A Comparative Study to Find a Suitable Method for Text Document Clustering

S.C.Punitha
Assistant Professor, Head of the department, Department of Computer Science, P.S.G.R. Krishnammal College for women, Coimbatore, India.

M.Punithavalli
Director of the Computer Science,
Sri Ramakrishna College of engineering,
Coimbatore, India

ABSTRACT
Text mining is used in various text related tasks such as information extraction, concept/entity extraction, document summarization, entity relation modeling (i.e., learning relations between named entities), categorization/classification and clustering. This paper focuses on document clustering, a field of text mining, which groups a set of documents into a list of meaningful categories. The main focus of this paper is to present a performance analysis of various techniques available for document clustering. The results of this comparative study can be used to improve existing text data mining frameworks and improve the way of knowledge discovery. This paper considers six clustering techniques for document clustering. The techniques are grouped into three groups namely Group 1 – K-means and its variants (traditional K-means and K* Means algorithms), Group 2 - Expectation Maximization and its variants (traditional EM, Spherical Gaussian EM algorithm and Linear Partitioning and Reallocation clustering (LPR) using EM algorithms), Group 3 - Semantic-based techniques (Hybrid method and Feature-based algorithms). A total of seven algorithms are considered and were selected based on their popularity in the text mining field. Several experiments were conducted to analyze the performance of the algorithm and to select the winner in terms of cluster purity, clustering accuracy and speed of clustering.

Keywords
Text mining, Traditional K-Means, Traditional EM Algorithm, sGEM, HSTC model, TCFS method

1. Introduction

In the current information explosion era, amount of data stored in form of text, image, video and audio is enormous and is expected to grow in future. Moreover, with the technological breakthroughs in fields like e-libraries and e-publishing, is increasing the magnitude and usage of digital documents. This increase, in turn, has increased the demand for tools that can be used to analyze and discover useful knowledge. Usage of data mining techniques on text documents is an area of research referred to has text mining or text data mining and is a field of research that has attracted several researchers (Zhang and Pan, 2011; Gad and Kamel, 2010).

Text mining is used in various text related tasks such as information extraction, concept/entity extraction, document summarization, entity relation modeling (i.e., learning relations between named entities), categorization/classification and clustering. This paper focuses on document clustering, a field of text mining, where automatic methods that group a set of documents into a list of meaningful categories, in such a way that the documents in a category are similar to each other and dissimilar to documents in other categories (Andrews and Edward, 2007; Shaw at, 2008).

They have wide usage in several applications areas such as security applications (study of text encryption), biomedical applications, software applications (search and indexing), online media applications (improve search experience), marketing applications (predictive analysis), sentimental analysis (favourability testing) and academic applications (journals). All these techniques work with the common aim of extracting higher quality information from text. For this purpose, the market is always in search for techniques that will improve the knowledge discovery process. The main focus of this paper is to present a performance analysis of various techniques available for text document analysis, in particular, document clustering process. The results of this comparative study can be used to improve existing text data mining frameworks and improve the way of knowledge discovery.

This paper considers six clustering techniques for document clustering. The techniques are grouped into three groups namely Group 1 : K-means and its variants (traditional K-means and K* Means algorithms)

Group 2 : EM (Expectation Maximization) and its variants (traditional EM, Spherical Gaussian EM algorithm and Linear Partitioning and Reallocation clustering (LPR) using EM algorithms),

Group 3 : Semantic-based techniques (Hybrid method and Feature-based algorithms)

Thus a total of seven algorithms are considered and were selected based on their popularity in the text mining field. The rest of the paper is organized as follows. The seven selected clustering algorithms are described in Sections 2

Manuscript received October 5, 2012
Manuscript revised October 20, 2012

to 4, Section 5 presents the results of the performance analysis and Section 6 concludes the work with future research direction.

2. Group 1 Algorithms

This section describes the techniques under Group 1 techniques, namely, the traditional K-means and K*-Means algorithms.

