Indian Journal of Science and Technology, *Vol 9(48), DOI: 10.17485/ijst/2016/v9i48/105232, December 2016*
ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

Cosine Similarity with Centroid Implication for Text Clustering of Document Files

Anubhuti Singh, Chetna Dabas* and J. P. Gupta

Jaypee Institute of Information and Technology, Noida - 201301, Uttar Pradesh, India;
anubhuti.singh04@gmail.com, cherry.dabas@gmail.com

Abstract

Objectives: To address a pair wise text comparison of large dataset while making use of cosine similarity metric and adjacent method and to develop a model for parallel processing of giant data while using distributed algorithms on parallel clusters. **Methods/Statistical Analysis**: This works makes use of K-means algorithm based on map-reduce on document files with effective number of clusters in a Java environment. This work reflects an approach to classify text documents using feature selection method makes use of cosine similarity method. Within fixed number of iterations, efficient numbers of clusters have been implemented. The implementation has been carried out in Java environment. **Findings**: The proposed work reflects an approach to classify text documents using feature selection method. **Application/Improvements:** While using cosine similarity methods, the results retrieved are quite improved and acceptable.

Keywords: Cosine Similarity, Document Files, Text Clustering

1. Introduction

Nowadays the big data has become very challenging and specifically in various domains like computer science and pharmaceutical sciences[1–4]. Each day terabytes of data in form of semi structured, unstructured and structured forms is generated (especially text data) via various sources such as humans, machines or otherwise. This research paper addresses an advanced technique (resemblance of text). Big data if used properly can bring huge benefits to the business, science and humanity. The various properties of big data like volume, velocity, variety, variation and veracity render the existing techniques of data analysis ineffective. Big data analysis needs fusion of techniques for data mining with those of machine learning. The k-means algorithm is one such algorithm which has presence in both the fields. The proposed work carries implementation of a k means hybrid approximation algorithm. The simulation work is carried out in Java environment.

In recent years a lot of attention has been focussed on data clustering techniques.In proposed an algorithm that is intended to provide efficient text categorization technique using the text summarization technique and cluster analysis technique. A hybrid text clustering technique is developed for categorizing the text in a given domain. In text mining and text categorization, the resource consumption is a major issue; therefore feature extraction technique is utilized to reduce the amount of text. This reduced text represents the whole text document. Additionally, the k-mean clustering algorithm's Euclidian distance based approach is utilizedfor finding similarity. İn focused on the text classification method from magnamous set of documents. Information Retrieval (IR) and Machine Learning (ML) are used in this technique for text classification. In projected text classification with both SVM and KNN algorithms. The SVM-KNN algorithm can advance the act of classifier by the reaction and enhancement of organizing guess probability. The real effect of SVM-KNN algorithm is verified in associated Chinese web page arrangement test system[4–8]. An author considered high dimensional data into consideration in the similar context. In recent years an author proposed partition-based clustering method. K-mean firstly initial-

**Author for correspondence*

izes the center and then calculates the distance of another element creating the cluster of those elements whose distance is very less from the center. K-mean is a simple, flexible and easy to understand and implements which works for numeric dataset. In 2014, proposed the method which divides the square root distance with standard deviation resulting into enhanced k-mean performing much better than that of k-methods and k-mean and it takes less time to formulate the clusters. Moreover it is not only applicable for small dataset but also works very efficiently for very large datasets[9–13].

2. Background

As per the literature requirements, the cosine similarity and k-neighboring method are chosen.

The next subsections throw a little light on these techniques:

2.1 Cosine Similarity Technique

It is the similarity measure between two vectors (or two documents on the Vector Space). It calculates the cosine angle between two documents. This measure is the evaluation for the measurement of orientation. This technique calculates the angle between documents not the magnitude of the documents. Cosine similarity can be represented.

$$\vec{a}.\vec{b} = ||\vec{a}||\ ||\vec{b}||\cos Q$$

2.2 K-Neighboring Technique

K-nearest neighbour is a classification algorithm that combines the k nearest points. It is supervised classification algorithm. It is very simple and relatively high convergence speed algorithm. However, in some applications, it may fail to produce adequate results, whilst in others its operation may render impractical. Yet, the fact that it has only one parameter, the number of neighbours used (k), makes it easy to fine tune to a variety of situations. Its chief procedure entails of the following steps: considering a data set consisting up of N points, then classifying these N points into identical cohesive groups by clustering k closest points.

