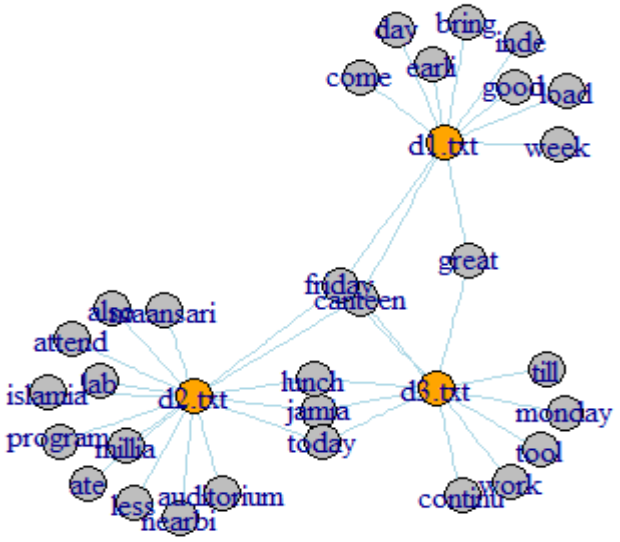

Figure 2. Bipartite graph of example dataset.

Step 1 of FTMBG deletes all those terms from the graph that do not cross the user-defined minimum support threshold. It also deletes those terms that fall above the maximum support threshold value. We used maximum support threshold to eliminate those terms that are present in too many documents because such terms are of no use to clustering. In our example, for sake of simplificity, we have kept minimum support threshold as 50% and maximum support threshold as 100%. Figure 3 shows the state of graph G_B after execution of Step 1 of FTMBG on our example graph. Each term-vertex (grey-vertex) represents a frequent 1-termset. In other words all the global frequent terms are represented here by the term-vertex set.

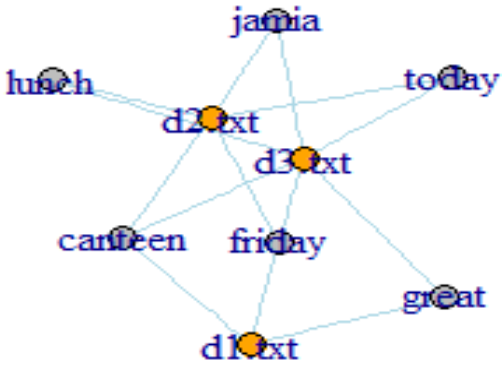

Figure 3. State of graph after execution of Step 1 of FTMBG.

In Step 2, each term-vertex of Figure 4 is checked for how many common neighbors does it have with another term-vertex and if the number crosses the minimum support threshold, both (as one unit) are preserved as a frequent 2-termset. This termset is also added to the graph. Figure 4 shows the state of graph after Step 2.

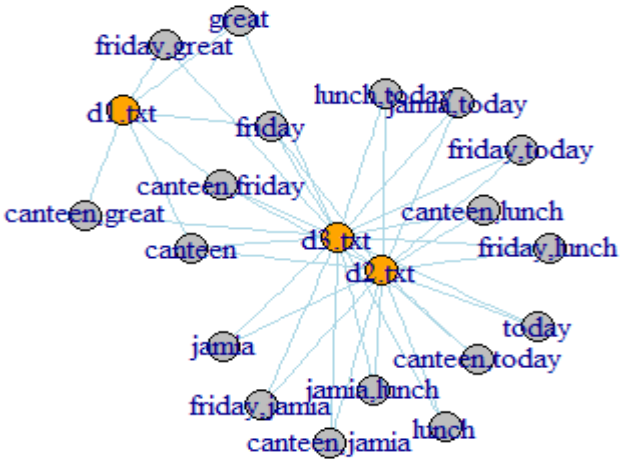

Figure 4. State of graph after Step 2.

In first iteration of Step 3, each term-vertex (with one-word name) is checked for how many common neighbors does it have with another term-vertex (with two-word name) (note: the index of each vertex, be it one-word name vertex or two-word name vertex is known to the algorithm) and if the number crosses the minimum support threshold, both (as a three-word name unit) are preserved as a frequent 3-termset. In second iteration of Step 3, each term-vertex (with one-word name) is checked for how many common neighbors does it have with another term-vertex (with three-word name) and if the number crosses the minimum support threshold, both (as a four-word name unit) are preserved as a frequent 4-termset; and the iterations continue until graph comes back to the state in which it was after Step 1 (i.e., state of Figure 3). The coming back to this state indicates that, now, no unfound frequent termsets exist. Figures 5, 6, 7, and 8 show the state after first, second, third and fourth iteration of Step 3, respectively.

