ISSN: 2250–3676

[IJESAT] INTERNATIONAL JOURNAL OF ENGINEERING SCIENCE & ADVANCED TECHNOLOGY Volume-2, Issue-3, 508 – 512

CLUSTERING BASED ON COSINE SIMILARITY MEASURE

K.P.N.V.Satya sree[1] , Dr.J V R Murthy[2]

[1]*Assistant Professor, Department of Computer Science, Vignan Nirula Institute of Technology in Science for WOMEN, Guntur, A.P,* ***satyasreekpnv@gmail.com***

[2]*Professor, Department of Computer Science, Jawaharlal Nehru Technological University Kakinada, Kakinada, A.P,* ***mjonnalagedda@gmail.com***

Abstract

All clustering methods have to assume some cluster relationship among the data objects that they are applied on. Hierarchical clustering builds(agglomerative),or breaks up(divisive), a hierarchy of clusters. The traditional representation of this hierarchy is a tree. In this paper, we introduce to develop a novel hierarchal 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. Given a data set satisfying the distribution of a mixture of Gaussians, the degree of overlap between components affects the number of clusters "perceived" by a human operator or detected by a clustering algorithm. In other words, there may be a significant difference between intuitively defined clusters and the true clusters corresponding to the components in the mixture.

Index Terms: *Hierarchical Document clustering, text mining, similarity measure.*

----------***----------

1. INTRODUCTION:

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 paper 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.

2. 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.

2.1 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 oh tf - itf.

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.

3. 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

Simple counting:

This can begin by counting the number of times each of the words appear in each of the documents, Given this simple counting method, searches for "rose" can be sorted by its "term frequency" (TF) — the quotient of the number of times a word appears in each document (C), and the total number of words in the document (T) — TF = C / T. In the first case, rose has a TF value of 0.13. In the second case TF is 0.12, and in the third case it is 0.14. Thus, by this rudimentary analysis, Document 3 is most significant in terms of the word "rose", and Document 2 is the least. Document 3 has the highest percentage of content containing the word "rose".

Accounting for common words

Unfortunately, this simple analysis needs to be offset considering frequently occurring terms across the entire corpus. Good examples are stop words or the word "human" in MEDLINE. Such words are nearly meaningless because they appear so often. Consider the table which includes the number of times each word is found in the entire corpus (DF), and the quotient of the total number of documents (D or in this case, 3) and DF — IDF = D / DF. Words with higher scores are more significant across the entire corpus. Search terms whose IDF ("inverse document frequency") score approach 1 are close to useless because they exist in just about every document:

3.2 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

Table 2: Documents having certain words

3.3 Cosine Similarity Measure

Cosine similarity is a measure of similarity between two vectors of *n* dimensions by finding the cosine of the angle between them, often used to compare documents in text mining. Given two vectors of attributes, *A* and *B*, the cosine similarity, θ, is represented using a dot product and magnitude as

$$\text{similarity} = \cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|}.$$

For text matching, the attribute vectors A and B are usually the tf vectors of the documents. The cosine similarity can be seen as a method of normalizing document length during comparison.

Similarity Measures

The concept of similarity is fundamentally important in almost every scientific field. For example, in mathematics, geometric methods for assessing similarity are used in studies of congruence and homothety as well as in allied fields such as trigonometry. Fuzzy set theory has also developed its own measures of similarity, which find application in areas such as management, medicine and meteorology. An important problem in molecular biology is to measure the sequence similarity of pairs of proteins.

A review or even a listing of all the uses of similarity is impossible. Instead, perceived similarity is focused on. The degree to which people perceive two things as similar fundamentally affects their rational thought and behavior. Negotiations between politicians or corporate executives may be viewed as a process of data collection and assessment of the similarity of hypothesized and real motivators. The appreciation of a fine fragrance can be understood in the same way. Similarity is a core element in achieving an understanding of variables that motivate behavior and mediate affect.

In many experiments people are asked to make direct or indirect judgments about the similarity of pairs of objects. A variety of experimental techniques are used in these studies, but the most common are to ask subjects whether the objects are the same or different, or to ask them to produce a number, between say 1 and 7, that matches their feelings about how similar the objects appear (e.g., with 1 meaning very dissimilar and 7 meaning very similar). The concept of similarity also plays a crucial but less direct role in the modeling of many other psychological tasks. This is especially true in theories of the recognition, identification, and categorization of objects, where a common assumption is that the greater the similarity between a pair of objects, the more likely one will be confused with the other. Similarity also plays a key role in the modeling of preference and liking for products or brands, as well as motivations for product consumption.