2.1 Traditional K-Means Algorithm

The K-means algorithm assigns each point to a cluster whose center (also called centroid) is nearest. The center is the average of all the points in the cluster, that is, its coordinates are the arithmetic mean for each dimension separately over all the points in the cluster. K-Means can be thought of as an algorithm relying on hard assignment of information to a given set of partitions. At every pass of the algorithm, each data value is assigned to the nearest partition based upon some similarity parameter such as Euclidean distance of intensity (Kanungo, et al. 2002). The partitions are then recalculated based on these hard assignments. With each successive pass, a data value can switch partitions, thus altering the values of the partitions at every pass. K-Means algorithms typically converge to a solution very quickly as opposed to other clustering algorithms.

The algorithm steps are:

1. Choose the number of clusters, k.
2. Randomly generate k clusters and determine the cluster centers, or directly generate k random points as cluster centers.
3. Assign each point to the nearest cluster center.
4. Re-compute the new cluster centers.
5. Repeat steps 2 and 3 until convergence.

The traditional K-means algorithm needs the input information of exactly the number of clusters that are to be as distinct as possible. In general, the traditional K-means method produces exactly k different clusters, which have greatest possible distinction (Bradley and Fayyad, 1998). The algorithm has the advantage of being simple and fast even with large datasets.

2.2. K*-Means Algorithm

This section presents a STep-wise Automatic Rival-penalized (STAR) K-means algorithm, which is a generalized version of the traditional K-means clustering algorithm (Cheung, 2003). The initial step, generally considered as a pre-processing step, allows each cluster to acquire at least one seed point. Then, the next step fine-tunes the units adaptively using a learning rule which automatically penalizes the winning chance of all rival seed points in the consequent competitions and tunes the winning one to adapt to an input. The detailed K*-Means algorithm is outlined below.

A Frequency Sensitive Competitive Learning algorithm is used to determine the 'k' initial seeds. The seeds are selected in such a way that it is not less than the exact number k* of clusters.
Randomly select a data point x_t and for j = 1, 2, . . . k let,

$$u_j = \begin{cases} 1 & \text{if } j = w = \arg\min_r \lambda_r \| x_t - m_r \| \\ 0 & \text{Otherwise} \end{cases} \quad (1)$$

where $\lambda_r = \frac{nj}{\sum_{r=1}^{k} nr}$ and n_r represents the cumulative number of occurrences of $u_r = 1$.

Update the winning seed point m_w using Equation (2),

$$m_w^{new} = m_w^{old} + \eta(x_t - m_w^{old}) \quad (2)$$

where η is the smallest positive learning rate.

Repeat Steps 2 and 3 until the k series of u_j (j = 1, 2, . . . k) remain unchanged for all x_ts.

Initialize $\alpha_j = 1/k$, for j = 1, 2, … ,k and let Σ_j be the covariance matrix of those data points with $u_j = 1$.

Given a data point x_t, calculate $I(j|x_t)$s using Equation (3),

$$I(j/x) = \begin{cases} 1 & \text{if } j = 2 = \arg\min_r \rho_r \\ 0 & \text{Otherwise} \end{cases} \quad (3)$$

Update the winning seed point m_w (Equation 4)

$$m_w^{new} = m_w^{old} + \eta \sum_{w}^{-1} (x_t - m_w^{old}) \quad (4)$$

Update parameters α_js and Σ_w.

Repeat Steps 6 and 7 until the k series of $I(j|x_t)$, with j = 1, 2, . . . k, remain unchanged for all x_ts.

The k*means clustering algorithm overcomes some of the limitations of conventional k means clustering algorithm. The k*means clustering algorithm eliminates the problem of dead unit that was available with conventional k means clustering algorithm.

3. Group 2 Algorithms

This section describes the techniques under Group 2 techniques, namely, the traditional EM algorithm, Spherical Gaussian EM algorithm and Linear Partitioning and Reallocation clustering using EM Algorithm.

3.1. Traditional EM Algorithm

The main goal of EM techniques is to detect clusters in observations (or variables) and to assign those observations to the clusters. The fundamental approach and logic of this clustering method is to compute a single continuous large variable in a huge sample of observations (He *et al.*, 2003; Boley and Borst, 1999). Further, considering that the sample of given dataset consists of two clusters of observations with different means within each sample, and then the distribution of values for the continuous large variable follow a normal distribution.

