Himanshu Gupta *et al*, International Journal of Computer Science and Mobile Applications,
Vol.2 Issue. 5, May- 2014, pg. 7-13 **ISSN: 2321-8363**

k-means Based Document Clustering with Automatic "k" Selection and Cluster Refinement

Himanshu Gupta, Dr. Rajeev Srivastava

Department of Computer Engineering
Indian Institute of Technology, BHU (INDIA)
himanshu.gupta.cse09@itbhu.ac.in
rajeev.cse@iitbhu.ac.in

Abstract— ***In recent years use of web has been increased manifold. Efficiency is as important as accuracy. Automatic document clustering is an important part of many important fields such as data mining, information retrieval etc. Most of the document clustering techniques are based on k-means and it's variants. K-means is a fast algorithm but there are some shortcomings with this technique. K in k-means stands for no of clusters which a user has to provide but most of the times users don't have any clue about k. In our implementation of document clustering technique we used SVD (Singular Vector Decomposition) to find out no of clusters (value of k) required. Then k-means algorithm is used to create clusters and in last phase of algorithm the clusters are refined by feature voting. Refinement phase enable us to make our algorithm much faster than k-means algorithm.***

Keywords— ***Document Clustering, k-means, Feature Voting, SVD, Vector Space Model, Cosine similarity***

I. INTRODUCTION

Clustering is an unsupervised learning technique which is used to group a set of objects into subsets or clusters. The goal is to group these objects such that objects in the same clusters should be as similar as possible and objects in different clusters must be as dissimilar as possible[1,2,3]**.** Document clustering is used to organize text documents which are useful for information retrieval, data mining. There are two types of clustering techniques: hierarchical and partition [2]. Hierarchical techniques are known for creating better quality clusters but they are relatively slow. Time complexity of this approach is quadratic. Most of the widely used partition techniques are k-means and it's variants[4]. Time complexity of partition techniques is almost linear. K-means clustering algorithm clusters data into predefined no of clusters. It starts with initialization of cluster centroids randomly then assignment of data objects to the closest (most similar) centroid. This process is repeated again and again until a termination criterion (either after certain no of iteration clusters don't change or predefined no of iteration have been done) is met.

The drawbacks of k-means are user have to provide the value of k which most probably he does not have any clue about. Another drawback is that cluster results are sensitive to initialization of centroids. So if centroids are not initialized properly then results may converge to local optima[5].

These problems are addressed in out implementation. Value of k is decided using Singular Vector Decomposition(SVD). SVD is applied on term-document matrix and k-rank approximation is done on $\sum$ matrix which we get from SVD. Which gives us value of k. Then k-means is applied. In k-means, initialization of centroids is done using roulette wheel selection method. Due to using SVD the process is slowed down which is compensated by refinement. K-means is only run for 200 rounds to get initial clusters.

Once initial clusters are obtained from k-means refinement is done using feature voting. Due to the fact that refinement step is only comprised of matrix accessing and searching, our method is very efficient.

II. PRELIMINARIES

A. VectorSpaceModel:

Set of text documents is represented as Vector Space Model (VSM)[6]. VSM can be represented as $V=\{x_1,x_2,....x_n\}$. Each document x_i is represented as a vector which is called the feature vector. A vector x can be represented as, $x= \{w_1,w_2,.....,w_n\}$. Where w_i represents the term weight. The term weight can be calculated using TF-IDF (term frequency-inverse document frequency) scheme. Weight of i in document j can be calculated as:

$$W_{ji} = tf_{ji} * idf_{ji} = tf*log_2(n /df_{ji}) \qquad (1)$$

Where tf_{ji} is the number of times term i has appeared in document j. df_{ji} represents the no of documents in which term i has appeared and n is the total no of documents in collection. This Vector Space Model can also be seen as term document matrix of txd where t is total no of terms and d is no of documents.

B. Similarity Metric

Cosine correlation is used as similarity metric. As the documents are represented as feature vectors cosine similarity is a very good metric to measure similarity. If there are two vectors A and B then cosine correlation[7] can be given as:

$$\text{Cos(A,B)} = \frac{\text{A.B}}{|\text{A}|*|\text{B}|} \qquad (2)$$

Where A.B denotes the dot product and |A| denotes the length of the vector.

C. Singular Vector Decomposition (SVD)

The decomposition breaks a $t\times d$ matrix A into three matrices U, Σ and V such that $A=U\Sigma V^T$. U is the $t\times t$ orthogonal matrix whose column vectors are called the left singular vectors of A, V is the $d\times d$ orthogonal matrix whose column vectors are termed the right singular vectors of A, and Σ is the $t\times d$ diagonal matrix having the singular values of A ordered decreasingly ($\sigma_1 \geq \sigma_2 \geq ... \geq \sigma_{min(t,d)}$) along its diagonal. The rank r_A of the matrix A is equal to the number of its non- zero singular values. In our algorithm we will use the diagonal matrix of singular values of A[8].