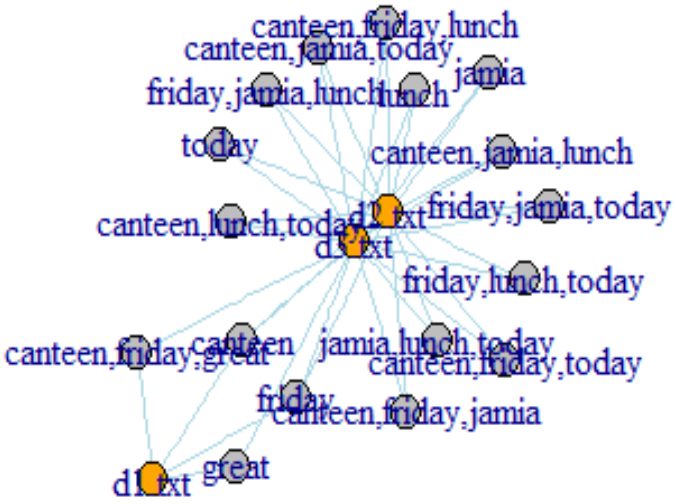

Figure 5. State of graph after first iteration of Step 3.

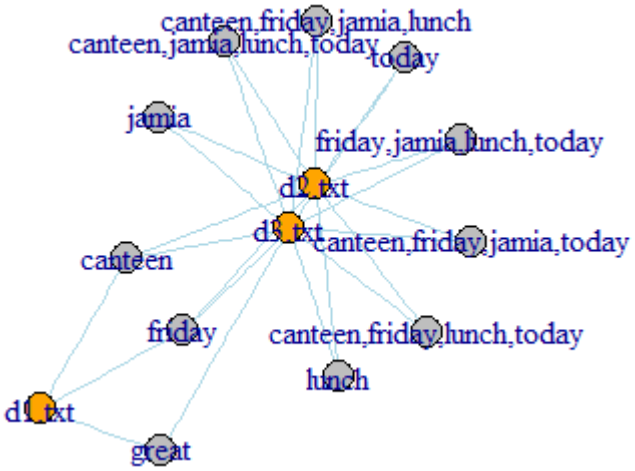

Figure 6. State of graph after second iteration of Step 3.

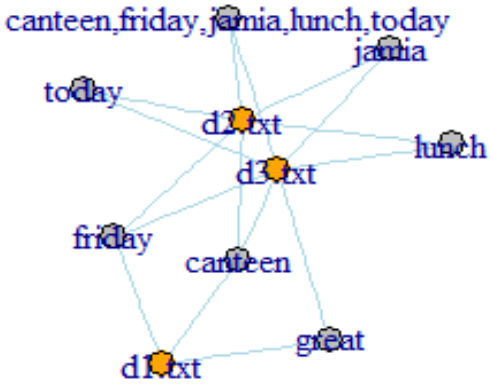

Figure 7. State of graph after third iteration of Step 3.

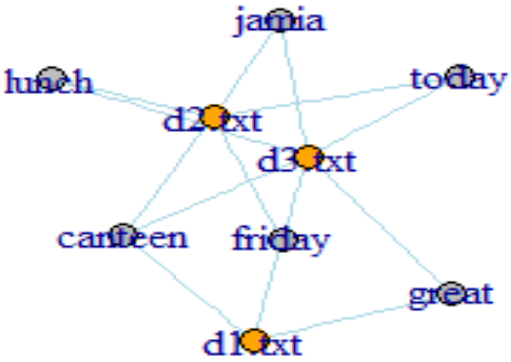

Figure 8. State of graph after fourth iteration of Step 3.

FTMBG algorithm discovers all frequent termsets in the document corpus. Before going on to the next phase we remove all those frequent termsets that have a frequent superset, thus keeping only maximal frequent termsets. Then, we represent each document in the document set by only these maximal frequent termsets, thus drastically reducing dimensionality. Those documents that fail to get a representation in terms of maximal frequent termsets are represented by maximal subsets of maximal frequent termsets (this is done in accordance with the monotonic property-that every subset of a frequent termset is also frequent). Figure 9 shows the bipartite graph representation of the modified document set of our example.

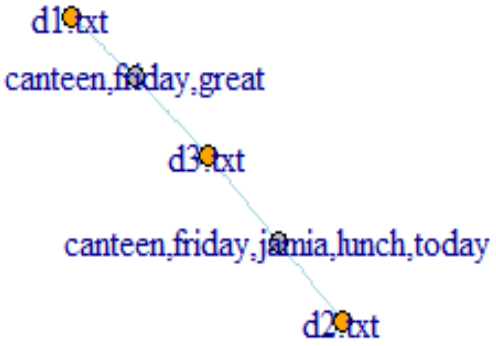

Figure 9. Bipartite graph representation of the modified document set.

