Analysis of Stemming Algorithm for Text Clustering

N. Sandhya[1], Y. Sri Lalitha[2] , V.Sowmya[3] , Dr. K. Anuradha[4] and Dr. A. Govardhan[5]

[1] Associate Professor, CSE Department, Gokaraju Rangaraju Institute of Engineering and technology, Hyderabad, Andhra Pradesh, 500 072, India

[2] Associate Professor, CSE Department, Gokaraju Rangaraju Institute of Engineering and technology, Hyderabad, Andhra Pradesh, 500 072, India

[3] Associate Professor, CSE Department, Gokaraju Rangaraju Institute of Engineering and technology, Hyderabad, Andhra Pradesh, 500 072, India

[4]Professor and Head, CSE Department, Gokaraju Rangaraju Institute of Engineering and technology, Hyderbad, Andhra Pradesh, 500 072, India

[5] Principal, JNTUH College of Engineering Jagityal, Andhra Pradesh, 500 501, India

Abstract

Text document clustering plays an important role in providing intuitive navigation and browsing mechanisms by organizing large amounts of information into a small number of meaningful clusters. In Bag of words representation of documents the words that appear in documents often have many morphological variants and in most cases, morphological variants of words have similar semantic interpretations and can be considered as equivalent for the purpose of clustering applications. For this reason, a number of *stemming Algorithms*, or *stemmers*, have been developed, which attempt to reduce a word to its *stem* or root form. Thus, the key terms of a document are represented by stems rather than by the original words. In this work we have studied the impact of stemming algorithm along with four popular similarity measures (Euclidean, cosine, Pearson correlation and extended Jaccard) in conjunction with different types of vector representation (boolean, term frequency and term frequency and inverse document frequency) on cluster quality. For Clustering documents we have used partitional based clustering technique K Means.

Performance is measured against a human-imposed classification of Classic data set. We conducted a number of experiments and used entropy measure to assure statistical significance of results. Cosine, Pearson correlation and extended Jaccard similarities emerge as the best measures to capture human categorization behavior, while Euclidean measures perform poor. After applying the Stemming algorithm Euclidean measure shows little improvement.

Keywords: *Text clustering, Stemming Algorithm, Similarity Measures, Cluster Accuracy.*

1. Introduction

With ever increasing volume of text documents, the abundant texts flowing over the Internet, huge collections of documents in digital libraries and repositories, and digitized personal information such as blog articles and emails are piling up quickly every day. For text documents, clustering has proven to be an effective approach and an interesting research problem. Clustering of text documents plays a vital role in efficient Document Organization, Summarization, Topic Extraction and Information Retrieval. Initially used for improving the precision or recall in an Information Retrieval System [1,2], more recently, clustering has been proposed for use in browsing a collection of documents [3] or in organizing the results returned by a search engine in response to user's query [4] or help users quickly identify and focus on the relevant set of results. Customer comments are clustered in many online stores, such as Amazon.com to provide collaborative recommendations. In collaborative bookmarking or tagging, clusters of users that share certain

traits are identified by their annotations. Document clustering has also been used to automatically generate Hierarchical clusters of documents [5]. The automatic generation of taxonomy of Web documents as the one provided by Yahoo! (www.yahoo.com) is often cited as a goal.

This paper is organized as follows. Section 2 describes the document representation used in the experiments, section 3 deals with the related work in finding stem of a word and an insight into clustering algorithms, Section 4 discusses the similarity measures and their semantics. Section 5 presents the K-means clustering algorithm and Section 6 explains experiment settings, evaluation approaches, results and analysis and Section 7 concludes and discusses future work.

2. Document Representation

The representation of a set of documents as vectors in a common vector space is known as the vector space model. Despite of its simple data structure without using any explicit semantic information, the vector space model enables very efficient analysis of huge document collections. The vector space model represents documents as vectors in *m*-dimensional space, i.e. each document *d* is described by a numerical vector of terms. Thus, documents can be compared by use of simple vector operations.

