

## A Survey On Document Clustering With Hierarchical Methods And Similarity Measures

N.RaghavaRao<sup>1</sup> K. Sravankumar<sup>2</sup> P. Madhu<sup>3</sup>

<sup>1</sup>( Asso.Prof ,HOD CSE, DRK institute of science of technology/ JNTUH, India)

<sup>2</sup>(M.Techstudent ,DRK institute of science of technology/ JNTUH, India)

<sup>3</sup>((M.Techstudent ,DRK college of engineering & technology/ JNTUH, India)

### **ABSTRACT:**

Clustering is a division of data into groups of similar objects. Representing the data by fewer clusters necessarily loses certain fine details, but achieves simplification. Cluster is nothing but a group. In the view point of data engineering cluster is a group of objects with similar nature. The grouping mechanism is called as clustering. The similar documents are grouped together in a cluster, if their cosine similarity measure is less than a specified threshold

**In this paper we mainly focuses on view points and measures in hierarchical** clustering. We introduce a novel multi-viewpoint based similarity measure and two related clustering methods , the objects assumed to not be in the same cluster with the two objects being measured. Using multiple viewpoints, more informative assessment of similarity could be achieved.

**KEY Terms**—Document clustering, text mining, similarity measure .Hierarchical methods

### **Overview :**

Document clustering techniques mostly rely on single term analysis of the document data set, such as the Vector Space Model. To achieve more accurate document clustering, more informative features including phrases and their weights are particularly important in such scenarios. Document clustering is particularly useful in many applications such as automatic categorization of documents, grouping search engine results, building taxonomy of documents, and others. For this Hierarchical Clustering method provides a better improvement in achieving the result. Our project presents two key parts of successful Hierarchical document clustering. The first part is a document index model, the Document Index Graph, which allows for incremental construction of the index of the document set with an emphasis on efficiency, rather than relying on single-term indexes only. It provides efficient phrase matching that is used to judge

the similarity between documents. This model is flexible in that it could revert to a compact representation of the vector space model if we choose not to index phrases. The second part is an incremental document clustering algorithm based on maximizing the tightness of clusters by carefully watching the pair-wise document similarity distribution inside clusters. Both the phases are based upon two algorithmic models called Gaussian Mixture Model and Expectation Maximization. The combination of these two components creates an underlying model for robust and accurate document similarity calculation that leads to much improved results in Web document clustering over traditional methods.

### **Existing approaches**

- The aim of clustering is to find intrinsic structures in data, and organize them into meaningful subgroups for further study and analysis. There have been many clustering algorithms published every year.
- Existing Systems greedily picks the next frequent item set which represent the next cluster to minimize the overlapping between the documents that contain both the item set and some remaining item sets.
- The clustering result depends on the order of picking up the item sets, which in turns depends on the greedy heuristic. This method does not follow a sequential order of selecting clusters. Instead, we assign documents to the best cluster.

### **PROPOSED APPROACH**

The main work is to develop a novel hierarchical algorithm for document clustering which provides maximum efficiency and performance. It is particularly focused in studying and making use of cluster overlapping phenomenon to design cluster merging criteria. Proposing a new way to compute the overlap rate in order to improve time efficiency and “the veracity” is mainly concentrated. Based on the Hierarchical Clustering Method, the usage of Expectation-Maximization (EM) algorithm in the Gaussian Mixture Model to count the parameters and make the two sub-clusters combined when their overlap is the largest is narrated.

- Experiments in both public data and document clustering data show that this approach can improve the efficiency of clustering and save computing time.

### **CHALLENGES IN HIERARCHICAL DOCUMENT CLUSTERING**

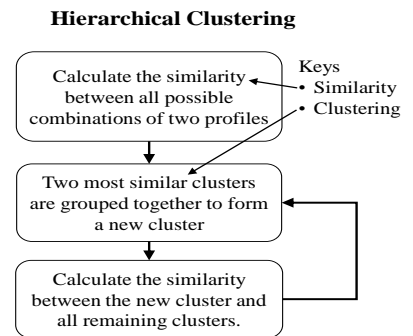
- High dimensionality: Each distinct word in the document set constitutes a dimension. So there may be 15~20 thousands dimensions. This type of high dimensionality greatly affects the scalability and

efficiency of many existing clustering algorithms. This is been cleared described in the following paragraphs.

- High volume of data: In text mining, processing of data about 10 thousands to 100 thousands documents are involved.
- Consistently high accuracy: Some existing algorithms only work fine for certain type of document sets, but may not perform well in some others.
- Meaningful cluster description: This is important for the end user. The resulting hierarchy should facilitate browsing.