3. Method

A method is addressed in the presented work for calculating text similarity smearing cosine method that contains of 3 phases. At the first phase, the counting of occurrence of each word in the documents is done. Afterwards, in each document, for every one term, the measurement of term frequency is done. Lastly, in order to predict the similarity, the cosines of the pairs are framed. The pseudo code and detailed analysis is presented.

Pseudocode

```
Initialize= docn
for each element ∈( termnew , docn) where n = 1, 2, .......n1
write ( docId, ( termnew, o ) )
Find doc length
X= 0
for everytuple ∈( termnew, o ) do
X= X + o, and match doc length
If matches then
return ( (docn, X), (termnew, o) )
End
```

Cosine similarity is implemented and the whole working model is shown in Figure 1:

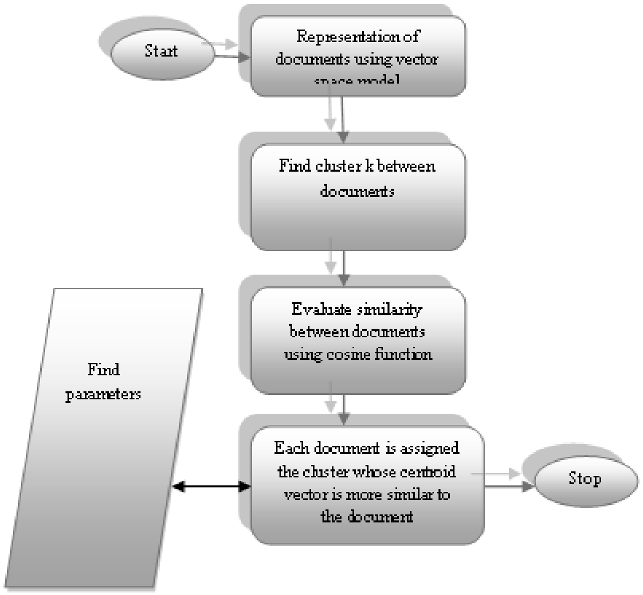
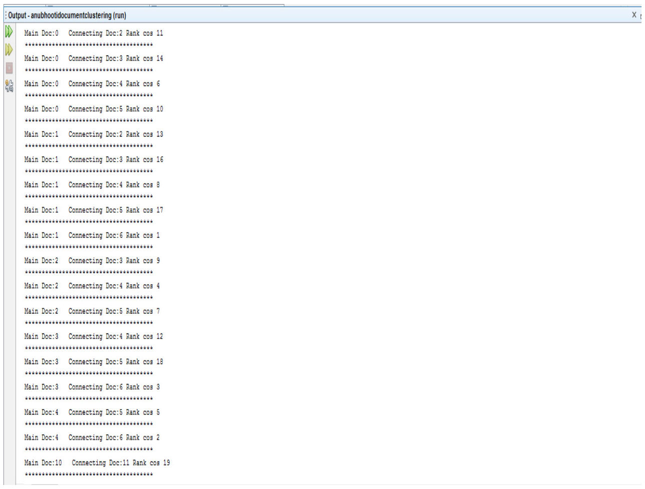

Start
Representation of documents using vector space model
Find cluster k between documents
Evaluate similarity between documents using cosine function
Find parameters
Each document is assigned the cluster whose centroid vector is more similar to the document
Stop

Figure.1 Model for text document clustering

4. Results and Discussion

Data clustering is considered as an importance technique especially for document browsers and search engines. For operators, it facilitates a decent accurate and complete interpretation of the data enclosed in the data sets. The well-known approaches of data clustering, not only report the singular problems of clustering whereas it faces the problem of large dimensionality of the dataset, very vast choice of the datasets and ability to understand the cluster description. The previous algorithms do not have the inbuilt property of clustering though it has the ability to extract features from the document. In the proposed algorithm we have taken different classes of the documents and tried to make clusters using cosine similarity technique on neighbors in addition to Centroid method. The implementation been carried out in Java Environment with MyEclipse Enterprise Workbench Version 8.0M1 with computing system with RAM 4 GB and windows (Version 8.1) 64 bit OS.