This completes Phase 1 of our proposed approach.

4. Clustering of Modified Document Set

In Phase 2, clustering algorithm has been applied to the modified document set. Among the hierarchical clustering approaches UPGMA is the most appropriate for document clustering and among the partitional approaches a variant of K-means-Bisecting K-means has been found to be the best [18]. We use the latter-Bisecting K-means in Phase 2, the reason being its linear complexity.

Before applying Bisecting K-means, we perform the following operations on the modified document set:

1. Create a Document-Term Matrix (DTM) of the modified document set.
2. Assign a weight to each term (set) in the modified document set-the weight being the number of words in that term (set).
3. Construct a modified DTM (MDTM) for which we propose a novel mathematical Equation:

$$MDTM[i, j] = \left(\frac{w_j}{noterms}\right) * DTM[i, j] \qquad (1)$$

where $MDTM$ $[i, j]$ is the floating point value of the i^{th} row j^{th} column of the modified document term matrix, w_j is the weight of the j^{th} term(set) of the DTM, noterms is the total number of frequent 1-termsets and DTM $[i, j]$ is the integer value of the i^{th} *row* j^{th} column of the document term matrix.

For calculating the similarity between two documents we use cosine similarity measure:

$$sim(d_i, d_j) = \frac{x_i . y_j}{\| x_i \| \| y_j \|} \qquad (2)$$

Where x_i and y_j are vectors representing, respectively, documents d_i and d_j in the MDTM, $\|x_i\|$ is the Euclidean norm of vector $x_i=(x_{i1}, x_{i2},\ldots., x_{ir})$ defined as $(x_{i1}^2+x_{i2}^2+\ldots.+x_{ir}^2)^{1/2}$ and $\|y_i\|$ is the Euclidean norm of vector y_i.

The cosine similarity measure is not applied directly to the DTM of the modified document set but to a modified form of DTM (MDTM).

Table 1 shows the DTM of the modified document set (Figure 9) of our earlier example (Figure 2).

Table 1. Document term matrix.

	canteen,friday,great	canteen,friday,jamia,lunch,today
d1	1	0
d2	0	1
d3	1	1

It can be seen that FTMBG has reduced the dimensions of our document set from 30 to only 2.

Table 2 shows the MDTM for our example.

Table 2. Modified form of document term matrix.

	canteen,friday,great	canteen,friday,jamia,lunch,today
d1	0.5	0
d2	0	0.83
d3	0.5	0.83

After applying bisecting K-means (putting K=2) to this MDTM, we obtain clusters {d1} and {d2, d3}.

5. Experimental Evaluation

This section presents the experimental evaluation of our proposed method by comparing its results with popular document clustering algorithms. Three (widely used) datasets were used for the evaluation-Re0, Wap and Hitech (Table 3). These datasets are heterogeneous with regard to number of terms and document distribution. In these datasets each document has been pre-classified into a single topic, this eased the task of evaluation. Our clustering algorithm is but unaware of this classification. The data sets have been obtained from [7].

Table 3. Summary of evaluated datasets.

Dataset	# of Documents	# of Terms	# of Classes	Class Size
Re0	1504	2886	13	11-608
Wap	1560	8460	20	5-341
Hitech	2301	13170	6	116-603

Since F-measure has been found to be the most appropriate method for evaluating the quality of clusters, we have chosen it for the evaluation of our resultant clusters. Entropy has not been considered because of its inherent bias – it favors larger number of clusters. The F-measure combines the precision and recall concepts of information retrieval [16]. We calculate Recall and Precision of a cluster for each given class as follows:

$$\mathrm{Re}call(i, j) = \frac{n_{ij}}{n_i} \qquad (3)$$

$$\mathrm{Pr}ecision(i, j) = \frac{n_{ij}}{n_j} \qquad (4)$$

Where n_{ij} is the number of documents of class i in cluster j, n_i is the number of documents in class i and n_j is the number of documents in cluster j. The F-measure of cluster j and class i is given by the harmonic mean of Recall and Precision:

$$F(i,j) = \frac{2 * \mathrm{Re}call(i,j) * \mathrm{Pr}ecision(i,j)}{\mathrm{Re}call(i,j) + \mathrm{Pr}ecision(i,j)} \quad (5)$$

5.1. Results

The F-measure results are shown in Table 4. The minimum and maximum support threshold is 5% and 90%, respectively. For ease of comparison, we have also plotted the results (see Figures 10, 11, and 12). The table clearly shows that in many cases our algorithm produces better quality clusters than one or the other popular clustering algorithm. The table shows that our algorithm outperforms Bisecting K-means in terms of quality of clusters. For Wap dataset, our algorithm beats its counterpart HFTC [1] and for Hitech dataset it is almost as good as UPGMA.

