ISSN:0975-9646

A Theoretical Study of Text Document Clustering

Yogesh Jain [#1], Amit Kumar Nandanwar [*2]

[#]*M.Tech Scholar, C.S.E., Vidya Niketan Samiti Group of Institutions, Bhopal, India*
Assistant Professor, C.S.E., Vidya Niketan Samiti Group of Institutions, Bhopal, India

Abstract— **The objective of clustering is to partition an unstructured set of objects into clusters (groups). One often wants to group similar objects in same cluster and dissimilar in different clusters as far as feasibly possible. Clustering is a widely studied data mining problem in text domain. The aim of this paper is to provide an understanding about applying clustering to text documents. It thoroughly discusses about document pre-processing, applications of text clustering, key methods for text clustering, their relative advantages and limitations. Besides this, we will also discuss recent advances in this area.**

Keywords— **Text clustering, Feature selection, Preprocessing.**

I. Introduction

Today we are facing an ever increasing volume of text documents [25]. The voluminous texts flowing over the Internet, enormous collection of documents in digital libraries and repositories, and digital information such as blogs and emails are growing up rapidly day by day. It brings upon the challenges that how to organize text documents effectively and efficiently. The problem of clustering has been studied widely in the database and statistics literature in the context of a wide variety of data mining tasks [19, 5]. The clustering problem can be defined as finding groups of similar objects in the data. The similarity between two different objects is measured with the use of a similarity function. The problem of clustering can be very valuable in the text domain, as the objects to be clustered can be of different granularities such as documents, paragraphs, sentences, terms etc. In order to apply most of the clustering algorithms, two things are required: representing an object, and a similarity measure between objects. A clustering algorithm finds a partition of a set of objects that fulfills some criterion based on similarity measure.

The text clustering is the problem of automatically grouping of free text documents. The groups are usually described by a set of keywords or phrases that described the common content of the documents in the group. To perform a clustering process, the objects should have some kind of attributes to measure the distance or similarity among the objects. These attributes are usually called as features of the object. Most of the proposals in this field consider the document as a set of words. In these representations, each feature corresponds to a single word found in the document set. As a document set may contain several thousand of words, these results in a very high impracticable dimensionality. To reduce the document space dimensionality some word reduction methods are applied in the pre-processing phase. The most common method to reduce the number of different words is to eliminate the words with low information value. These words are called stop words. The stop words are collected into a dictionary or a list. Another way for reduction is based on the statistical properties of the words: the infrequent and the frequent words are filtered out from the original text.

II. Properties Of Text Documents

Naive techniques do not typically work well for clustering text data because text data has a number of unique properties which requires specialized algorithms. The distinguishing characteristics of text representation are as follows:

A. The dimensionality of the text representation is very dense whereas the underlying data is sparse. In other words, the lexicon from which documents are drawn may be of the order of 100, but the document itself may contain only a few hundred words. This problem becomes even more serious when a document to be clustered is very short such as sentences or tweets.

B. While the lexicon of a given documents may be large, the words are typically correlated with one another. It means that the number of principal components in data is much smaller than the feature space. This necessitates the careful design of algorithms which can account for word correlations in the clustering process.

C. The number of words in different documents may vary widely and hence it is essential to normalize the document representations appropriately during clustering task.

The sparse and high dimensional representation of different documents calls for the design of text-specific clustering algorithms for document representation and processing. If the number of words is equal to N, the number of phrases containing k words is N^k. Thus, in order to reduce computational cost, specialized algorithms are required for different phases of the text document clustering process. Many techniques have been proposed to optimize document representation for improving the accuracy of matching a document with a query [4, 15].

III. Properties Of Text Documents

The major research in text clustering has been done in context of following two kinds of text data: Dynamic Applications and Heterogeneous Applications. Today, a large amount of text data is being created by dynamic applications, for example social networks or online chat applications. Such streaming applications have to be applicable in case of non-filtered text. Text applications are increasingly arising in heterogeneous applications where the text is available in context of hyperlinks and other heterogeneous multimedia data.

The cluster hypothesis proposes that “closely associated documents tend to be relevant to the same request” [13], i.e. similar documents are believed to be relevant to the same queries put to a search engine. This has made many researchers believe that a credible clustering could make search time shorter as clusters could be retrieved instead of documents. [12, 14] have shown an efficient way to cluster web search engine results. The search engine Vivisimo1 uses clustering on retrieved documents.