There are three document encoding methods namely, *Boolean, Term Frequency and Term Frequency with Inverse Document Frequency.*

The simplest document encoding is to use binary term vectors, i.e. a vector element is set to one if the corresponding word is used in the document and to zero if the word is not. Using Boolean encoding the importance of all terms is considered as similar. To improve the performance, term weighting schemes are used, where the weights reflect the importance of a word in a specific document of the considered collection. Large weights are assigned to a term that are used frequently in relevant documents but rarely in the whole document collection [11] and is represented by the term frequency (TF) vector:

$$d_{tf} = [tf_1, tf_2, \ldots, tf_D] \quad (1)$$

Where, tf_i is the frequency of term i in the document, and D is the total number of unique terms in the text database.

Terms that occur in few documents are helpful to discriminate the documents from the rest of the collection. The inverse document frequency term weighting is used to assign higher weights to the more discriminative words. IDF is defined via the fraction N/n_i, where, N is the total number of documents in the collection and n_i is the number of documents in which term i occurs.

Due to the large number of documents in many collections, this measure is usually squashed with a log function. The resulting definition IDF is thus:

$$\mathrm{idf}_i = \log\left(\frac{N}{n_i}\right) \quad (2)$$

Combining term frequency with IDF results in a scheme known as tf-idf weighting.

$$w_{i,j} = tf_{i,j} \times \mathrm{idf}_i \quad (3)$$

Thus, the tf–idf representation of the document d is:

$$d_{tf-idf} = [tf_1 \log(n/df_1), tf_2 \log(n/df_2), \ldots, tf_D \log(n/df_D)] \quad (4)$$

To account for the documents of different lengths, each document vector is normalized to a unit vector (i.e., $\|d_{\text{df-idf}}\|=1$). In the rest of this paper, we assume that this vector space model is used to represent documents during the clustering. Given a set C_j of documents and their corresponding vector representations, the centroid vector c_j is defined as:

$$c_j = \frac{1}{|C_j|} \sum_{d_i \in C_j} d_i \quad (5)$$

where each d_i is the document vector in the set C_j, and j is the number of documents in Cluster C_j. It should be noted that even though each document vector d_i is of unit length, the centroid vector c_j is not necessarily of unit length. In this paper we experimented with all the three representations of Vector Space Model (VSM).

3. Related Work

In Bag of words representation of documents the words that appear in documents often have many morphological variants and in most cases, morphological variants of words have similar semantic interpretations and can be considered as equivalent for the purpose of clustering applications. For this reason, a number of *stemming Algorithms*, or *stemmers*, have been developed, which attempt to reduce a word to its *stem* or root form. Thus, the key terms of a document are represented by stems rather than by the original words.

Stemming refers to the process of removing affixes (prefixes and suffixes) from words. In the information retrieval context, stemming is used to conflate word forms to avoid mismatches that may undermine recall. As a simple example, consider searching for a document entitled “How to write”. If the user issues the query “writing” there will be no match with the title. However, if

the query is stemmed, so that "writing" becomes "write", then retrieval will be successful. In many languages stemming increases the number of documents retrieved by between 10 and 50 times. Nonetheless, stemming has shown to produce reliable retrieval improvement [15]. Furthermore, affixes often carry information such as part of speech, plurality, and/or tense that is crucial for the development of more sophisticated information systems. For efficient clustering of related documents we require a high precision stemmer as a preprocessing step [12].

The most widely cited stemming algorithm was introduced by Porter (1980). The Porter stemmer applies a set of rules to iteratively remove suffixes from a word until none of the rules apply. The Porter stemmer has a number of well-documented limitations. The words like "fisher", "fishing", "fished", etc. gets reduced to it's stem word "fish". The Porter stemmer follows a strategy of suffix stripping. Like many existing stemmers it ignores prefixes completely, so "reliability" and "unreliability" remain as unrelated tokens. The Lovins stemmer [16] is similar in mechanism but has a larger set of suffixes (each of which may include multiple morphemes) and does not apply its rules iteratively. While it tends to be more conservative than the Porter stemmer still suffers from over conflation and non-word stems.

Many clustering techniques have been proposed in the literature. Clustering algorithms are mainly categorized into Hierarchical and Partitioning methods [2, 3, 4, 5]. Hierarchical clustering method works by grouping data objects into a tree of clusters [6]. These methods can further be classified into agglomerative and divisive Hierarchical clustering depending on whether the Hierarchical decomposition is formed in a bottom-up or top-down fashion. K-means and its variants [7, 8, 9] are the most well-known partitioning methods [10].