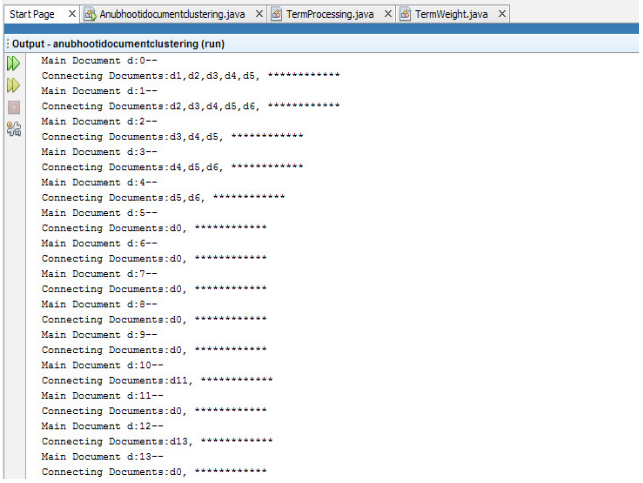


```
Main Doc:0   Connecting Doc:2 Rank cos 11
**************************************
Main Doc:0   Connecting Doc:3 Rank cos 14
**************************************
Main Doc:0   Connecting Doc:4 Rank cos 6
**************************************
Main Doc:0   Connecting Doc:5 Rank cos 10
**************************************
Main Doc:1   Connecting Doc:2 Rank cos 13
**************************************
Main Doc:1   Connecting Doc:3 Rank cos 16
**************************************
Main Doc:1   Connecting Doc:4 Rank cos 8
**************************************
Main Doc:1   Connecting Doc:5 Rank cos 17
**************************************
Main Doc:1   Connecting Doc:6 Rank cos 1
**************************************
Main Doc:2   Connecting Doc:3 Rank cos 9
**************************************
Main Doc:2   Connecting Doc:4 Rank cos 4
**************************************
Main Doc:2   Connecting Doc:5 Rank cos 7
**************************************
Main Doc:3   Connecting Doc:4 Rank cos 12
**************************************
Main Doc:3   Connecting Doc:5 Rank cos 18
**************************************
Main Doc:3   Connecting Doc:6 Rank cos 3
**************************************
Main Doc:4   Connecting Doc:5 Rank cos 5
**************************************
Main Doc:4   Connecting Doc:6 Rank cos 2
**************************************
Main Doc:10   Connecting Doc:11 Rank cos 19
**************************************
```

Figure 2. Snapshot of cluster formation of main documents and connecting documents.

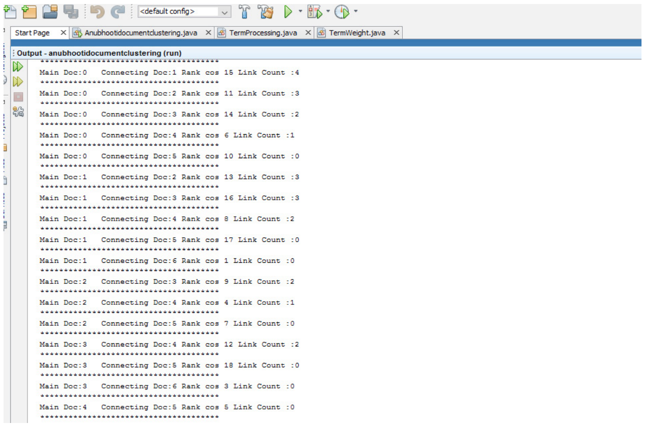
Figure 2 depicts the snapshot cluster formation of main and connecting documents on the computing machine in Java implementation environment. K-means algorithm is popular because of its simplicity for execution and competence to harvest important results. In K-means, we divide dataset into pre-defined k partitions where each partition is recognized as an individual distinct cluster which has the minimum ground on the intra cluster distance-mean algorithm of partition and it will labor rendering k number of clusters at the period of input.