The Scatter/Gather system (or any clustering method) has also been proposed for browsing document collections [17]. A document collection is presented to a user as a set of clusters. The user may mark one or several clusters for further investigation and request that these are re-clustered giving a more fine tuned grouping. In this way the user may iteratively and interactively explore the collection and get an overview of its content as well as find particular themes that appear in it.

Clustering methods can be used to automatically group the retrieved documents into a list of meaningful categories, as is achieved by Enterprise Search engines such as Northern Light and Vivisimo or open source software such as Carrot2. Also, Google is known to use clustering methods to match certain websites with a query, since a website can be viewed as a collection of topics (multi-topic document), and a query itself is a topic or a combination of several topics.

Finally, with the rising of social network in recent years, such as Facebook and Twitter, more semantic data are available now that convey a considerable amount of information. On Twitter, there are approximately 95M tweets per day, which is equivalent to 1100 tweets per second [23]. Researchers from the Northeastern University College of Computer and Information Sciences, and Harvard Medical School have developed an innovative way of tracking the nation's mood using tweets. All these researches show the power of social computing in providing accurate assessments on many sorts of issues, at almost no cost and on a large scale. Likewise, document clustering techniques can be used to group tweets into relevant topics, in aid of the current mere 'Trends' function used by Twitter. For all these reasons, we find document clustering techniques valuable and therefore worth studying.

IV. DOCUMENT PRE-PROCESSING

Before we represent documents as *TF-IDF* vectors, we need some preprocessing. Firstly, we need to remove stop words, such as *'a'*, *'any'*, *'what'*, *'I'*, etc, since they are frequent and carry no information. A stop words list can be found online. Secondly, we need to stem the word to its origin, which means we only consider the root form of words. For example, ran, running, runs are all stemmed to run, and happy, happiness, happily are all stemmed to happy. A more elaborate way of stemming is by using the *WordNet*, which in addition to suffix-stripping also groups words into *synsets*, and leads to an ontology-based (instead of word-based) document clustering method.

V. FEATURE SELECTION AND TRANSFORMATION METHODS

The quality of any data mining method such as classification and clustering is highly dependent on the noisiness of features that are used for the clustering process. For example, commonly used words such as “the”, may not be very useful in improving the clustering quality. Therefore, it is critical to select the features effectively, so that the noisy words in the corpus are removed before the clustering.

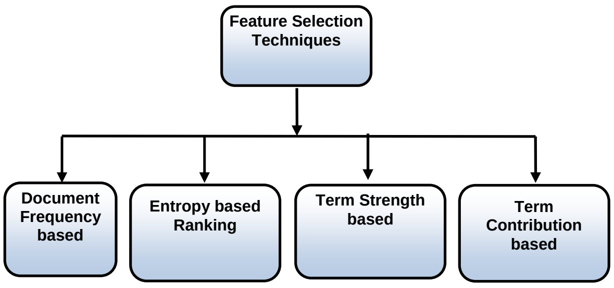

Fig. 1 Example of a figure caption

As shown in Fig. 1, the simplest possible method for feature selection in document clustering is *document frequency* that is used to filter out irrelevant features. While the use of inverse document frequencies reduces the importance of such words, this may not alone be sufficient to reduce the noise effects of very frequent words. In other words, words which are too frequent in the corpus can be removed because they are typically common words, which are not discriminative from a clustering perspective. Such words are also referred to as *stop words*.

Another method namely the *term strength* is essentially used to measure how informative a word is for identifying two related documents. For example, for two related documents x and y, the term strength $s(t)$ of term t is defined in terms of the following probability:

$$s(t) = P(t \in y | t \in x)$$

Here, the main issue is how one might define the document x and y as related. One possibility is to use manual (or user) feedback to define when a pair of documents is related. It is possible to use automated similarity functions such as the cosine function to define the relatedness of document pairs. A pair of documents is defined to be related if their cosine similarity is above a user-defined threshold. In such cases, the term strength $s(t)$ can be defined by randomly sampling a number of pairs of such related documents as follows:

$$s(t) = \frac{\text{Number of pairs in which } t \text{ occurs in both}}{\text{Number of pairs in which } t \text{ occurs in the first of the pair}}$$