4. DESIGN LAYOUT

procedure INTIALIZATION

Select *k* seeds $s_1, \ldots . s_k$ randomly

$Cluster[\, d_i] \leftarrow p \;=\; argmax_r \{s_r^t d_i\}, \forall\, i = 1, \ldots . n$

$D_r \leftarrow \sum_{d_i \in s_r} d_i, n_r \leftarrow \,|\, s_r \,|\,, \forall\, r = 1, \ldots . k$

end procedure

procedure REFINEMENT

repeat

$\{\, v[1 : n]\} \leftarrow$ random permutation of $\{1, \ldots . . , n\}$

for $j \leftarrow 1 : n$ **do** $\; i \;\leftarrow v[\, j\,]$

$p \leftarrow cluster[\, d_i\,]$

$\Delta I_p \;\leftarrow I(\, n_p - \,1, D_p - \, d_i\,) \;-\; I(\, n_p, D_p)$

$q \;\leftarrow argmax_{r, r \neq p} \{\, I(\, n_r + 1, D_r + d_i\,) \;-\; I(\, n_r, D_r)\}$

$\Delta I_q \;\leftarrow I(\, n_q + 1, D_q + d_i\,) \;-\; I(\, n_q, D_q)$

if $\Delta I_p + \Delta I_q > 0$ **then**

Move d_i **to cluster** q: $cluster[d_i] \leftarrow q$

Update D_p , n_p , D_q , n_q

end if

end for

until No move for all *n* documents

end procedure

(1) Initializing the weights parameters.

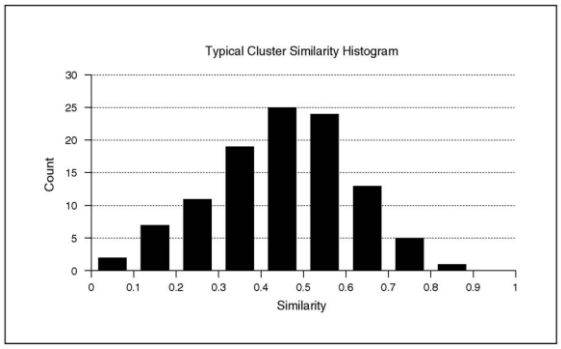
(2) Using the EM algorithm to estimate their means and covariance.

(3) Grouping the data to classes by the value of probability density to each class and calculating the weight of each class.

(4) Repeat the first step until the cluster number reaches the desired number or the largest OLR is smaller than the predefined threshold value. Go to step 3 and output the result. A distinctive element in this algorithm is to use the overlap rate to measure similarity between clusters.

5. PERFORMANCE ANALYSIS

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.



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 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.

In this paper, selecting different dimensional space and frequency levels leads to different accuracy rate in the clustering results. How to extract the features reasonably will be investigated in the future work. There are a number of future research directions to extend and improve this work. One direction that this work might continue on is to improve on the accuracy of similarity calculation between documents by employing different similarity calculation strategies. Although the current scheme proved more accurate than traditional methods, there are still rooms for improvement.

REFERENCES:

1) IEEE Transactions on Knowledge and Engineering, VOL. XX, NO. YY, 2011" "clustering with multi-viewpoint based similarity measure"
2) Cole, A. J. & Wishart, D. (1970). An improved algorithm for the Jardine-Sibson method of generating overlapping clusters. The Computer Journal 13(2):156-163.
3) D'andrade,R. 1978, "U-Statistic Hierarchical Clustering" Psychometrika, 4:58-67.
4) Johnson,S.C. 1967, "Hierarchical Clustering Schemes" Psychometrika, 2:241-254.
5) Shengrui Wang and Haojun Sun. Measuring overlap-Rate for Cluster Merging in a Hierarchical Approach to Color Image Segmentation. International Journal of Fuzzy Systems,Vol.6,No.3,September 2004.
6) Jeff A. Bilmes. A Gentle Tutorial of the EM Algorithm and its Application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models. ICSI TR-97-021, U.C. Berkeley,1998.

BIOGRAPHIES:

K.P.N.V.Satyasree,*Assistant Professor, Department of Computer Science, Vignan Nirula Institute of Technology in Science for WOMEN, Guntur ,A.P,* ***satyasreekpnv@gmail.com***

Dr.J V R Murthy,*Professor, Department of Computer Science, Jawaharlal Nehru Technological University Kakinada, Kakinada, A.P,* ***mjonnalagedda@gmail.com***