Document Clustering Using Differential Evolution

Ajith Abraham[1], Swagatam Das[2] and Amit Konar[2]

[1]IITA Professorship Program, School of Computer Science and Engineering, Chung Ang University (CAU), Seoul
[2] Dept. of Electronics and Telecommunication Engg, Jadavpur University, Kolkata 700032, India

Abstract—**This paper investigates a novel approach for partitional clustering of a large collection of text documents by using an improved version of the classical Differential Algorithm (DE). Fast and accurate clustering of documents plays an important role in the field of text mining and automatic information retrieval systems. The k-means has served as the most widely used partitional clustering algorithm for text documents. However, in most cases it provides only locally optimal solutions. In this work, the clustering problem has been formulated as an optimization task and is solved using a modified DE algorithm. To reduce the computational time, a hybrid k-means with DE method has also been proposed. The new algorithms were tested on a number of document datasets. Comparison with k-means, a state of the art PSO and one recently proposed real coded GA based text clustering methods reflects the superiority of the proposed techniques in terms of speed and quality of clustering.**

I. Introduction

Clustering of text documents plays a vital role in efficient document organization, summarization, topic extraction and information retrieval. Although 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 a user's query [4]. Document clustering has also been used to automatically generate hierarchical clusters of documents [5]. The automatic generation of a taxonomy of Web documents like that provided by Yahoo! (www.yahoo.com) is often cited as a goal.

Clustering involves the optimal partitioning of a given set of N data points into K subgroups, such that data points belonging to the same group are as similar to each other as possible whereas data points from two different groups share the maximum difference. Unsupervised document clustering may be broadly classified into two types – 'hierarchical' and 'partitional' [6]. Hierarchical techniques produce a nested sequence of partitions, with a single, all-inclusive cluster at the top and singleton clusters of individual points at the bottom. Each intermediate level can be viewed as combining two clusters from the next lower level (or splitting a cluster from the next higher level). The result of a hierarchical clustering algorithm can be graphically displayed as a tree, called a dendogram. This tree graphically displays the merging process and the intermediate clusters. In contrast to hierarchical techniques,
partitional clustering techniques create a one-level (un-nested) partitioning of the data points. If K is the desired number of clusters, then partitional approaches typically find all K clusters at once.

In recent years, it has been recognized that the partitional clustering technique is well suited for clustering a large document dataset due to their relatively low computational requirements [7, 8]. The time complexity of the partitioning technique is almost linear, which makes it widely used. The bestknown partitioning clustering algorithm is the K-means algorithm and its variants [9]. This algorithm is simple, straightforward and is based on the firm foundation of the analysis of variances. In addition to the K-means algorithm, several algorithms, including Genetic Algorithm (GA) [10, 11], Self-Organizing Maps (SOM) [12], and finally Particle Swarm Optimization (PSO) [13] have been used for document clustering.

A review of the current literature reveals that DE [14] has not been employed for text document classification till date. In the present work, we determine the optimal partitioning of a large document dataset by using an improved version of DE. Das *et al.* [15] presented an improved version of the classical DE scheme called Differential Evolution with RANDom Scale Factor (DERANDSF). In the present paper, DERANDSF with another slight modification is seen to outperform K-means, a state of the art version of genetic algorithm (GA), and an improved particle swarm optimization (PSO)-based hard clustering algorithm in terms of accuracy, speed and robustness when applied to document clustering. We have also proposed a hybrid clustering algorithm based on k-means algorithm with the modified DE, which improves considerably in terms of computational speed. To compare the performance of the various clustering algorithms, we have used a test-suit of six well known document databases in which the number of documents range from 600 to 1700, number of classes range from 7 to 25 and the number of terms per document may be as high as 12,000. Our experimental results indicate that the improved version of DE peformed very well in a statistically significant manner when compared to other algorithms considered in majority of cases.

The rest of the paper is organized as follows. In Section II, we briefly describe the methods of representing text documents as data points in a multi-dimensional space and also formulate the document clustering as an optimization problem. Section III introduces the modified DE algorithm while Section IV discusses its use in the domain of document clustering. Section V provides the detailed experiment set-up, other algorithms considered, description of the test datasets and simulation strategies. In Section VI we present the results of the experimental study and Section VII makes a number of observations based on them. Finally, the paper is concluded in Section VIII.

II. Text Document Clustering Problem

A. Representation of Documents

To apply any clustering algorithm on a dataset, documents must at first be represented in a suitable form. Documents are represented by the widely used vector-space model introduced by Salton *et al.*[16]. In this model, each document is treated as a vector $\vec{d}$. Each dimension in the vector $\vec{d}$ stands for a distinct term in the term space of the document collection. We represent each document as vector $\vec{d} = [w_1, w_2, ...w_n]$, where w_i is the term weight of the term t_i in one document. The term weight value represents the significance of this term in a document. To calculate the term weight, the occurrence frequency of the term within a document and in the entire set of documents must be considered. The most widely used weighting scheme combines the Term Frequency with Inverse Document Frequency (TF-IDF) [18, 19]. The weight of term i in document j is given by

$$w_{ji} = tf_{ji} \times idf_{ji} = tf_{ji} . \log_2 (\frac{N}{df_{ji}}) \tag{1}$$

where tf_{ji} is the number of occurrences of term i in the document j; df_{ji} indicates the term frequency in the collections of documents; and n is the total number of documents in the collection. This weighting scheme discounts the frequent words with little discriminating power.