The EM (expectation maximization) algorithm extends this basic approach to clustering in two important ways: Instead of assigning cases or observations to clusters to maximize the differences in means for continuous variables, the EM clustering algorithm computes probabilities of cluster memberships based on one or more probability distributions (Singh, 2004). The goal of the clustering algorithm then is to maximize the overall probability or likelihood of the data, given the (final) clusters. Unlike the classic implementation of K-means clustering, the general EM algorithm can be applied to both continuous and categorical variables (note that the classic K-means algorithm can also be modified to accommodate categorical variables) (McLachlan and Krishnan, 1997). The EM algorithm is very similar in setup to the K-Means algorithm. Similarly, the first step is to choose the input partitions. The EM cycle begins with an Expectation step, which is defined by the following equation:

$$E[Z_{ij}] = \frac{p(x = x_i \mid \mu = \mu_j)}{\sum_{n=1}^{k} p(x = x_i \mid\mid \mu = \mu_n)} = \frac{e^{-\frac{1}{2\sigma^2}(x_i - \mu_j)^2}}{\sum_{n=1}^{k} e^{-\frac{1}{2\sigma^2}(x_i - \mu_n)^2}} \quad (5)$$

This equation states that the expectations or weight for pixel z with respect to partition j equals the probability that x is pixel xi given that μ is partition μ_i divided by the sum over all partitions k of the same previously described probability. This leads to the lower expression for the weights. The sigma squared seen in the second expression represents the covariance of the pixel data. Once the E step has been performed and every pixel has a weight or expectation for each partition, the M step or maximization step begins. This step is defined by the following equation:

$$\mu_j \leftarrow \frac{1}{m} \sum_{i=1}^{m} E[Z_{ij}]x_i \quad (6)$$

This equation states that the partition value j is changed to the weighted average of the pixel values where the weights are the weights from the E step for this particular partition. This EM cycle is repeated for each new set of partitions until, as in the K-Means algorithm, the partition values no longer change by a significant amount.

3.2. Spherical Gaussian EM algorithm (sGEM)

It is possible to refine the partitioning results by reallocating new cluster membership. The basic idea of the reallocation method (Rasmussen, 1992) is to start from some initial partitioning of the data set, and then proceed by moving objects from one cluster to another cluster to obtain an improved partitioning. Thus, any iterative optimization-clustering algorithm can be applied to do such operation. The problem is formulated as a finite mixture model, and applies a variant of the EM algorithm for learning the model.

The most critical problem is how to estimate the model parameters. The data samples are assumed to be drawn from the multivariate normal density and it is further also assumed that the features are statistically independent and a component generates its members from the spherical Gaussian with the same covariance matrix (Dasgupta and Schulman, 2000). Figure 1 gives an outline sGEM algorithm. The algorithm tries to maximize log Lc at very step, and iterates until convergence. For example, the algorithm terminates when $\Delta \log L_c < \delta$, where δ is a pre defined threshold.

3.3. Linear Partitioning and Reallocation using EM Algorithm (LPR)

The problem of clustering a dataset into groups can be obtained by calculating the mean value of the data set first and then compare each point with the mean value. If the point value is less the mean value, it is assigned to the first group. Otherwise, it is assigned to the second group. The problem arises while considering high dimensional data set. Based on the idea of the PDDP (Principal Direction Divisive Partitioning) algorithm, this problem can be dealt by projecting all the data points onto the principal direction the principal eigenvector of the covariance matrix of the data set, and then the splitting process can be performed based on this principal direction.