Table 4. Comparison of F-Measure.

Dataset	# of Clusters	Bisecting K-means	UPGMA	HFTC	CGFTP
Re0	15	0.34	0.36	0.43	0.40
	30	0.38	0.47	0.43	0.39
	60	0.28	0.42	0.43	0.34
	Avg.	0.35	0.41	0.43	0.38
Wap	15	0.57	0.49	0.35	0.49
	30	0.44	0.58	0.35	0.49
	60	0.37	0.59	0.35	0.39
	Avg.	0.46	0.55	0.35	0.46
Hitech	15	0.44	0.33	0.37	0.39
	30	0.29	0.33	0.37	0.35
	60	0.21	0.47	0.37	0.31
	Avg.	0.31	0.40	0.37	0.35

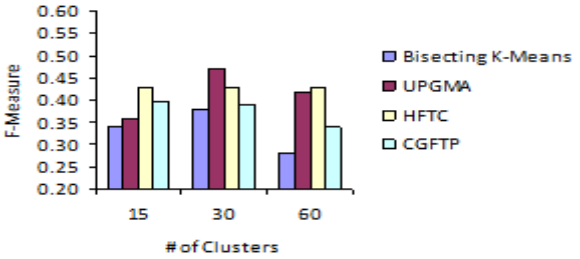

Figure 10. F-Measure comparison on Re0 dataset.

Figure 11. F-Measure comparison on Wap dataset.

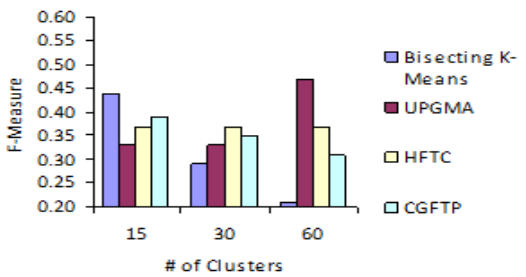

Figure 12. F-Measure comparison on hitech dataset.

Upon executing our algorithm on Re0 dataset for number of clusters=15, one of the produced clusters was of size 190. A cluster with 190 documents is very difficult to be interpreted but since we had kept only maximal frequent termsets in the documents, we could easily interpret such clusters. Just a walk through the cluster to find the within cluster frequent termsets gives a clear idea of what the cluster stands for.

6. Conclusions

In this paper, a novel approach to frequent termsets based document clustering was introduced. An algorithm was presented that executes on a bipartite graph. This bipartite graph is a mapping of the input document corpus.

Experimental evaluation on real document datasets demonstrated that in many cases our approach produced better quality clusters compared to one or the other popular document clustering approach. In addition to that, our algorithm also assists in interpreting the resultant clusters with ease.

Finally, we would like to outline a few directions for future research. Hierarchical clusterings are of importance to many applications, so we can follow our FTMBG algorithm by some hierarchical algorithm to cater needs of such applications. FTMBG can also be used for transaction data analysis, for market basket analysis etc., we plan to consider these applications in future.

References

[1] Beil F., Ester M., and Xu X., "Frequent Term-Based Text Clustering," *in Proceedings of 8th International Conference on Knowledge Discovery and Data Mining*, Alberta, pp. 436-442, 2002.

[2] Buckley C. and Lewit A., "Optimizations of Inverted Vector Searches," *in Proceedings of 8th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Montreal, pp. 97-110, 1985.

[3] Chen C., Tseng F., and Liang T., "Mining Fuzzy Frequent Itemsets for Hierarchical Document Clustering," *Information Processing and Management*, vol. 46, no. 2, pp. 193-211, 2010.

[4] Fung B., Wang K., and Ester M., "Hierarchical Document Clustering Using Frequent Itemsets," *in Proceedings of 3rd SIAM International Conference on Data Mining*, San Francisco, pp. 59-70, 2003.

[5] Gutierrez A., Martinez J., Garcia M., and Carrasco J., "Mining Patterns For Clustering on Numerical Datasets Using Unsupervised Decision Trees," *Knowledge-Based Systems*, vol. 82, pp. 70-79, 2015.