B. Similarity Metric

To use a clustering algorithm we need to judge the similarity between two documents in some way. We used the cosine distance, which is represented as

$$\cos(\vec{d}_1, \vec{d}_2) = \vec{d}_1 \bullet \vec{d}_2 / \|\vec{d}_1\| . \|\vec{d}_2\| \tag{2}$$

where $\bullet$ denotes the 'dot product' and $\| \ \|$ denotes the norm of a vector. Usually to negotiate documents of various lengths the document vectors are normalized to unit length. We also define a centroid vector m for each set S of documents and their corresponding vector representations. It is given by,

$$\vec{m} = \frac{1}{N} \sum_{\vec{d} \in S} \vec{d} \tag{3}$$

where N is the number of documents in dataset S. We note that calculating the similarity of a document and a cluster centroid is equivalent to calculating the average similarity between that document and all the documents, which are contained in the cluster the centroid represents. Mathematically,

$$\vec{d}_i \bullet \vec{m} = \frac{1}{N} \sum_{\vec{d} \in S} \vec{d}_i \bullet \vec{d} = \frac{1}{N} \sum_{\vec{d} \in S} \cos(\vec{d}_i, \vec{d}) \tag{4}$$

C. A Formal Statement of the Document Clustering Problem

Let $S = \{d_1, d_2 ... d_n\}$ be a set of n document vectors, each having p components. These vectors can also be represented by a profile data matrix $\mathbf{Z}_{n \times p}$ having n p-dimensional row vectors. The i^{th} row vector $\vec{Z}_i$ characterizes the i^{th} object from the set S and each element $z_{i,j}$ in $\vec{Z}_i$ corresponds to the j^{th} component (j = 1, 2, ...,p) of the i^{th} vector (i =1,2,...., n). Given such a $\mathbf{Z}_{n \times p}$, a partitional clustering algorithm tries to find a partition $C = \{C_1, C_2, ..., C_k\}$ such that the similarity of the vectors in the same cluster C_i is maximum and patterns from different clusters differ as far as possible. The partitions should maintain the following properties:

1) Each cluster should have at least one vector assigned. i.e., $C_i \neq \emptyset \quad \forall i \in \{1,2,...k\}$
2) Two different clusters should have no document vector in common. i.e., $C_i \cap C_j = \emptyset, \quad \forall_i \neq j$ and $i, j \in \{1,2,...,k\}$
3) Each pattern should definitely be attached to a cluster. i.e. $\bigcup_{i=1}^{k} C_i = S$

Since the given dataset can be partitioned in a number of ways maintaining all of the above properties, a fitness function (some measure of the adequacy of the partitioning) must be defined. Then the problem turns out to be one of finding a partition $\mathbf{C}^*$ of optimal or near-optimal adequacy as compared to all other feasible solutions $C = \{C^1, C^2 ... C^{N(n,k)}\}$ where

$$N(n,k) = \frac{1}{k!} \sum_{i=1}^{k} (-1)^i \binom{k}{i} (k-i)^i \tag{5}$$

is the number of feasible partitions. This is the same as

$$\underset{\mathrm{C}}{\text{Optimize}} \ f(Z_{n \times p}, C) \tag{6}$$

where C is a single partition from the set C and f is a statistical-mathematical function that quantifies the goodness of a partition on the basis of the distance measure of the patterns. It has been shown in [20] that the clustering problem is NP-hard when the number of clusters exceeds 3.

III. Modified Differential Evolution

In 1995, Storn and Price made an attempt to replace the classical crossover and mutation operators in GA by alternative operators [14], and found a suitable vector differential operator to handle the problem. They proposed a new algorithm based on this operator, and called it Differential Evolution (DE). DE begins with a randomly initialized population of p-dimensional real-valued parameter vectors. Each vector, also known as a 'genome' or 'chromosome', forms a candidate solution to the multi-dimensional optimization problem. The initial population (at time t = 0) is chosen randomly and should be representative of as much of the search space as possible. Subsequent generations in DE can be represented by discrete time steps: t = 1, 2, ..., etc. Since the parameter vectors are likely to be changed over different generations the following notation has been adopted here for representing the i^{th} vector of the population at the current generation (at time t):

$$\vec{X}_i(t) = [X_{i,1}(t), X_{i,2}(t), ...X_{i,p}(t)] \quad (7)$$

For each individual vector $\vec{X}_k$, belonging to current population, DE randomly samples three other individuals $\vec{X}_i$, $\vec{X}_j$ and $\vec{X}_m$ from the same generation (for distinct k, i, j and m). It then calculates the component wise difference $\vec{X}_i - \vec{X}_j$, scales it by a scalar R (ϵ [0,1]) and creates a trial offspring vector by adding the result to the chromosomes of $\vec{X}_m$. Thus, for the n^{th} component of each parameter vector, we have