Hierarchical clustering is often portrayed as the better quality clustering approach, but is limited because of its quadratic time complexity. In contrast, K-means and its variants have a time complexity which is linear in the number of documents, but are thought to produce inferior clusters.

There are a number of Partitional techniques, but we shall only describe the K-means algorithm which is widely used in document clustering. K-means is based on the idea that a center point can represent a cluster. In particular, for K-means we use the notion of a centroid, which is the mean or median point of a group of points. Note that a centroid almost never corresponds to an actual data point. The algorithm is discussed in detail in section 5.

4. Similarity Measures

Document clustering groups similar documents to form a coherent cluster. However, the definition of a pair of documents being similar or different is not always clear and normally varies with the actual problem setting. For example, when clustering research papers, two documents are regarded as similar if they share similar thematic topics. When clustering is employed on web sites, we are usually more interested in clustering the component pages according to the type of information that is presented in the page. For instance, when dealing with universities web sites, we may want to separate professor's home pages from student's home pages, and pages for courses from pages for research projects. This kind of clustering benefits further analysis and utilize the dataset such as information retrieval and information extraction, by grouping similar types of information sources together.

Accurate clustering requires a precise definition of the closeness between a pair of objects, in terms of either the pair wise similarity or distance. A variety of similarity or distance measures have been proposed and widely applied, such as cosine similarity, Jaccard coefficient, Euclidean distance and Pearson Correlation Coefficient.

4.1 Cosine Similarity Measure

For document clustering, there are different similarity measures available. The most commonly used is the cosine function. For two documents d_i and d_j, the similarity between them can be calculated

$$cos(d_i , d_j) = \frac{d_i . d_j}{|| d_i || || d_j ||} \quad (6)$$

Since the document vectors are of unit length, the above equation is simplified to:

$$cos (d_i, d_j) = d_i . d_j \quad (7)$$

When the cosine value is 1 the two documents are identical, and 0 if there is nothing in common between them (i.e., their document vectors are orthogonal to each other).

4.2 Jaccard Coefficient

The Jaccard coefficient, which is sometimes referred to as the Tanimoto coefficient, measures similarity as the intersection divided by the union of the objects. For text document, the Jaccard coefficient compares the sum weight of shared terms to the sum weight of terms that are

present in either of the two documents but are not the shared terms.

The Cosine Similarity may be extended to yield Jaccard Coeff. in case of Binary attributes

$$\text{Jaccard Coff (A,B)} = \frac{\sum_i A_i \,.\, B_i}{\sum_i \|A_i\|^2 + \sum_i \|B_i\|^2 - \sum_i A_i * B_i} \quad (8)$$

$$\text{Jaccard Index (A, B)} = \frac{A \cap B}{AU\, B} \quad (9)$$

4.3 Euclidean Similarity

This is the most usual, "natural" and intuitive way of computing a distance between two samples. It takes into account the difference between two samples directly, based on the magnitude of changes in the sample levels. This distance type is usually used for data sets that are suitably normalized or without any special distribution problem.

$$\text{Euclidean Distance (A, B)} = \sqrt{\sum_i (A_i - B_i)^2} \quad (10)$$

$$\text{Euclidean Similarity (A, B)} = 1 - \sqrt{\sum_i (A_i - B_i)^2} \quad (11)$$

4.4 Pearson Correlation Coefficient

This distance is based on the Pearson correlation coefficient that is calculated from the sample values and their standard deviations. The correlation coefficient 'r' takes values from –1 (large, negative correlation) to +1 (large, positive correlation). Effectively, the Pearson distance dp is computed as dp = 1 - r and lies between 0 (when correlation coefficient is +1, i.e., the two samples are most similar) and 2 (when correlation coefficient is -1).

$$SIM_P(\vec{t_a}, \vec{t_b}) = \frac{m\sum_{t=1}^{m} w_{t,a} \times w_{t,b} - TF_a \times TF_b}{\sqrt{[m\sum_{t=1}^{m} w_{t,a}^2 - TF_a^2][m\sum_{t=1}^{m} w_{t,b}^2 - TF_b^2]}}$$

$$\text{where } TF_a = \sum_{t=1}^{m} w_{t,a} \text{ and } TF_b = \sum_{t=1}^{m} w_{t,b}. \quad (12)$$

Where t_a and t_b are m-dimensional vectors over the term set $T = \{t_1, \ldots, t_m\}$.