In the presented work, estimation of Centroid is performed followed up with the allocation of points in the dataset to the specific partition with the nearest centroid is done. Then, the new Centroid for each partition is computed, while looking at similar documents. As a part of this process, while keeping into consideration the distances of objects from the fresh centroids, the data objects in gets reshuffled. Error function is evaluated at each step. Fault function performs a crucial role here and it checked weather the fresh centroids contains lesser value in comparison with prior centroids, if found so, then the new fresh Centroid would be saved otherwise the measure track gets updated. The code to carry out implementation was consisted primarily of 3 classes corresponding to document clustering (responsible for conduction actual clustering and invoking the term weight and term processing classes), term weight (responsible for computation of computing weight corresponding to each document) and term processing (responsible for computation of main and connecting document).


Start Page
Anubhootidocumentclustering.java
TermProcessing.java
TermWeight.java
Output - anubhootidocumentclustering (run)

```
Main Document d:0--
Connecting Documents:d1,d2,d3,d4,d5, ************
Main Document d:1--
Connecting Documents:d2,d3,d4,d5,d6, ************
Main Document d:2--
Connecting Documents:d3,d4,d5, ************
Main Document d:3--
Connecting Documents:d4,d5,d6, ************
Main Document d:4--
Connecting Documents:d5,d6, ************
Main Document d:5--
Connecting Documents:d0, ************
Main Document d:6--
Connecting Documents:d0, ************
Main Document d:7--
Connecting Documents:d0, ************
Main Document d:8--
Connecting Documents:d0, ************
Main Document d:9--
Connecting Documents:d0, ************
Main Document d:10--
Connecting Documents:d11, ************
Main Document d:11--
Connecting Documents:d0, ************
Main Document d:12--
Connecting Documents:d13, ************
Main Document d:13--
Connecting Documents:d0, ************
```

Figure 3. Snapshot of application of addressed method for clustering.


<default config>
Start Page
Anubhootidocumentclustering.java
TermProcessing.java
TermWeight.java
Output - anubhootidocumentclustering (run)

```
**************************************
Main Doc:0   Connecting Doc:1 Rank cos 15 Link Count :4
**************************************
Main Doc:0   Connecting Doc:2 Rank cos 11 Link Count :3
**************************************
Main Doc:0   Connecting Doc:3 Rank cos 14 Link Count :2
**************************************
Main Doc:0   Connecting Doc:4 Rank cos 6 Link Count :1
**************************************
Main Doc:0   Connecting Doc:5 Rank cos 10 Link Count :0
**************************************
Main Doc:1   Connecting Doc:2 Rank cos 13 Link Count :3
**************************************
Main Doc:1   Connecting Doc:3 Rank cos 16 Link Count :3
**************************************
Main Doc:1   Connecting Doc:4 Rank cos 8 Link Count :2
**************************************
Main Doc:1   Connecting Doc:5 Rank cos 17 Link Count :0
**************************************
Main Doc:1   Connecting Doc:6 Rank cos 1 Link Count :0
**************************************
Main Doc:2   Connecting Doc:3 Rank cos 9 Link Count :2
**************************************
Main Doc:2   Connecting Doc:4 Rank cos 4 Link Count :1
**************************************
Main Doc:2   Connecting Doc:5 Rank cos 7 Link Count :0
**************************************
Main Doc:3   Connecting Doc:4 Rank cos 12 Link Count :2
**************************************
Main Doc:3   Connecting Doc:5 Rank cos 18 Link Count :0
**************************************
Main Doc:3   Connecting Doc:6 Rank cos 3 Link Count :0
**************************************
Main Doc:4   Connecting Doc:5 Rank cos 5 Link Count :0
**************************************
```

Figure 4. Snapshot of main document with connecting document.