Begin

Initialization : Set $(z_i)_j^{(0)}$ from a partitioning of the data and $t \leftarrow 0$

Repeat

E – Step : For each d_i, $1 \le i \le n$ and c_j, $1 \le j \le k$ Find iits new component index as

$$(z_i)_j^{(t+1)} = \begin{cases} 1 & \text{if } j^* = \arg\max_j \log(P^{(t)}(c_j \mid d_i; \theta_j)) \\ 0 & \text{Otherwise} \end{cases}$$

M-Step : Re-estimate the model parameters

$$P_{(c_j)}{}^{(t+1)} = \frac{1}{n} \sum_{i=1}^{n} (Z_i)_j^{(t+1)}$$

$$m_j^{(t+1)} = \frac{\sum_{i=1}^{n} d_i (z_i)_j^{(t+1)}}{\sum_{i=1}^{n} (z_i)_j^{(t+1)}}$$

$$\sigma^{2(t+1)} = \frac{1}{nd} \sum_{i=1}^{n} \sum_{j=1}^{n} \| d_i - m_j \|^2 (z_i)_j^{(t+1)}$$

Until $\Delta \log L_e(\Theta) < \delta$

end

Figure 1: sGEM Algorithm

In geometric terms, the data points are partitioned into two sub clusters using the hyper plane normal to the principal direction passing through the mean vector (Boley, 1998). This hyper plane is referred as the linear partitioning hyper plane. Figure 2 illustrates the principal direction and the linear partitioning hyper plane on the 2d2k data set, containing 1000 points distributed in 2 Gaussians.

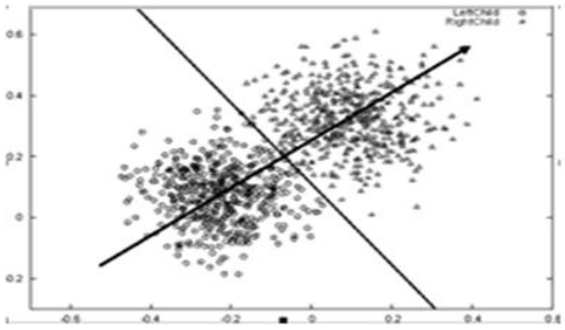

Figure 2: Principal Direction and linear partitioning Hyper plane

The PDDP algorithm begins with all the document vectors in a large single cluster. This procedure continues by recursively splitting the cluster into two sub clusters using the linear partitioning hyperactive plane according to the discriminant functions of the algorithm. This procedure terminates by splitting based on some heuristic, e.g. a pre-defined number of clusters. Finally, a binary tree is yielded out as the output, whose leaf nodes form the resulting clusters. To keep this binary tree balanced, it selects an un split cluster to split by using the scatter value, measuring the average distance from the data points in the cluster to their centroid. The severe problem of the PDDP algorithm is that it cannot achieve good results when clusters are not well separated from one another. This figure 2 and 3 illustrates this drawback. Figure 3 shows two partitions produced by performing the first iteration of the PDDP algorithm on a dimensional data set. The data set consists of 334 points. The actual class labels are not given, but one can observe that it is composed of five compact clusters (He *et al.*, 2003). Based on the principal direction and the corresponding linear partitioning hyper plane, it can be seen that the PDDP algorithm starts with significantly wrong partitioning on the middle left hand cluster. Figure 4 shows three partitions after the second iteration. The partitioning is performed until convergence is reached.