### HIERARCHICAL ANALYSIS MODEL

A hierarchical clustering algorithm creates a hierarchical decomposition of the given set of data objects. Depending on the decomposition approach, hierarchical algorithms are classified as agglomerative (merging) or divisive (splitting). The agglomerative approach starts with each data point in a separate cluster or with a certain large number of clusters. Each step of this approach merges the two clusters that are the most similar. Thus after each step, the total number of clusters decreases. This is repeated until the desired number of clusters is obtained or only one cluster remains. By contrast, the divisive approach starts with all data objects in the same cluster. In each step, one cluster is split into smaller clusters, until a termination condition holds. Agglomerative algorithms are more widely used in practice. Thus the similarities between clusters are more researched.



### Hierarchical Clustering

#### HOW THEY WORK?

Given a set of  $N$  items to be clustered, and an  $N*N$  distance (or similarity) matrix, the basic process of hierarchical clustering is this:

STEP 1 - Start by assigning each item to a cluster, so that if you have  $N$  items, you now have  $N$  clusters, each containing just one item. Let the distances (similarities) between the clusters the same as the distances (similarities) between the items they contain.

STEP 2 - Find the closest (most similar) pair of clusters and merge them into a single cluster, so that now you have one cluster less with the help of it.

STEP 3 - Compute distances (similarities) between the new cluster and each of the old clusters.

STEP 4 - Repeat steps 2 and 3 until all items are clustered into a single cluster of size  $N$ .

Step 3 can be done in different ways, which is what distinguishes single-linkage from complete-linkage and average-linkage clustering. In single-linkage clustering (also called the connectedness or minimum method), considering the distance between one cluster and another cluster to be equal to the

shortest distance from any member of one cluster to any member of the other cluster.

If the data consist of similarities, consider the similarity between one cluster and another cluster to be equal to the greatest similarity from any member of one cluster to any member of the other cluster. In complete-linkage clustering (also called the diameter or maximum method), consider the distance between one cluster and another cluster to be equal to the greatest distance from any member of one cluster to any member of the other cluster. In average-linkage clustering, consider the distance between one cluster and another cluster to be equal to the average distance. This kind of hierarchical clustering is called agglomerative because it merges clusters iteratively. There is also a divisive hierarchical clustering which does the reverse by starting with all objects in one cluster and subdividing them into smaller pieces. Divisive methods are not generally available, and rarely have been applied.

Of course there is no point in having all the N items grouped in a single cluster but, once the complete hierarchical tree is obtained and need k clusters, k-1 longest links are eliminated.

### TERM FREQUENCY - INVERSE DOCUMENT

**FREQUENCY** The TF-IDF is a text statistical-based technique which has been widely used in many search engines and information retrieval systems. Assume that there is a corpora of 1000 documents and the task is to compute the similarity between two given documents (or a document and a query). The following describes the steps of acquiring the similarity value: **3.1 Document pre-processing steps**

- Tokenization: A document is treated as a string (or bag of words), and then partitioned into a list of tokens.

- Removing stop words: Stop words are frequently occurring, insignificant words. This step eliminates the stop words.
- Stemming word: This step is the process of conflating tokens to their root form (connection -> connect).

### Document representation

Generating N-distinct words from the corpora and call them as index terms (or the vocabulary). The document collection is then represented as a N-dimensional vector in term space.

### Computing Term weights

- Term Frequency.
- Inverse Document Frequency.
- Compute the TF-IDF weighting.

#### 3.1.1. Measuring similarity between two documents:

Capturing the similarity of two documents using cosine similarity measurement. The cosine similarity is calculated by measuring the cosine of the angle between two document vectors. Using the code:

The main class is TFIDF Measure. This is the testing code:

```
void Test (string[] docs, int i, int j)
// docs is collection of parsed documents
{
StopWordHandler stopWord=new StopWordsHandler() ;
TFIDFMeasure tf=new TFIDFMeasure(doc) ; float
simScore=tf.GetSimilarity( i, j); // similarity of two given
documents at the // position i,j respectively }
```

#### Extension

This library also includes stemming (Martin Porter algorithm), and N-gram text generation modules. If a token-based system did not work as expected, then make another choice with N-gram based. Thus, instead of expanding the list of tokens from the document, generating a list of N-grams is adopted, where N should be a predefined number. The extra N-gram based similarities (bi, tri, quad...-gram) also help to compare the result of the statistical-based method with the N-gram based method. Consider two documents as two flat texts and then run the measurement to compare. Example of some N-grams for the word "TEXT":

- uni(1)-gram: T, E, X, T
- bi(2)-gram: T, TE, EX, XT, T
- tri(3)-grams: TE, TEX, EXT, XT, T
- quad(4)-grams: TEX, TEXT, EXT, XT, T

#### 3.1.2. The problem, straight Boolean logic

To many of users the phrase “relevancy ranked search results” is a mystery. A better phrase might have been “statistically significant search results”. Taking such an approach, the application of statistical analysis against texts does have its information retrieval advantages over straight Boolean logic.