The Euclidean distance is a distance measure, while the cosine similarity, Jaccard coefficient and Pearson coefficient are similarity measures. We apply a simple transformation to convert the similarity measure to distance values. Because both cosine similarity and Jaccard coefficient are bounded in [0, 1] and monotonic, we take D = 1 − SIM as the corresponding distance value. For Pearson coefficient, which ranges from −1 to +1, we take D = 1 − SIM when SIM ≥ 0 and D = |SIM| when SIM < 0.

5. Clustering Algorithm

For our analysis, we have chosen K-means algorithm to cluster documents. This is an iterative Partitional clustering process that aims to minimize the least squares error criterion [6]. As mentioned previously, Partitional clustering algorithms have been recognized to be better suited for handling large document datasets than Hierarchical ones, due to their relatively low computational requirements [7, 8, 9]. The standard K-means algorithm works as follows. Given a set of data objects D and a pre-specified number of clusters k, k data objects are randomly selected to initialize k clusters, each one being the centroid of a cluster. The remaining objects are then assigned to the cluster represented by the nearest or most similar centroid. Next, new centroids are recomputed for each cluster and in turn all documents are re-assigned based on the new centroids. This step iterates until a converged and fixed solution is reached, where all data objects remain in the same cluster after an update of centroids. The generated clustering solutions are locally optimal for the given data set and the initial seeds. Different choices of initial seed sets can result in very different final partitions. Methods for finding good starting points have been proposed [10]. However, we will use the basic K-means algorithm because optimizing the clustering is not the main focus of this paper.

The K-means algorithm works with distance measures which basically aims to minimize the within-cluster distances. Therefore, similarity measures do not directly fit into the algorithm, because smaller values indicate dissimilarity.

1. Select K points as the initial centroids.
2. Assign all points to the closest centroid.
3. Recompute the centroid of each cluster.
4. Repeat steps 2 and 3 until the centroids don't change.

5.1 Porter Stemming Algorithm

The Porter Stemmer is a conflation Stemmer developed by Martin Porter at the University of Cambridge in 1980. The stemmer is a context sensitive suffix removal algorithm. It is the most widely used stemmer and implementations are available in many languages. This stemmer is a linear step stemmer divided into a five linear steps that are used to produce the final stem. Within each step, if a suffix rule matched to a word, then the conditions attached to that rule are tested on what would be the resulting stem, if that suffix was removed, in the way defined by the rule. For example such a condition may be, the number of vowel

characters, which are followed be a consonant character in the stem (Measure), must be greater than one for the rule to be applied. The resultant stem being returned by the Stemmer after control has been passed from step five. See Porter Stemmer figure 1. However a number of definitions regarding the stemmer need to be made before the steps can be explained. The following definitions are presented in [17].

A *consonant* is a letter other than A, E, I, O or U and other than Y preceded by a consonant. For example in the word *boy* the consonants are B and Y, but in *try* they are T and R. A *vowel* is any letter that is not a consonant. A list of consonants greater than or equal to length one will be denoted by a *C* and a similar list of vowels by a *V* [17].

Any word can therefore be represented by the single form;

$$[C]\ (VC)^m\ [V]$$

Where the *m* denotes m repetitions of VC and the square brackets *[]* denote the optional presence of their contents [17]. The value *m* is called the *measure* of a word and can take any value greater than or equal to zero, and is used to decide whether a given suffix should be removed. All such rules are of the form; (*condition*) S1 → S2 which means that the suffix S1 is replaced by S2 if the remaining letters of S1 satisfy the *condition* [17].

The first step of the algorithm is designed to deal with past participles and plurals. This step is the most complex and is separated into three parts in the original definition, 1a, 1b and 1c. The first part deals with plurals, for example *sses* → *ss* and removal of *s*. The second part removes *ed* and *ing*, or performs *eed* → *ee* where appropriate. The second part continues only if *ed* or *ing* is removed and transforms the remaining stem to ensure that certain suffices are recognized later. The third part transforms a terminal *y* to an *i*, this part is inserted as step 2.
The remaining steps are relatively straightforward and contain rules to deal with different order classes of suffices, initially transforming double suffices to a single suffix and then removing suffices providing the relevant conditions are met [17].