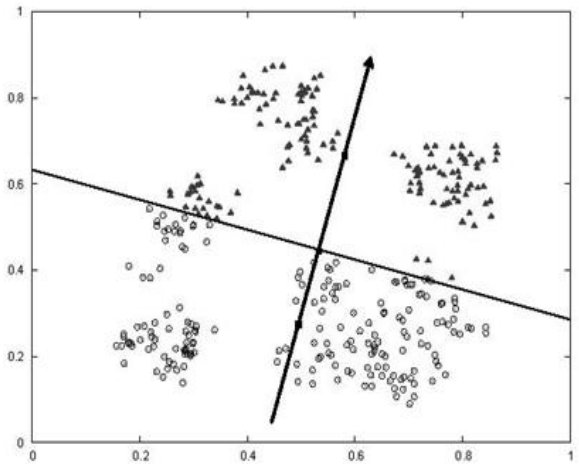

Figure 3: 2 partitions after 1st iteration

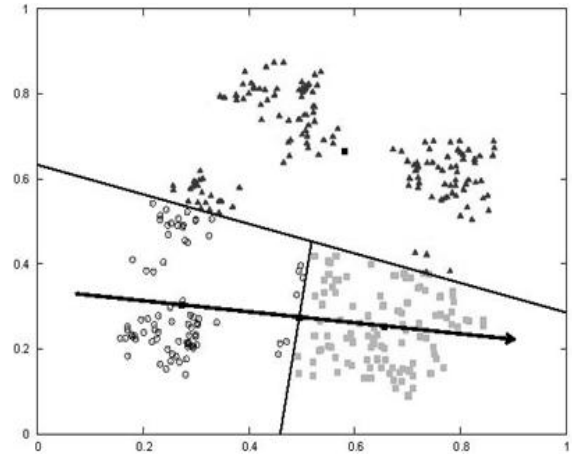

Figure 4: 3 partitions after the 2nd iteration

4. Group 3 Algorithms

Decherchi *et al*. (2009) proposed a hybrid scheme that combined pattern recognition grouping algorithm with semantic driven method to arrange unstructured documents into content-based homogeneous groups. This model is referred as HSTC (Hybrid Scheme for Text Clustering) in this paper. They used a semantic-based metric measure distance to calculate the similarity ratio between documents by performing a content and behavioral bases analysis. This had the advantage of taking into account the lexical and structural properties along with the style characteristics of the processed documents. They used a Radial Basis Function (RBF) for clustering. Another work that used semantic characteristics for text document was proposed by Raja and Narayanan (2010) and Thangamani and Thangaraj (2010). The model used a new Text Clustering with Feature Selection (TCFS) method to improve text document clustering. The system was designed to identify the semantic relations using ontology, which represents the term and concept relationship. From these relationships, a concept weight is calculated and used during clustering. Both the systems offer efficient methods that enhance the document clustering process. This section explains both these algorithms.

4.1. HSTC Model

The HSTC Model takes advantage of content-based processing for efficient clustering. The algorithm performs clustering on a dataset D containing 'n' documents represented as $D = \{D_j; j = 1 .. n_D\}$ having a collection of terms $T = \{t_i; i = 1 .. n_T\}$ obtained after performing pre-processing. The preprocessing performs stop-word removal and stemming to removal repeated and irrelevant terms. A content-based distance measure is used as similarity measure. This measure combines the distribution-based measure with the behavioural characteristics of the document features. The inclusion of behavioral characteristics includes document structure and style information into similarity evaluation, so as to improve the overall clustering performance. While calculating the document distance measure, a document 'D' is represented using two vectors, V* and V**. V*(D) represents the content description of D and is a set of terms where each term 't' is associated with its normalized frequency 'tf'. Thus, the kth element of vector V*(Di) can be calculated using Equation (7).

$$V^* = tf_{k,i} \Big/ \sum_{l=1}^{n_r} tf_{l,i} \tag{7}$$

where $tf_{k,i}$ is the frequency of the kth term in document D_i. Thus V* represents a document as a vector using term frequencies to set weights associated to each element. The distance between a pair of documents (D_i, D_j) is calculated using Equation (8) and is represented as $\Delta(f)$.

$$\Delta^{(f)}(D_i, D_j) = \left[\sum_{k=1}^{n_r} | V^*_{k,u} - V^*_{k,v} |^p \right]^{1/p} \tag{8}$$

In the HSTC model, p = 1 and therefore actually implements Manhattan distance metric. The second vector V** takes into consideration the structural properties of a document and is represented as a set of probability distributions associated with the term vector. Here, each term $t \in T$ occurring in a document D is associated with a distribution function that gives the spatial probability density function (pdf) of 't' in D. Such a distribution, pt,u(s), is generated under the hypothesis that, when detecting the kth occurrence of a term 't' at the normalized position $sk \in [0,1]$ in the text, the spatial pdf of the term can be approximated by a Gaussian distribution centered around sk. In other words, if the term tj is found at position sk within a document, a second document with similar

structure is expected to include the same term at the same position or in a neighborhood thereof, with a probability defined by a Gaussian pdf. To derive a formal expression of the pdf, assume that the ith document, D_i, holds no occurrences of terms after simplifications. If a term occurs more than once, each occurrence is counted individually when computing no, which can be viewed as a measure of the length of the document. The spatial pdf is defined using Equation (9).