Take for example, the following three documents consisting of a number of words. A search for “rose” against the corpus will return three hits, but which one should start reading from? The new document? The document by a particular author or in a particular format? Even if the corpus contained 2,000,000 documents and a search for “rose” returned a mere 100 the problem would remain. Which ones should we spend our valuable time accessing? Yes, we could limit our search in any number of ways, but unless we are doing a known item search it is quite likely the search results will return more than we use, and information literacy skills will only go so far. Ranked search results, a list of hits based on term weighting has proven to be an effective way of addressing this problem. All it requires is the application of basic arithmetic against the documents being searched.

Document 1		Document 2		Document 3	
Word	TFIDF	Word	TFIDF	Word	TFIDF
airplane	0.326	Milton	0.439	building	0.367
shoe	0.261	shakespeare	0.293	ceiling	0.245
computer	0.196	car	0.256	cleaning	0.245
perl	0.163	book	0.220	carpet	0.184
chair	0.152	pond	0.146	justice	0.163
justice	0.152	slavery	0.146	perl	0.153
forest	0.130	rose	0.122	rose	0.143
love	0.130	newton	0.110	chair	0.122
might	0.130	chair	0.098	libraries	0.122
rose	0.130	thesis	0.073	newton	0.061
blue	0.065	truck	0.073	science	0.061
thesis	0.065	justice	0.049	car	0.031

Table1: Total Classification

### TFIDF Analysis

By taking into account these two factors — term frequency (TF) and inverse document frequency (IDF) — it is possible to assign “weights” to search results and therefore ordering them statistically. Put another way, a search result’s score (“ranking”) is the product of TF and IDF: **TFIDF = TF \* IDF** where:

- $TF = C / T$  where C = number of times a given word appears in a document and T = total number of words in a document

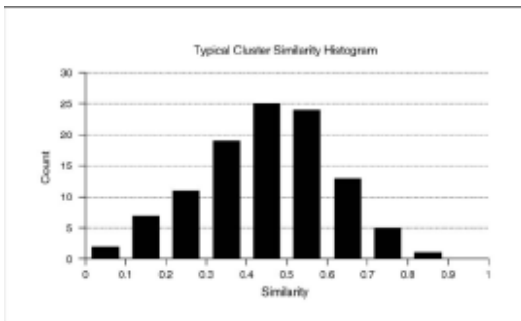
- $IDF = D / DF$  where D = total number of documents in a corpus, and DF = total number of documents containing a given word
- Given TFIDF, a search for “rose” still returns three documents ordered by Documents 3, 1, and 2. A search for “newton” returns only two items ordered by Documents 2 (0.110) and 3 (0.061). In the later case, Document 2 is almost one and a half times more “relevant” than document 3. TFIDF scores can be summed to take into account Boolean unions (or) or intersections (and).

### Automatic classification

TDIDF can also be applied a priori to indexing/searching to create browse lists hence, automatic classification. Consider the table where each word is listed in a sorted TFIDF order: Given such a list it would be possible to take the first three terms from each document and call them the most significant subject “tags”. Thus, Document #1 is about airplanes, shoes, and computers. Document #2 is about Milton, Shakespeare, and cars. Document #3 is about buildings, ceilings, and cleaning. Probably a better way to assign “aboutness” to each document is to first denote a TFIDF lower bounds and then assign terms with greater than that score to each document. Assuming a lower bounds of 0.2, Document #1 is about airplanes and shoes. Document #2 is about Milton, Shakespeare, cars, and books. Document #3 is about buildings, ceilings, and cleaning.

Doc 1	Doc 2	Doc 3
Word	Word	Word
Airplane	book	Building
Blue	car	Car
Chair	chair	Carpet
Computer	justice	Ceiling
Forest	milton	chair
justice	newton	cleaning
Love	pond	justice
Might	rose	libraries
Perl	shakespeare	newton
Rose	slavery	perl
Shoe	thesis	rose
Thesis	truck	science

The clustering approach proposed here is an incremental dynamic method of building the clusters. An overlapped cluster model is adopted here. The key concept for the similarity histogram-based clustering method is to keep each cluster at a high degree of coherency at any time. Representation of the coherency of a cluster is called as Cluster Similarity Histogram.