Where, the first document of the pair may simply be picked randomly. Another method called *Entropy based ranking* measures the quality of a term by the entropy reduction when it is removed. Here the entropy $E(t)$ of the term t in a collection of n documents is defined as follows:

$$E(t) = -\sum^{n}\sum^{n}(S_{ij} \times \log(S_{ij}) + (1 - S_{ij}) \times \log(1 - S_{ij}))$$

Here $S_{ij} \in (0, 1)$ is the similarity between the i^{th} and j^{th} document in the collection, after the term t is removed, and is expressed as follow:

$$S_{ij} = 2^{\frac{-dist(i,j)}{dist}}$$

Where $dist(i, j)$ is the distance between the terms i and j after the term t is removed, and $dist$ is the average distance between the documents after the term t is removed. Another approach called *Term Contribution* considers the contribution of terms for document similarity. The contribution of a term in the similarity of two documents is the product of their normalized frequencies in the two documents.

Feature selection attempts to select features from original data set whereas in feature transformation, new features are defined as a functional representation of the features in the original data set. A number of well known feature transformation methods such as *Latent Semantic Indexing (LSI)*, *Probabilistic Latent Semantic Analysis (PLSA)*, and *Non-negative Matrix Factorization (NMF)* are available to improve the quality of representing a document and make it more amenable for clustering. The LSI based methods are used to reduce the dimensions of noisy data. The *PLSA* is similar to LSI technique, but is based upon probabilistic modeling. The *NMF* technique is a latent space method, which like *LSI*, represents the documents in a new axis system based on an analysis of the term-document matrix. But unlike *LSI*, the vectors in the basis system directly correspond to the cluster topics.

VI. Text Clustering Techniques

The clustering techniques can broadly be classified into five types as shown in Fig. 2: *Distance-based* clustering algorithms are designed by using a similarity function to measure the closeness between the text objects. The *Euclidean distance* of two documents X_1 and X_2 is defined as:

$$D(X_1, X_2) = \sqrt{(X_1 - X_2)(X_1 - X_2)^T}$$
$$= \sqrt{\sum_{i=1}^{m}(x1_i - x2_i)^2}$$

The smaller the value of $D(X_1,X_2)$ is, the more similar the two documents are. The most well known similarity function, which is used commonly in the text domain, is the cosine similarity function. The cosine similarity measurement between vector v_1 and v_2 can be computed as follow:

$$\cos(v_1, v_2) = \frac{(v_1 \cdot v_2)}{(\|v_1\| \times \|v_2\|)}$$

One challenge in clustering short segments of text (e.g., tweets or sentences) is that exact keyword matching may not work well. One general strategy for solving this problem is to expand text representation by exploiting related text documents, which is related to smoothing of a document language model in information retrieval [24].

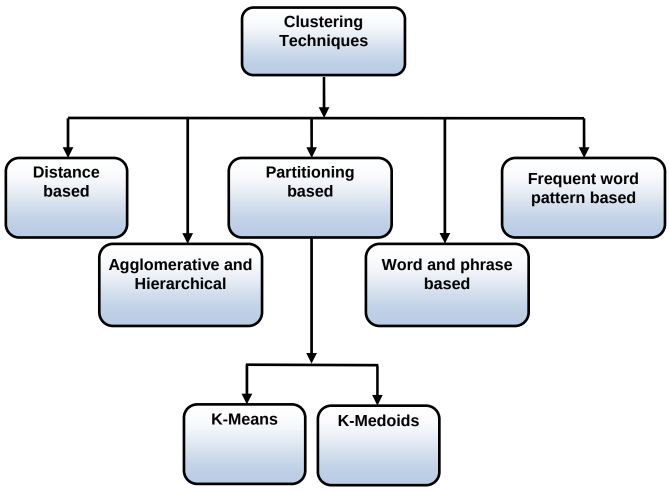

Fig. 2. Broad classification of Document Clustering Techniques

The method of agglomerative hierarchical clustering is particularly useful to support a variety of searching methods because it naturally creates a tree-like hierarchy, which can be leveraged for the search process. It forms clusters in a bottom-up manner. Let us suppose there be two documents A and B then the distance between these two documents is defined by:

$$d(A, B) = LL(A) + LL(B) - LL(A \cup B)$$

Where LL(X) represents the log likelihood of an article or cluster X given by a unigram model given below:

$$LL(X) = log \prod_{w \in X} p_x(w)^{c_x(w)}$$
$$= \sum_{w \in X} c_x(w)\ log\ c_x(w) - N_x log N_x$$