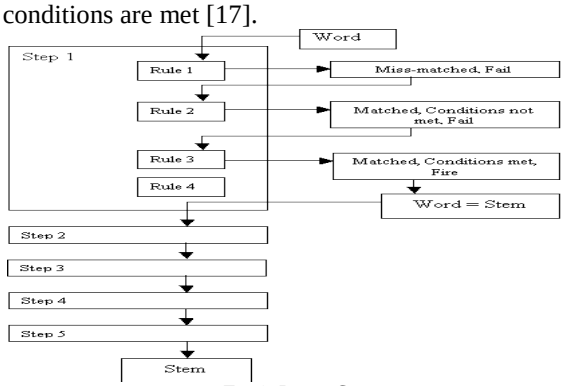

Fig 1: Porter Stemmer

6. EXPERIMENT

It is very difficult to conduct a systematic study comparing the impact of similarity measures on cluster quality with and without preprocessing the documents, because objectively evaluating cluster quality is difficult in itself. In practice, manually assigned category labels are usually used as baseline criteria for evaluating clusters. As a result, the clusters, which are generated in an unsupervised way, are compared to the pre-defined category structure, which is normally created by human experts. This kind of evaluation assumes that the objective of clustering is to replicate human thinking, so a clustering solution is good if the clusters are consistent with the manually created categories. However, in practice datasets often come without any manually created categories and this is the exact point where clustering can help. The rest of this section first describes the characteristics of the datasets, then explains the evaluation measures, and finally presents and analyzes the experiment results.

6.1 Dataset

This work experiments with one bench mark dataset Classic dataset collected from uci.kdd repositories. Classic dataset consists of four different collections CACM, CISI, CRAN and MED. We have considered 800 documents of the total 7095 documents.

In this datasets, some of the documents consists single word only, so it is meaningless to take such documents for document dataset. For eliminating these invalid documents we apply file reduction on each category, which returns the documents that supports mean length of each category. For file reduction we construct the Boolean matrices of all documents by category wise and calculate mean length of each category and removed the documents from the dataset which doesn't support mean length. By this we got valid documents. From these valid documents we have collected 800 documents of four categories each. From classic dataset 200 documents of each category again totaling to 800 documents.

6.2 Pre-Processing

Preprocessing consists of steps that take as input a plain text document and output a set of tokens (which can be single terms or n-grams) that are to be included in the vector model. In this work we performed removal of stop words and after taking users choice to perform stemming and built vector space model. We have pruned words that appear with very low frequency throughout the corpus with the assumption that these words, even if they had any discriminating power, would form too small clusters to be useful. Words which occur frequently are also removed. In

this work we have compared the performance of kmeans algorithm on documents without stemming with the documents with stemming.

6.3 Evaluation

For clustering quality evaluation are using entropy as a measure of quality of the clusters (with the caveat that the best entropy is obtained when each cluster contains exactly one data point). Let CS be a clustering solution. For each cluster, the class distribution of the data is calculated first, i.e., for cluster j we compute p_{ij}, the "probability" that a member of cluster j belongs to class i. Then using this class distribution, the entropy of each cluster j is calculated using the standard formula

$$E_j = -\sum_i p_{ij} \log(p_{ij}) \quad (13)$$

where the sum is taken over all classes. The total entropy for a set of clusters is calculated as the sum of the entropies of each cluster weighted by the size of each cluster:

$$E_{CS} = \sum_{j=1}^{m} \frac{n_j * E_j}{n} \quad (14)$$

where n_j is the size of cluster j, m is the number of clusters, and n is the total number of data points.

6.4 Results Analysis

The seed points are statically chosen, but efficiency can be improved if seeds selected are random or run the code more than once to check the efficiency. As shown in tables 3, 4 Euclidean performs worst without applying stemming algorithm. As shown in Tables 1, 2 and Tables 3, 4 Euclidean distance performs worst with and without preprocessing the data. We also observe from tables 3, 4 that Jaccard Measure performs well after applying the stemming algorithm. We observe that Pearson performs the best with and without preprocessing of the data. From our results it is observed that Boolean representation with Pearson measure, Frequency count with Cosine and Euclidean also has non-zero clusters when we do not apply the stemming algorithm. Hence the overall entropy representation table for Boolean, Frequency Count and Term frequency and Inverse Document Frequency shows NaN values for other measures as some of the clusters are empty. On an average, the Jaccard and Pearson measures are slightly better in generating more coherent clusters, which means the clusters have lower entropy scores. Tables 5,6 shows one partition as generated by the Boolean Pearson measure using Reuter's dataset, and Tables 7,8 shows one partition as generated by the TF-IDF Jaccard Coefficient measure using Classic dataset which has the lowest entropy value.