III. METHODOLOGY

The algorithm has four phases which are following

A. Term-Document matrix building:

In this phase a term document matrix A is build so that SVD decomposition can be performed on A. This matrix can be seen as vector space model in which each vector is a column of this matrix.

B. Calculation of K in K-Means

In this step first of all SVD decomposition of term-document matrix is performed. To calculate the value of k, a method is presented below which is based on the Frobenius norm of the term document matrix and it's k-rank approximation. The primary motivation for our choice

was the fact that the other approaches cannot be easily parameterized. In this method a quality threshold q is also assumed which help in determining the extent the k rank approximation should retain the original information[8].

k is set to the value which satisfies the following inequality:

$$\frac{||A_k||_F}{||A||_F} = \frac{\sqrt{\sum_{i=1}^{k} \sigma_i^2}}{\sqrt{\sum_{i=1}^{r_A} \sigma_i^2}} \geq q \qquad (3)$$

In this formula, A is original matrix, A_k is it's k-rank approximation, r_A denotes the rank of A. , σ_i is its ith singular value (the ith diagonal element of the SVD's Σ matrix) and $||A||_F$ denotes the Frobenius norm of the A matrix. q is quality threshold whose value is between 0.7 to 0.9 (default .775).

C. Apply K-means on Document Collection:

Once the value of k is determined we apply k-means clustering algorithm to get initial clusters. Cosine similarity is used as similarity metric. Given data objects and value of k, basic K-means algorithm is given below:

1. Choose initial centroid vectors.

2. Assign each document vector to closest cluster centroid.

3. Recalculate the cluster centroid c_j using following formula:

$$c_j = \frac{1}{n_j} \sum_{\forall d_i \in S_j} d_i \qquad (4)$$

Where d_i denotes the document vector which belongs to cluster S_j,c_j stands for centroid vector and n_j is the total number of documents belong to cluster S_j.

4. Repeat until termination criterion is met.

Here termination criterion is 200 iteration.

D. Refinement of Clusters:

Once we get clusters from third phase we can apply refinement on these clusters. Refinement is done using feature voting[9]. When the discriminative feature f_i has the highest occurrence frequency in cluster c_x, we say that f_i is discriminative for c_x, and save the cluster label x for f_i (denoted as s_i) for the later feature voting operation. By definition, s_i can be expressed as:

$$s_i = \max_x \; g(f_i, c_x) \qquad (5)$$

Where g is function which find frequency of feature *fi* in cluster c_x .

Once the discreminative features have been selected following voting scheme is applied iteratively to refine document clusters:

1. Inherit initial cluster set C= {c_1, c_2,…….. c_k} using k-means.

2. Use C to find the descriminative feature set F={f_1, f_2,…. fp} with their respective cluster lables S={s_1, s_2,…. S_p}.

3. For each document d_j in document collection find it's new cluster lable l_j by the majority voting of descriminative features as following:

Suppose document d_j contains some descriminative features

$F^j = \{f_1^j, f_2^j \dots f_n^j\} \in F$ and $S^j = \{s_1^j, s_2^j \dots s_n^j\}$

is associated cluster lables. Then new cluster lable for document dj will be decided as:

$$newlable_j = \max_{s_y \in S^j} count\left(s_y, S^j\right) \in F \qquad (6)$$

Where count (s_y, S^j) represents the number of times s_y appear in S^j.

4. Compare new clusters set with previous cluster set. If result converges then terminate the process else update new cluster set as C and goto step 2.

Above voting process is self-refinement iterative process. In starting we have initial set of document clusters with low accuracy. From initial clusters process finds features that are discriminatvie for the clusters and then by voting on each document's cluster label refinement is done. Due to this process the document clusters slowly improved and clustered accordingly.

In this paper following method is proposed:

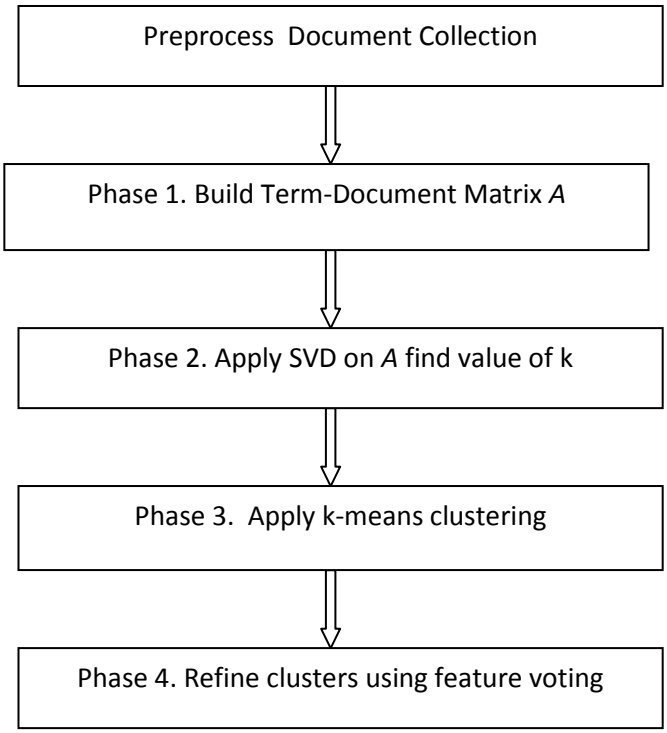

Figure 1. Proposed method

IV. EXPERIMENTAL SETUP

Algorithm implemented in Java language. Netbeans IDE is used. For SVD decomposition JAMA MATRIX [10] package is used. "Reuters-21578-topics" is used to create different datasets in which there were 10 classes. Documents were pre-processed by removing stop words and different forms of a word are reduced to single word using porter's stemmer [11].

Table 1: summary of text document collection

Dataset	Number of documents	Number of classes
D1	25	5
D2	35	6
D3	45	7
D4	55	7
D5	65	7

V. RESULT & DISCUSSION

A. Finding value of k: Following table shows the number of classes which are used to make datasets And the number of clusters suggested by our method

Table 2

Dataset	Number of clusters suggested by method	Number of classes
D1	5	5
D2	6	6
D3	6	7
D4	7	7
D5	6	7

From table 2 it is clear that our method is quite accurate in predicting number of clusters. And it could be brought closer by tuning candidate level threshold. And it's also represent that the documents which belong to different classes may be similar that's why there may not be need of that much clusters. However, method is useful to make cluster algorithms automatic in choosing value of k.

B. Efficiency of method: For measuring the efficiency of the algorithm, algorithm is compared with the pure k-means algorithm. In our algorithm first k-means is applied for maximum of 200 iterations and then refinement is performed. In pure k-means algorithm 600 iterations are used. Both the algorithms were almost equal from accuracy's point of view. Table 3 represents the comparison on the basis of efficiency.

Table 3: comparison between k-means & proposed method

Dataset	Time taken by pure k-means(in sec)	Time taken by our algorithm (in sec)
D1	57.10	26.93
D2	28.34	18.76
D3	64.50	37.73
D4	130.05	88.79
D5	135.83	91.95

It's clear from above results that our algorithm has significant advantages over k-means algorithm in terms of efficiency. This is because of the fact that in refinement step of our algorithm major operations which are done were accessing the matrix values and addition and searching.

C. Accuracy of method: Following table shows the accuracy of the algorithm. Overall cosine similarity is used as the criterion for accuracy. Cosine similarity before refinement and after refinement is compared.

Table 4 : accuracy before and after refinement

Dataset	Cosine similarity before refinement	Cosine similarity after refinement
D1	6.419	9.703
D2	7.148	10.059
D3	10.952	15.283
D4	17.829	20.592
D5	18.984	22.093

Above results show that accuracy is significantly increased in our algorithm. About 25-40% increase in accuracy is measured.

CONCLUSIONS

The method represented in this paper computes the value of k automatically and with great precession. By using refinement by feature voting increase the efficiency greatly. Method is both efficient and accurate and creates good quality clusters.

REFERENCES

1. Berkhin, P., 2002. Survey of clustering data mining techniques. Accrue Software Research Paper.
2. Jain A. K., Murty M. N., and Flynn P. J., 1999, Data Clustering: A Review, ACM Computing Survey, Vol. 31, No. 3, pp. 264-323.
3. Steinbach M., Karypis G., Kumar V., 2000. A Comparison of Document Clustering Techniques. TextMining Workshop, KDD
4. Hartigan, J. A. 1975. Clustering Algorithms. John Wiley and Sons, Inc., New York, NY.
5. Selim, S. Z. And Ismail, M. A. 1984. K-means type algorithms: A generalized convergence theorem and characterization of local optimality.
6. Everitt, B., 1980. Cluster Analysis. 2nd Edition. Halsted Press, New York
7. Salton G. and Buckley C., 1988. Term-weighting approaches in automatic text retrieval. Information Processing and Management, 24 (5): pp. 513-523.
8. Lingo: Search Results Clustering Algorithm Based on Singular Value Decomposition Stanis law Osi´nski, Jerzy Stefanowski, and Dawid Weiss
9. Document Clustering with Cluster Refinement and ModelSelection Capabilities, Xin Liu, Yihong Gong, Wei Xu and Shenghuo Zhu.
10. The Java Matrix Package: http://math.nist.gov/javanumerics/jama
11. Porter, M.F., 1980. An Algorithm for Suffix Stripping. Program