Where $c_x(w)$ and $p_x(w)$ are the count and probability, respectively, of word w in cluster X, and N_x is the total number of words occurring in cluster X. The general concept of agglomerative clustering is to successively merge documents into clusters based on their similarity with one another. Almost all the hierarchical clustering algorithms successively merge groups based on the best pair-wise similarity between these groups of documents. The main differences between these classes of methods are in terms of how this pair-wise similarity is computed between the different groups of documents.

Conceptually, the process of agglomerating documents into successively higher levels of clusters creates a cluster hierarchy (or dendogram) for which the leaf nodes correspond to individual documents, and the internal nodes correspond to the merged groups of clusters. When two groups are merged, a new node is created in tree corresponding to the larger merged group. The two children of a node correspond to the two groups of documents which have been merged to it. There are various methods for merging groups of documents for different agglomerative methods such as Single Linkage Clustering, Group-Average Linkage Clustering, and Complete Linkage Clustering.

Partitioning based algorithms are widely used in literature to efficiently create clusters of objects. Two most widely [5, 19] used partitioning algorithms are k-medoid clustering algorithm and k-means clustering algorithm. The *K-means* clustering algorithm uses a set of *K* representatives around which the clusters are built. Quality of *K-means* clustering is measured through the within-cluster squared error criterion. It aims to minimize the following objective function:

$$J = \sum_{j=1}^{k}\sum_{i=1}^{n}\left\|x_i^{(j)} - c_j\right\|^2 \quad \textit{Where } 1 \leq m < \infty$$

Where $\left\|x_i^{(j)} - c_j\right\|^2$ is a chosen distance measure between a data point $x_i^{(j)}$ and the cluster centre c_j is an indicator of the distance of the *n* data points from their respective cluster centers. The simplest form of the *K-means* approach is to start off with a set of k seeds from the original corpus, and assign documents to these seeds on the basis of closest similarity. In the next iteration, the centroid of the assigned points to each seed is used to replace the seed in the last iteration. In other words, the new seed is defined, so that it is a better central point for this cluster. This approach is continued until convergence. However, these representatives are not necessarily obtained from the original data and are refined somewhat differently than a *K-medoids* approach.

In *K-medoid*, a set of points is used from the original data as the anchors (or medoids) around which the clusters are built. This algorithm generally returns a higher value of $\left\|x_i^{(j)} - c_j\right\|^2$ and is computationally harder than *K-means* due to the fact that computing medoid is harder than computing average. The key aim of the algorithm is to determine an optimal set of representative documents from the original corpus around which the clusters are built.

Since text documents are drawn from an inherently high-dimensional domain, it can be useful to view the problem in a dual way, in which important clusters of words may be found and utilized for finding clusters of documents. Good clusters of words may be leveraged in order to find good clusters of documents and vice-versa. For example, the work in [10] determines frequent itemsets of words in the document collection, and uses them to determine compact clusters of documents. This is somewhat analogous to the use of clusters of words for determining clusters of documents. The most general technique for simultaneous word and document clustering is referred to as co-clustering [7, 20]. It is important to understand that the problem of word clusters and document clusters are essentially dual problems which are closely related to one another. The former is related to dimensionality reduction, whereas the latter is related to traditional clustering. The boundary between the two problems is quite fluid, because good word clusters provide hints for finding good document clusters and vice-versa.