Figure 3 shows the formation of clusters between the main document and connecting document on the basis of cosine similarity. For 13 documents, 13 clusters are being made.

Figure 4 illustrates the computation of the rank and link counts on the basis of the cosine similarity between the main and the connecting documents.

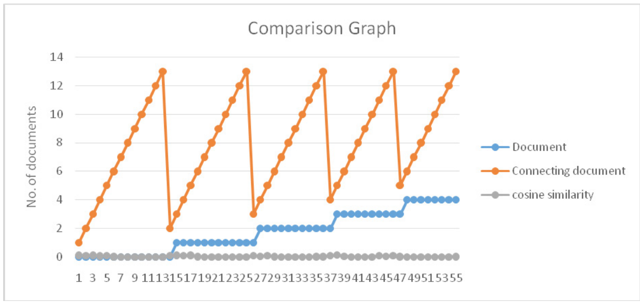


Figure 5. Comparison between cosine values of main document and connecting document.

Figure 5 shows the different cosine values for different documents ranging from 1 to 13 documents. The graph illustrates the number of connecting documents attached to the main document on the basis of the cosine similarity. For instance, From document 13 to 25, the connecting documents is increasing rapidly ranging from 2 to about approximately above 12 (number of documents) each having assigned the cosine similarity of 13 to 25. With the considered approach for clustering data it was observed that initially the number of clusters formed were four (initialized) which reduced to two at last.

5. Conclusions

In the proposed work, normal text ordering has been done in view of text classification and text-based genre. Here simulations are carried out to classify documents using centroid selection method. From implementation results it is concluded that using cosine similarity method results are quite acceptable. These results may play important role while gaining improved accuracy in classifying documents used in diverse areas and streams across the globe.

6. References

1. Yao C, Zhang X, Bai X, Liu W, Ma Y, Tu Z. Rotation-invariant features for multi-oriented text detection in natural images. PloS one. 2013; 8(8):1–28.
2. Elberrichi Z, Rahmoun A, Bentaalah MA. Using WordNet for text categorization. The International Arab Journal of Information Technology. 2008; 5(1):16–26.
3. Wang Q, Garrity GM, Tiedje JM, James R, Cole C. Naive bayesian classifier for rapid assignment of rrna sequences into the new bacterial taxonomy. Applied and Environmental Microbiology. 2007; 73(16):5261–7.
4. Kazmierska J, Malicki J. Application of the naive bayesian classifier to optimize treatment decisions. Radiotherapy and Oncology. 2008; 86(2):211–6.
5. Daniel RM, Shukla AK. Improving text search process using text document clustering approach. IJSR. 2014; 3(3):14–24. ISSN: 2319-7064.
6. Lin L. Research on text classification based on SVM-KNN. 5th IEEE International Conference on Software Engineering and Service Science (ICSESS); China. 2014. p. 842–4.
7. Vishwanath V. Machine learning approach for text and document mining. Computer Science. 2014; 7(1):41–8.
8. Raghuveer K, Murthy KN. Text categorization in Indian languages using machine learning approaches. IICAI; 2007. p. 1864–83.
9. Anuradha A. Neural network approach for text classification using relevance factor as term weighing method. IJCA Journal. 2013; 68(17):37–41.
10. Vora P. A survey on K-mean clustering and particle swarm optimization. International Journal of Science and Modern Engineering. 2013; 1(3):1–14.
11. Li M, Chen X, Li X, Ma B, Vitanyi PMB. The similarity metric. IEEE Transactions on Information Theory. 2004; 50(12):3250–64.
12. Li B, Lu Q, Yu S. An adaptive k-nearest neighbor text categorization strategy. ACM Transactions on Asian Language Information Processing (TALIP). 2004; 3(4):215–26.
13. Nigam G, Dabas C. Effective compressive sensing for clustering in wireless sensor networks. Indian Journal of Science and Technology. 2016; 9(38):1–5.