[6] Hernandez-Reyes E., Garcia-Hernández R., Carrasco-Ochoa J., and Martinez-Trinidad J., "Document Clustering Based on Maximal Frequent Sequences," *in Proceedings of 5th International Conference on Natural Language Processing*, Turku, pp. 257-267, 2006.

[7] Karypis G., Karypis Lab. http://glaros.dtc.umn.edu/gkhome/cluto/cluto/download, Last Visited, 2016.

[8] Kiran G., Shankar K., and Pudi V., "Frequent Itemset based Hierarchical Document Clustering using Wikipedia as External Knowledge," *in Proceedings of 14th International Conference on Knowledge-Based and Intelligent Information Engineering Systems*, Cardiff, pp. 11-20, 2010.

[9] Koller D. and Sahami M., "Hierarchically Classifying Documents Using Very Few Words," *in Proceedings of 14th International Conference on Machine Learning*, Nashville, pp. 170-178, 1997.

[10] Kowalski G., *Information Retrieval Systems-Theory and Implementation*, Kluwer Academic Publishers, 1997.

[11] Kozlowski M., "Web Search Results Clustering Using Frequent Termset Mining," *in Proceedings of 6th International Conference on Pattern Recognition and Machine Intelligence*, Warsaw, pp. 525-534, 2015.

[12] Krishna S. and Bhavani S., "An Efficient Approach for Text Clustering Based on Frequent Itemsets," *European Journal of Scientific Research*, vol. 42, no. 3, pp. 385-396, 2010.

[13] Li Y., Chung S., and Holt J., "Text Document Clustering Based on Frequent Word Meaning Sequences," *Data and Knowledge Engineering*, vol. 64, no. 1, pp. 381-404, 2008.

[14] Malik H., Kender J., Fradkin D., and Moerchen F., "Hierarchical Document Clustering Using Local Patterns," *Data Mining and Knowledge Discovery*, vol. 21, no. 1, pp. 153-185, 2010.

[15] Morzy T., Wojciechowski M., and Zakrzewicz M., "Pattern-Oriented Hierarchical Clustering," *in Proceedings of 3rd East European Conference on Advances in Databases and Information Systems*, Maribor, pp. 179-190, 1999.

[16] Rijsbergen C., *Information Retrieval*, Buttersworth, 1979.

[17] Shankar K., Kiran G., and Pudi V., "Evolutionary Clustering using Frequent Itemsets," *in Proceedings of 1st International Workshop on Novel Data Stream Pattern Mining Techniques*, Washington, pp. 25-30, 2010.

[18] Steinbach M., Karypis G., and Kumar V., "A Comparison of Document Clustering Techniques," Technical Report, University of Minnesota, 2000.

[19] Tunali V., Turgay B., and Ali C., "An Improved Clustering Algorithm for Text Mining: Multi-Cluster Spherical K-Means," *The International Arab Journal of Information Technology*, vol. 13, no. 1, pp. 12-19, 2016.

[20] Xiong H., Steinbach M., Tang P., and Kumar V., "HICAP: Hierarchical Clustering With Pattern Preservation," *in Proceedings of SIAM International Conference on Data Mining*, Florida, pp. 279-290, 2004.

[21] Yang Y. and Padmanabhan B., "GHIC: A Hierarchical Pattern-Based Clustering Algorithm for Grouping Web Transactions," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 9, pp. 1300-1304, 2005.

[22] Zamir O., Etzioni O., Madani O., and Karp R., "Fast and Intuitive Clustering of Web Documents," *in Proceedings of 3rd International Conference on Knowledge Discovery and Data Mining*, Newport Beach, pp. 287-290, 1997.

[23] Zhang W., Yoshida T., Tang X., and Wang Q., "Text Clustering Using Frequent Itemsets," *Knowledge-Based Systems*, vol. 23, no. 5, pp. 379-388, 2010.

Syed Shah is pursuing Ph.D. (in Computer Engineering) from Jamia Millia Islamia, New Delhi, India. He has 7 years of academic experience. His research interests include data mining and big data analytics.

Mohammad Amjad has obtained his B.Tech. degree in Computer Engineering from Aligarh Muslim University, Aligarh. He obtained his M.Tech. degree in Information Technology from IP University, New Delhi and Ph.D. in Computer Engineering from Jamia Millia Islamia, New Delhi. Dr. Amjad is currently working as Assistant Professor in the Department of Computer Engineering, Faculty of Engineering & Technology, Jamia Millia Islamia (Central University), New Delhi. He has four years industry experience and 15 years of teaching experience. He has contributed thirty research papers in various reputed journals, national and international conferences including countries like USA and China. He is actively involved in research and development activities in areas of MANET, WSN, software engineering, mobile computing, network security systems and allied areas.