Table 1: Entropy Results of Different Vector Space Representations Using Classic dataset without Porter stemming algorithm

Entropy	Cosine	Jaccard	Euclidean	Pearson
Boolean	NaN	NaN	NaN	0.08
Frequency Count	NaN	0.20	NaN	0.08
TF-IDF	0.16	0.13	NaN	0.08

Table 2: Entropy Results of Different Vector Space Representations Using Classic dataset with Porter stemming algorithm

Entropy	Cosine	Jaccard	Euclidean	Pearson
Boolean	NaN	NaN	NaN	0.08
Frequency Count	0.25	0.17	0.44	0.07
TF-IDF	0.08	0.11	0.44	0.07

We see from tables 1 and 2 that the Euclidean distance is again proved to be an ineffective metric for modeling the similarity between documents. But after applying Porter there is little improvement in the Euclidean measure. But Cosine tends to perform well in TF-IDF representation after applying porter algorithm. The Pearson's coefficient tends to outperform all the measures before and after stemming of the documents.

Table 3: TF-IDF Entropy Results using Classic dataset without Porter stemming

	Cosine	Jaccard	Euclidean	Pearson
Clusters[0]	0.31	0.0	0.41	0.03
Clusters[1]	0.01	0.23	0.28	0.04
Clusters[2]	0.06	0.10	0.10	0.07
Clusters[3]	0.13	0.15	NaN	0.17

Table 4: TF-IDF Entropy Results using Classic dataset with Porter stemming

	Cosine	Jaccard	Euclidean	Pearson
Clusters[0]	0.05	0.01	0.30	0.01
Clusters[1]	0.01	0.08	0.30	0.04
Clusters[2]	0.06	0.07	0.30	0.07
Clusters[3]	0.13	0.11	0.00	0.10

Here we see in tables 3 and 4 Jaccard measure performs well after applying porter algorithm.

Table 5: Clustering Results from Boolean Pearson Correlation Measure using Classic dataset without porter

	CACM	CISI	CRAN	MED
Cluster[0]	1	1	2	198
Cluster[1]	2	2	195	2
Cluster[2]	12	188	2	5
Cluster[3]	185	1	9	3

Table 6: Clustering Results from Boolean Pearson Correlation Measure using Classic dataset with porter

	CACM	CISI	CRAN	MED
Cluster[0]	0	0	3	189
Cluster[1]	4	1	193	4
Cluster[2]	7	186	1	5
Cluster[3]	189	13	3	2

Table 7: Clustering Results from TFIDF Jaccard Measure using Classic dataset without porter

	CACM	CISI	CRAN	MED	Label
Cluster[0]	0	0	0	164	**MED**
Cluster[1]	18	6	198	31	**CRAN**
Cluster[2]	10	166	1	2	**CISI**
Cluster[3]	172	28	1	3	**CACM**

Table 8: Clustering Results from TFIDF Jaccard Measure using Classic dataset with porter

	CACM	CISI	CRAN	MED	Label
Cluster[0]	0	1	0	166	**MED**
Cluster[1]	8	5	199	30	**CRAN**
Cluster[2]	3	166	0	4	**CISI**
Cluster[3]	189	28	1	0	**CACM**

We can see from the above tables 7 and 8 that the cluster accuracy with porter is 90% and of without porter is 87.5%. Hence applying stemming will improve cluster quality.

The Clustering accuracy r is defined as

$$r = \frac{\sum_{i=1}^{4} a_i}{n} \qquad (15)$$

where a_i is the number of instances occurring in both cluster i and its corresponding class and n is the number of instances in the dataset.