Frequent pattern mining [16] is a technique which has been widely used in the data mining literature in order to determine the most relevant patterns in transactional data. The clustering approach in [10] is designed on the basis of such frequent pattern mining algorithms. A frequent itemset in the context of text data is also referred to as a frequent term set. The main idea of the approach is to not cluster the high dimensional document data set, but consider the low dimensional frequent term sets as cluster candidates. This essentially means that a frequent terms set is a description of a cluster which corresponds to all the documents containing that frequent term set. Since a frequent term set can be considered a description of a cluster, a set of carefully chosen frequent terms sets can be considered a clustering. The appropriate choice of this set of frequent term sets is defined on the basis of the overlaps between the supporting documents of the different frequent term sets. Consider a collection of transactions to be clustered *{T_1, T_2, …, T_n}*. Each transaction T_i contains a subset of a list of candidate items *{i_1 i_2 ,… , i_s}*. A clustering *C* is a partition *{C_1 ,C_2 ,… ,C_k}* of *{T_1 ,T_2 ,…,T_n }* and each C_i is a cluster. Here we only consider hard clustering where a transaction belongs to exactly one cluster. The goal is to maximize objective function *M*, which is defined as follows:

$$M(C1, C2, \dots, Ck) = Difference(C1, C2, \dots, Ck) + \sum_{i=1}^{k} Similarity(C_i)$$

Where *S(P_a,C_d)* represents the support of pattern P_a in cluster C_d , and P_1 through P_m are the set of patterns based on which the difference between two clusters is computed as follow:

The intuition behind the definition of difference is that the support of any pattern in one cluster should be different from the support in the other cluster.

$$Difference(C_i, C_j) = \sum_{a=1}^{m} \frac{\left|S(P_a, C_i) - S(P_a, C_j)\right|}{\frac{1}{2}\times\left[S(P_a, C_i) - S(P_a, C_j)\right]}$$

VII. Related Work

Text clustering algorithms are divided into a wide variety of different types such as agglomerative clustering algorithms, partitioning algorithms, and standard parametric modeling based methods such as the EM-algorithm. Furthermore, text representations may also be treated as strings (rather than bags of words). These different representations necessitate the design of different classes of clustering algorithms. Different clustering algorithms have different tradeoffs in terms of effectiveness and efficiency. An experimental comparison of different clustering algorithms may be found in [8, 21].

The bag of words representation used for these clustering is often unsatisfactory as it ignores relationships between important terms that do not co-occur literally. In order to deal with the problem, [3] integrated the core ontologies as background knowledge into the process of clustering text documents. This model combines phrases analysis as well as words analysis with the use of *WordNet* as background Knowledge and NLP to explore better ways of document representation for clustering. The Semantic based analysis assigns semantic weights to both document words and phrases. The new weights reflect the semantic relatedness between the documents terms and capture the semantic information in the documents to improve the web document clustering.

In [7], authors focused on clustering Slovak text documents from *Wikipedia* into specific categories using different clustering algorithms such as agglomerative hierarchical clustering, divisive hierarchical clustering, K-Means, K-Medoids and self-organizing maps. They compared these algorithms according to several term weighting schemes such as *Term Frequency, Inverse Document Frequency, Residual IDF, Okapi* and others. They also used Principal Component Analysis to illustrate the document vectors in three-dimensional space. We used purity and entropy to evaluate the clustering results. The best results were obtained by agglomerative hierarchical clustering using TF-IDF as a term weighting scheme.

Shailendra et.al [21] proposed a novel text document clustering algorithm called Phrase affinity clustering (*PAC*) based on vector space model, phrases and affinity propagation clustering algorithm. *PAC* first finds the phrase by ukkonen suffix tree construction algorithm, second finds the vector space model using *TF-IDF* weighting scheme of phrase. Third calculate the similarity matrix form *VSD* using cosine similarity. At last, affinity propagation algorithm generates the clusters. F-Measure, Purity and Entropy of Proposed algorithm is better than *GAHC, ST-GAHC and ST-KNN* on *OHSUMED, RCV1* and *News group* data sets.

Wui Lee Chang et.al [22] proposed an Evolving Tree (*E-Tree*) model with Fuzzy C-Means (*FCM*) for undertaking text document visualization problems. It forms a hierarchical tree structure in which nodes are allowed to grow and split into child nodes, and each node represents a cluster of documents. This model adopts a relatively simple approach to split its nodes, and can be seen as an alternative to perform node splitting in *E-Tree*.