$$p_{t,u}(s) = \frac{1}{A}\sum_{k=1}^{n_o} G(s_k, \lambda) \quad (9)$$

where A and λ are normalization terms, G is the Gaussian pdf given by Equation (10)

$$G(s_k, \lambda) = \frac{1}{\sqrt{2\pi\lambda}} \exp\left[-\frac{(s-s_k)^2}{\lambda^2}\right] \quad (10)$$

From this the second term vector V** is calculated by considering a discrete approximation of Equation (9). Here, the document D is segmented evenly into S sections, from which S-dimensional vectors are generated for each term t ∈ T. Each element estimates the probability of a term 't' occurring in the corresponding section of the document. Thus, v**(D) is represented as an array of nT vectors having dimension S. The distance between the probability vectors thus created (V**) is calculated by using Euclidean metric (Equation 11) and is represented as Δ(b) for two documents D_i and D_j.

$$\Delta^{(b)}(D_i, D_j) = \sum_{k=1}^{n_r} \Delta_{t_k}^{(b)}(D_i, D_j) = \sum_{k=1}^{n_r}\sum_{s=1}^{S}\left[v_{(k)s,i}^{n} - v_{(k)s,j}^{n}\right] \quad (11)$$

From the calculated Δ(f) and Δ(b), the final distance is calculated using Equation (12).

$$\Delta(D_i, D_j) = \alpha\, \Delta^{(f)}(D_i, D_j) + (1-\alpha)\, \Delta^{(b)}(D_i, D_j) \quad (12)$$

here α ∈ [0, 1] is the mixing coefficient weight. For the last stage, that is the actual clustering, a kernel-based k-means partitioning algorithm (Girolami, 2002) is used for grouping similar documents in a top-down hierarchical process. In particular, a k-means clustering adopting rbf-kernel (radial basis function kernel) is used.

4.2. TCFS Method

The methodology used by TCFS method is similar to that of HSTC method, but it differs in three ways. The first is in the preprocessing stage, second is the clustering process and the third is in the clustering algorithm used. Both use the same similarity measure, cosine distance. In the preprocessing stage, apart from stop word elimination and stemming, a weight estimation function, that calculates the term weight and semantic weight, are included. Term weight is estimated using TF/IDF values that utilize information about term and number of times (n) it appears in the document. Using the term weight value a term cube is constructed. A term cube is a 3-D model representing the document, term and n relationship. The semantic weight is calculated by concept extraction, concept or semantic weight calculation and construction of semantic cube. The concept extraction module is designed to identify concept in each document. This process is done with the help of the ontology collection. The terms are matched with concepts, synonyms, meronyms and hypernyms in the ontology. The concept weight is estimated with the concept and its element count. The semantic cube is constructed with concepts, semantic weight and document. In cluster processing which groups the documents, two techniques, namely, term clustering and semantic clustering technique are used. Term clustering groups documents according to the term weight, while semantic clustering groups documents according to the semantic weight. For clustering, a classical k-means algorithm is used.

5. Experimental results

This section reports experimental results when applying the selected seven algorithms to cluster text documents. During experimentation, Reuters-21578 dataset was used. More information about Reuters-21578 can be found at http://www.daviddlewis.com/resources/testcollections/reuters21578/readme.txt. To ascertain the performance of the models, several experiments were conducted. All the experiments were conducted using a Pentium IV machine with 2GB RAM. Three performance metrics, namely, purity of a cluster, F-measure and CPU execution time were used. The overall purity obtained by the selected seven algorithms for different number of clusters is shown in Table 1.