## Cumulative Document

- The cumulative document is the sum of all the documents, containing meta-tags from all the documents.
- We find the references (to other pages) in the input base document and read other documents and then find references in them and so on.
- Thus in all the documents their meta-tags are identified, starting from the base document.

## CONCLUSION:

Given a data set, the ideal scenario would be to have a given set of criteria to choose a proper clustering algorithm to apply. Choosing a clustering algorithm, however, can be a difficult task. Even ending just the most relevant approaches for a given data set is hard. Most of the algorithms generally assume some implicit structure in the data set. One of the most important elements is the nature of the data and the nature of the desired cluster. Another issue to keep in mind is the kind of input and tools that the algorithm requires. This report has a proposal of a new hierarchical clustering algorithm based on the overlap rate for cluster merging. The experience in general data sets and a document set

indicates that the new method can decrease the time cost, reduce the space complexity and improve the accuracy of clustering. Specially, in the document clustering, the newly proposed algorithm measuring result show great advantages. The hierarchical document clustering algorithm provides a natural way of distinguishing clusters and implementing the basic requirement of clustering as high within-cluster similarity and between-cluster dissimilarity.

## REFERENCES:

- [1] X. Wu, V. Kumar, J. Ross Quinlan, J. Ghosh, Q. Yang, H. Motoda, G. J. McLachlan, A. Ng, B. Liu, P. S. Yu, Z.-H. Zhou, M. Steinbach, D. J. Hand, and D. Steinberg, "Top 10 algorithms in data mining," *Knowl. Inf. Syst.*, vol. 14, no. 1, pp. 1–37, 2007.
- [2] I. Guyon, U. von Luxburg, and R. C. Williamson, "Clustering: Science or Art?" *NIPS'09 Workshop on Clustering Theory*, 2009.
- [3] I. Dhillon and D. Modha, "Concept decompositions for large sparse text data using clustering," *Mach. Learn.*, vol. 42, no. 1-2, pp. 143–175, Jan 2001.
- [4] S. Zhong, "Efficient online spherical K-means clustering," in *IEEE IJCNN*, 2005, pp. 3180–3185.
- [5] A. Banerjee, S. Merugu, I. Dhillon, and J. Ghosh, "Clustering with Bregman divergences," *J. Mach. Learn. Res.*, vol. 6, pp. 1705–1749, Oct 2005.
- [6] E. Pekalska, A. Harol, R. P. W. Duin, B. Spillmann, and H. Bunke, "Non-Euclidean or non-metric measures can be informative," in *Structural, Syntactic, and Statistical Pattern Recognition*, ser. LNCS, vol. 4109, 2006, pp. 871–880.
- [7] M. Pelillo, "What is a cluster? Perspectives from game theory," in *Proc. of the NIPS Workshop on Clustering Theory*, 2009.

- [8] D. Lee and J. Lee, "Dynamic dissimilarity measure for support based clustering," *IEEE Trans. on Knowl. and Data Eng.*, vol. 22, no. 6, pp. 900–905, 2010.
- [9] A. Banerjee, I. Dhillon, J. Ghosh, and S. Sra, "Clustering on the unithypersphere using von Mises-Fisher distributions," *J. Mach. Learn. Res.*, vol. 6, pp. 1345–1382, Sep 2005.
- [10] W. Xu, X. Liu, and Y. Gong, "Document clustering based on nonnegative matrix factorization," in *SIGIR*, 2003, pp. 267–273.
- [11] I. S. Dhillon, S. Mallela, and D. S. Modha, "Information-theoretic co-clustering," in *KDD*, 2003, pp. 89–98.
- [12] C. D. Manning, P. Raghavan, and H. Schütze, *An Introduction to Information Retrieval*. Press, Cambridge U., 2009.
- [13] C. Ding, X. He, H. Zha, M. Gu, and H. Simon, "A min-max cut algorithm for graph partitioning and data clustering," in *IEEE ICDM*, 2001, pp. 107–114.
- [14] H. Zha, X. He, C. H. Q. Ding, M. Gu, and H. D. Simon, "Spectral relaxation for k-means clustering," in *NIPS*, 2001, pp. 1057–1064.
- [15] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, pp. 888–905, 2000.

#### AUTHORS PROFILES:



##### 1. N. RaghavaRao

M.Tech(CSE)  
HOD, Associate Professor  
DRK Institute Of Science & Technology



##### 2. K Sravan Kumar

M.Tech Student  
Branch: SE  
DRK Institute Of Science & Technology, Hyderabad.



##### 2. P. Madhu

M.Tech Student  
Branch: CS  
Drk College Of Engineering & Technology, Hyderabad.