Information-theoretic clustering aims to exploit information-theoretic measures as the clustering criteria. A common practice on this topic is the so-called *Info-Kmeans*, which performs *K-means* clustering with *KL-divergence* as the proximity function. While expert efforts on *Info-Kmeans* have shown promising results, a remaining challenge is to deal with high-dimensional sparse data such as text corpora. Indeed, it is possible that the centroids contain many zero-value features for high-dimensional text vectors, which leads to infinite *KL-divergence* values and creates a dilemma in assigning objects to centroids during the iteration process of *Info-Kmeans*. To meet this challenge, Jie Cao [11] et.al proposed a Summation-bAsed Incremental Learning (*SAIL*) algorithm for Info-Kmeans clustering. Specifically, by using an equivalent objective function, *SAIL* replaces the computation of *KL-divergence* by the incremental computation of Shannon entropy. This can avoid the zero-feature dilemma caused by the use of *KL-divergence*. To improve the clustering quality, they also proposed the *V-SAIL* algorithm accelerated by a multithreaded scheme in *PV-SAIL*. They showed that with *SAIL* as a booster, the clustering performance of *Info-Kmeans* can significantly be improved. Also, *V-SAIL* and *PV-SAIL* help to improve the clustering quality at a lower computational cost.

A. Benghabrit et.al [1] proposed a new sequential document clustering algorithm that uses a statistical and semantic feature selection methods. It improves the frequency mechanism with the semantic relations of text documents by iteratively selecting the relevant features and performing clustering until convergence. Andrew Skabar et.al [2] proposed a novel fuzzy clustering algorithm that operates on relational input data i.e. data in the form of a square matrix of pair wise similarities between data objects. It uses a graph representation of data, and operates in an Expectation-Maximization framework in which the graph centrality of an object in graph is interpreted as a likelihood. They showed that the algorithm is capable of identifying overlapping clusters of semantically related sentences, and therefore it is of potential use in a variety of text mining tasks.

VIII. Conclusion

Clustering of huge, diverse and rapidly changing text documents is a very complex task. Clustering results mainly depend upon the document set on which clustering is applied and the parameters used for clustering criteria. For getting better clustering result, it is very important to select clustering parameters very precisely. Clustering has gained much attention in last few years but more research is still needed for addressing the issues such as achievement of better quality-complexity tradeoffs, as well as to deal with each method's disadvantages. In addition, one of the challenges is to deal with dimensionality because text documents may contain huge amount of terms. Another important issue is that often a document belongs to more than one cluster, this type of problem is referred to as an "any-of" problem. To deal with this issue, algorithms should be developed that allow overlapping of clusters. In the end, additional efforts should be done to improvise the description of clusters contents from user's point of view; this can be done with proper labeling and providing detailed information for each cluster.

References

[1] A. Benghabrit, B. Ouhbi, H. Behja and B. Frikh, 2013. *Text clustering using statistical and semantic data*. Proceedings of the IEEE World Congress on Computer and Information Technology, Jun. 22-24, pp: 1-6. DOI: 10.1109/WCCIT.2013.6618782.

[2] Andrew Skabar and Khaled Abdalgader. 2013. *Clustering Sentence-Level Text Using a Novel Fuzzy Relational Clustering Algorithm*. IEEE Transactions on Knowledge and Data Engineering. 25: 62-75. DOI: 10.1109/TKDE.2011.205.

[3] B. Drakshayani and E. V. Prasad. 2012. *Text Document Clustering based on Semantics.* International Journal of Computer Applications, 45: 7-12. DOI: 10.5120/6766-9046.

[4] Beil, M. Ester, X. Xu. 2002. *Frequent term-based text clustering.* Proceedings of the ACM KDD Conference. USA, pp: 436-442. DOI: 10.1145/775047.775110.

[5] C. Salton and Buckley. 1988. *Term Weighting Approaches in Automatic Text Retrieval*. Information Processing and Management. 24:513–523.

[6] Charu C. Aggarwal and ChengXiang Zhai. 2012. *A Survey of Text Clustering Algorithms. Mining Text Data.* Springer US. pp: 77-128. ISBN: 978-1-4614-3222-7. DOI: 10.1007/978-1-4614-3223-4_4

[7] Daniel Zlacký, Ján Staš, Jozef Juhár and Anton Čižmár. 2013. *Slovak Text Document Clustering*. Acta Electrotechnica et Informatica, 13: 3–7, DOI: 10.2478/aeei-2013-0022.