Table 1: Purity of a Cluster

No. of Clusters	K-Means	K*-Means	EM	sGEM	LPR	HSTC	TCFS
15	0.66	0.68	0.71	0.72	0.73	0.74	0.75
30	0.68	0.71	0.73	0.74	0.73	0.79	0.81
50	0.69	0.74	0.77	0.79	0.78	0.81	0.83
75	0.70	0.71	0.75	0.76	0.77	0.83	0.85
100	0.72	0.73	0.76	0.78	0.79	0.86	0.88

From Table I, it is clear that the TCFS algorithm produces quality clusters followed by HSTC when compared with other algorithms. The performance of all the three EM based algorithms with respect to purity of clusters is very similar. A k-means variant, while showing good purity (all >0.68) still is low while comparing with other algorithms. This fact is further emphasized in Figure 5, which shows the average cluster purity obtained by the algorithms.

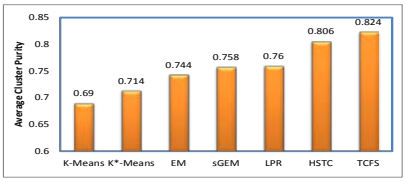

Figure 5: Average Cluster Purity

The F-measure calculated from the precision and recall is shown in Table 2.

Table 2: Accuracy of the Algorithm

	K-Means	K*-Means	EM	sGEM	LPR	HSTC	TCFS
Precision	0.515	0.525	0.634	0.639	0.641	0.694	0.698
Recall	0.832	0.844	0.877	0.880	0.881	0.899	0.902
F Measure	0.64	0.68	0.74	0.75	0.76	0.78	0.79

While considering the accuracy of the algorithm in terms of precision, recall and F measure, the trend obtained is similar to that of cluster purity. The maximum accuracy is attained by TCFS algorithm, while K-means shows the least clustering accuracy.

While considering the time taken or speed of clustering (Figure 6), it was found that the traditional K-Means algorithm was the fastest, followed by HSTC and TCFS. The EM variant models were slow when compared with all the other algorithms.

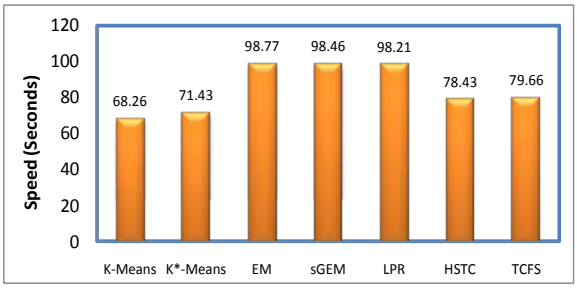

Figure 6: Speed of the Algorithms (Seconds)

All these results from the various experiments show that all the seven selected clustering algorithms produces accurate results. However, the TCFS model based on ontology produces significant improvement in clustering results when compared with other algorithms.

6. Conclusion

Increase in the volume of text data stored in digital form has increased the need for automated tool that help people find and manage these information in an efficient manner. Text clustering, which is the process of grouping documents having similar properties based on semantic and statistical content, is an important component in many information organization and management tasks. In the present research work seven approaches to document clustering was considered and their methods and performance were analyzed. The selected methods are K-Means, K*-Means, EM algorithm, sGEM, LAR, HSTC and TCFS algorithms. The first two approaches belong to the K-means family, the second three algorithms belong to the EM family, while the last two consider semantic and ontology of the text. Experiments proved that all the techniques are efficient in document clustering process, but the performance of TCFS is slightly better in terms clustering quality and the traditional K-Means algorithm is very fast in producing clustering results. In future, work that fuses the K-means variants and EM variants with ontology-based algorithms are to be performed to combine the advantage of quality clustering in a fast manner.

REFERENCES

[1] Andrews, N.O. and Edward, A. (2007) Fox, Recent Developments in Document Clustering, http://eprints.cs.vt.edu/archive/00001000/01/docclust.pdf, Last Access Date: 01-09-2011.

[2] Boley, D. (1998) Principal direction divisive partitioning, Data Mining and Knowledge Discovery, Vol. 2, No.4, Pp. 325–344.

[3] Boley, D., and Borst, V. (1999) Unsupervised clustering: A fast scalable method for large datasets. CSE Report TR99029, University of Minnesota.