7. Conclusions and Future Work

In this study we found that all the measures have significant effect on Partitional clustering of text documents before and after applying the stemming algorithms. Of course the Euclidean distance measure performs worst. Pearson correlation coefficient is slightly better as the resulting clustering solutions are more balanced and is nearer to the manually created categories. The Jaccard and Pearson coefficient measures find more coherent clusters. The Jaccard Measure works better after applying stemming algorithm. Considering the type of cluster analysis involved in this study, we can see that there are four components that affect the final results—representation of the documents, applying the stemming algorithms, distance or similarity measures considered, and the clustering algorithm itself. In our future work our intension is to apply semantics knowledge to the document representations to represent relationships between terms and study the effect of these stemming algorithms exhaustively.

REFERENCES

[1] C. J. Van Rijsbergen, (1989), Information Retrieval, Buttersworth, London, Second Edition.
[2] G. Kowalski, Information Retrieval Systems – Theory and Implementation, Kluwer Academic Publishers, 1997.
[3] D.R. Cutting, D.R. Karger, J.O. Pedersen, and J.W. Tukey, Scatter/Gather: A Cluster-based Approach to Browsing Large Document Collections, SIGIR '92, Pages 318 – 329, 1992.
[4] O. Zamir, O. Etzioni, O. Madani, R.M. Karp, Fast and Intuitive Clustering of Web Documents, KDD '97, Pages 287-290, 1997.
[5] D. Koller and M. Sahami, Hierarchically classifying documents using very few words, Proceedings of the 14th International Conference on Machine Learning (ML), pp. 170-178, 1997.
[6] G. Salton. Automatic Text Processing. Addison-Wesley, New York, 1989.
[7] M. Steinbach, G. Karypis, and V. Kumar. A Comparison of Document Clustering Techniques. In KDD Workshop on Text Mining, 2000.
[8] D. R. Cutting, J. O. Pedersen, D. R. Karger, and J. W. Tukey. Scatter/gather: A cluster-based approach to browsing large document collections. In Proceedings of the ACM SIGIR, 1992.
[9] B. Larsen and C. Aone. Fast and Effective Text Mining using Linear-time Document Clustering. In Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 1999.

[10] D. Arthur and S. Vassilvitskii. k-means++ the advantages of careful seeding. In Symposium on Discrete Algorithms, 2007.
[11] G. Salton and C. Buckley. Term weighting approaches in automatic text retrieval. *Information Processing & Management*, 24(5):513–523, 1988.
[12] Krovetz, R. (1993). Viewing morphology as an inference process. In Proceedings of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 191-202.
[13] Xu, J. and Croft, B. (1998). Corpus-Based Stemming using Co-occurrence of Word Variants. ACM Transactions on Information Systems, 16 (1).
[14] Arampatzis, A, van der Weide, Th.P., Koster, C.H.A., and van Bommel, P. (2000). Linguistically-motivated Information Retrieval. Encyclopedia of Library and Information Science, published by Marcel Dekker, Inc. - New York – Basel. To appear.
[15] Hull, D. (1996). Stemming algorithms: A case study for detailed evaluation. Journal of the American Society for Information Science, 47(1), 70-84.
[16] Lovins, J. B. (1968). "Development of a Stemming Algorithm", Mechanical Translation and Computational Linguistics, 11.
[17] Porter, M.F. (1980) *An Algorithm for Suffix Stripping*, Program, 14(3): 130-137

N.Sandhya B.Tech, M.Tech (Ph.D). I passed B.Tech in 2000 and M.Tech in 2007. Registered Ph.D in 2008. Has 11 years of experience in teaching. Working in GRIET. My areas of interest are Databases, Data Mining , Information Retrieval and Text Mining.

Y.Srilalitha M.Tech (Ph.D). I completed M.Tech in 2001. Registered Ph.D in 2008. Has 16 years of experience in teaching. Working in GRIET. My areas of interest are Information Retrieval, Data Mining and Text Mining.

V.Sowmya M.Tech (Ph.D). I completed M.Tech in 2009. Registered Ph.D in 2011. Has 6 years of experience in teaching. Working in GRIET. My areas of interest are Information Retrieval, Data Mining and Text Mining.

Dr.K.Anuradha M.Tech, Ph.D. I completed Ph.D in 2011. Working as professor and Head of the CSE Dept in GRIET. My areas of interest are Information Retrieval, Data Mining and Text Mining.

Dr.A.Govardhan M.Tech, Ph.D.
Working as professor and Principal of JNTU, Jagityal. Has an experience of 20 years in teaching. My areas of interest are Information Retrieval, Databases, Data Mining and Text Mining.