[8] Inderjit S. Dhillon, Subramanyam Mallela and Dharmendra S. Modha . 2003. *Information-theoretic Co- Clustering*. Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining. Aug. 24-27. NY, USA. pp: 89-98. DOI: 10.1145/956750.956764.

[9] Inderjit S. Dhillon. 2001. *Co-clustering Documents and Words using bipartite spectral graph partitioning*. Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining. Aug. 26-29, NY, USA. pp: 269-274. DOI: 10.1145/502512.502550.

[10] C. J. Van Rijsbergen. 1979. *Information Retrieval*. Journal of the American Society for Information Science. Wiley Periodicals Inc. 30: 374-375. DOI: 10.1002/asi.4630300621.

[11] Jie Cao, Zhiang Wu, Junjie Wu and Hui Xiong. 2013. SAIL: *Summation-Based Incremental Learning for Information-Theoretic Text Clustering*. IEEE Transactions on Cybernetics, 43: 570-584. DOI: 10.1109/TSMCB.2012.2212430

[12] L. Kaufman and P. J. Rousseeuw. 2005. *Finding Groups in Data: An Introduction to Cluster Analysis*. Wiley Interscience. ISBN: 978-0-471-73578-6.

[13] M. Beil, Martin Ester and Xiaowei Xu. 2002. *Frequent term-based text clustering*. Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining. Jul. 23-25. NY, USA. pp: 436-442. DOI: 10.1145/775047.775110.

[14] Michael Steinbach , George Karypis and Vipin Kumar. 2000. *A Comparison of Document Clustering Techniques*. KDD Workshop on text mining.

[15] Oren Zamir and Oren Etzioni. 1998. *Web document clustering: A feasibility demonstration*. Research and Development in Information Retrieval. Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval. NY, USA. pp: 46–54. DOI: 10.1145/290941.290956.

[16] Oren Zamir, Oren Etzioni, Omid Madani and Richard M. Karp. 1997. *Fast and intuitive clustering of web documents*. Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining. Aug. 14–17. California, USA. pp: 287–290. DOI:

[17] R. A. Baeza-Yates and B. A. Ribeiro-Neto. 2011. *Modern Information Retrieval - the concepts and technology behind search*. Pearson Education Ltd. Harlow, England. ISBN: 0321416910.

[18] R. Agrawal and R. Srikant. 1994. *Fast Algorithms for Mining Association Rules in Large Databases*. Proceedings of the 20th International Conference on Very Large Data Bases. San Francisco, Sep. 12-15. CA, USA. pp: 487-499.

[19] A. K. Jain and R. C. Dubes. 1988. *Algorithms for Clustering Data*. Prentice Hall, NJ, USA. ISBN: 0-13-022278-X

[20] D. R. Cutting, D. R. Karger, J. O. Pedersen and J. W. Tukey. 1992. Scatter/gather: *A cluster-based approach to browsing large document collections*. Proceedings of 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. NY, USA. pp: 318-329. DOI: 10.1145/133160.133214.

[21] Shailendra Kumar Shrivastava, J. L. Rana and R. C. Jain. 2013. *Text Document Clustering based on Phrase Similarity using Affinity Propagation*. International Journal of Computer Applications. 61: 38-44. DOI: 10.5120/10032-5077.

[22] Wui Lee Chang, Kai Meng Tay and Chee Peng Lim. 2013. *Enhancing an Evolving Tree-based text document visualization model with Fuzzy c-Means clustering*. Proceedings of the IEEE International Conference on Fuzzy Systems, Jul. 7-10. pp: 1-6. DOI: 10.1109/FUZZ-IEEE.2013.6622363

[23] Y. Zhao and G. Karypis. 2004. *Empirical and Theoretical comparisons of selected criterion functions for document clustering*. Machine Learning, Kluwer Academic Publishers Hingham, MA, USA. 55: 311–331. DOI: 10.1023/B:MACH.0000027785.44527.d6

[24] Yu Xiao. 2010. *A Survey of Document Clustering Techniques & Comparison of LDA and moVMF*.

[25] ChengXiang Zhai. 2008. *Statistical Language Models for Information Retrieval*. Synthesis Lectures on Human Language Technologies, Morgan & Claypool Publishers.