[4] Bradley, P. S. and Fayyad, U.M, (1998) Refining initial points for kmeans clustering, Proceedings of the Fifteenth International Conference on Machine Learning, Pp.91–99.

[5] Cheung, Y.M. (2003) k*-Means: A new generalized k-means clustering algorithm, Pattern Recognition Letters, Vol. 24, Pp.2883-2893.

[6] Dasgupta, S., and Schulman, L. J.A (2000) A Two-Round Variant of EM for Gaussian Mixtures, *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence* (UAI '00), Craig Boutilier and Mois\&\#233;s Goldszmidt (Eds.). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, Pp.152-159.

[7] Decherchi, S., Gastaldo, P., Redi, J. and Zunino, R. (2009) K-Means Clustering for Content-Based Document Management in Intelligence, Advances in Artificial Intelligence for Privacy Protection and Security, Intelligent Information Systems, Vol. 1. Pp. 287-323.

[8] Gad, W.K. and Kamel, M.S. (2010) Incremental clustering algorithm based on phrase-semantic similarity histogram, International Conference on Machine Learning and Cybernetics (ICMLC), Pp. 2088-2093.

[9] Girolami, M. (2002) Mercer kernel based clustering in feature space, IEEE Transactions on Neural Networks, Vol.13, Pp. 2780-2784.

[10] He, J., Tan, A.H., Tan, C.L., and Sung, S.Y. (2003) On Quantitative Evaluation of Clustering Systems. In W. Wu and H. Xiong, editors, Information Retrieval and Clustering. Kluwer Academic Publishers.
[11] Kanungo, T., Mount, D.M., Netanyahu, N.S., Piatko, C.D., Silverman, R. and Wu, A.Y. (2002) Efficient Algorithms for K-Means Clustering, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 24, Pp. 881-892
[12] McLachlan, G.J. and Krishnan, T. (1997) The EM Algorithm and Extensions", John Wiley & Sons.
[13] Raja, K. and Narayanan, C.P. (2010) Clustering Technique with Feature Selection for Text Documents, Proceedings of the Int. Conf. on Information Science and Applications ICISA 2010, Pp.296--300.
[14] Rasmussen, E. (1992) Clustering algorithms, W. Frakes and R. BaezaYates (Eds.), Information retrieval: data structures and algorithms, Prentice Hall.
[15] Shawkat, A.B.W. (2008) K-means Clustering Adopting RBF-Kernel, Data Mining and Knowledge Discovery Technologies, David Taniar (Ed.), Pp. 118-142.
[16] Singh, A.K. (2004) The EM Algorithm and Related Statistical Models, Technometrics, Vol. 48, No.1, Pp.148-148.
[17] Thangamani, M. and Thangaraj, P. (2010) Integrated Clustering and Feature Selection Scheme for Text Documents, Journal of Computer Science, Vol. 6, No.5,Pp. 536-541.
[18] Zhang, S. and Pan, X. (2011) A novel text classification based on Mahalanobis distance, 3rd International Conference on Computer Research and Development (ICCRD), Pp. 156-158.

S. C. Punitha, HOD, Department of Computer Science and Information Technology, P. S. G. R Krishnammal College for Women, Coimbatore. She completed her M.Sc., MFT., M.Phil., from Bharathiar University. She is pursuing her Ph.D., in Karunya University, Coimbatore in the field of Computer Science. She has an academic experience of 14 years. Her area of research interest includes Data mining and Artificial Intelligence. She is a BOS member in various autonomous colleges.

Dr. M. Punithavalli received the Ph.D degree in Computer Scienc from Alagappa University, Karaikudi in May 2007. She is currently serving as the Director of the Computer Science Department, Sri Ramakrishna college of engineering, Coimbatore. Her research interest lies in the area of Data mining, Genetic Algorithms and Image Processing. She has published Technical papers in International, National Journals and conferences. She is Board of studies member various universities and colleges. She is also reviewer in International Journals. She has given many guest lecturers and acted as chairperson in conference. Currently she is guiding PhD scholars.