$$\begin{aligned} U_{k,n}(t+1) &= X_{m,n}(t) + R.(X_{i,n}(t) - X_{j,n}(t) \text{ if } rand_n(0,1) < CR \\ &= X_{k,n}(t), \qquad \text{otherwise} \end{aligned} \quad (8)$$

where CR (ϵ[0,1]) is the crossover constant. To keep the population size constant over subsequent generations, the next step of the algorithm calls for 'selection' to determine which one between the parent and child will survive in the next generation (i.e. at time $t+1$). DE uses the Darwinian principle of "survival of the fittest" in its selection process which may be expressed as

$$\begin{aligned} \vec{X}_i(t+1) &= \vec{U}_i(t+1) \quad \text{if } f(\vec{U}_i(t+1)) < f(\vec{X}_i(t)) \\ &= \vec{X}_i(t) \quad \text{if } f(\vec{X}_i(t)) < f(\vec{U}_i(t+1)) \end{aligned} \quad (9)$$

where $f(.)$ is the function to be minimized. If the new offspring yields a better value of the fitness function, it replaces its parent in the next generation; otherwise the parent is retained in the population. Hence the population either gets better (with respect to the fitness values) or remains the same but never deteriorates. To improve the convergence properties of DE, we have tuned its parameters in two different ways. In the original DE [10] the difference vector (G_i - G_j) is scaled by a constant factor 'R'. The usual choice for this control parameter is a number between 0.4 and 1. We propose to vary this scale factor in a random manner in the range (0.5, 1) by using the relation

$$R = 0.5*(1+ \text{rand}(0, 1)) \quad (10)$$

where rand (0, 1) is a uniformly distributed random number within the range [0, 1]. The mean value of the scale factor is 0.75. This allows for stochastic variations in the amplification of the difference vector and thus helps retain population diversity as the search progresses. At the same time we decrease the crossover rate CR linearly with time from CR_{max} = 1.0 to CR_{min} = 0.5. If CR = 1.0, it means that all components of the parent vector $\vec{X}_k$ are replaced by the difference vector operator. But at the later stages of the optimizing process, if CR be decreased, more components of the parent vector are then inherited by the offspring. Such a tuning of CR helps to explore the search space exhaustively at the beginning, but adjust the movements of trial solutions finely during the later stages of search, so that they can explore the interior of a relatively small space in which the suspected global optimum lies. The time-variation of CR may be expressed as follows

$$CR = (CR_{max} - CR_{min}) * (MAXIT-iter) / MAXIT \quad (11)$$

where CR_{max} and CR_{min} are the maximum and minimum values of CR, *iter* is the current iteration number and *MAXIT* is the maximum number of allowable iterations.

IV. DE Based Document Clustering

In the past several years, DE and its variants have been proven both effective and quick to solve some optimization problems [21]. It was successfully applied in many difficult NP hard optimization problems [22, 23]. In document clustering research area, it is possible to view the clustering problem as an optimization problem that locates the optimal centroids of the clusters rather than to find an optimal partition. This view offers us a chance to apply DE algorithm for clustering problems.

A. *Chromosome Encoding*

To search for the globally optimal solution to a problem using DE, parameters of the problem have to be represented by real numbers. In the proposed method, if there are n document vectors, each p-dimensional, and if the user-specified number of clusters is k then each chromosome is a vector of real numbers of dimension $k \times p$. The entries are reserved for k number of p-dimensional cluster centers. The i^{th} chromosome is represented as:

$\vec{X}_i(t) =$	$\vec{m}_{i,1}$	$\vec{m}_{i,2}$	....	$\vec{m}_{i,k}$

An example of the chromosome-encoding scheme is illustrated as follows. Let p = 2 and k = 3, i.e., the space is two dimensional and the number of clusters being considered is three. Then the chromosome

$\mathbf{X}_i$ (t) = [61.6 75.3 19.3 10.7 18.3 30.2], represents the three cluster centers (61.6, 75.3), (19.3, 10.7) and (18.3, 30.2).

B. *Fitness Function*

To judge the quality of a partition provided by a chromosome, it is necessary to have a well-defined fitness function or cluster validity index, which will take care of the following aspects of the partitioning:

1) ***Cohesion***: patterns in one cluster should be as similar to each other as possible. The fitness variance of the patterns in a cluster is an indication of the cluster's compactness.

2) ***Separation***: clusters should be well separated. A distance measure among the cluster centers (may be their cosine distance) gives an indication of cluster separation. Suppose there are k clusters in a dataset. The center of any cluster i is denoted as $\vec{m}_i$ and n_i indicates the number of data points belonging to the i^{th} cluster. Then we first define the following function to judge the quality of a clustering solution. The lower the value of this function, the better is the partitioning of the given dataset

$$CS(k) = \frac{\frac{1}{k}\sum_{i=1}^{K}[\max_{j\in K, j\neq i}\{\cos(\vec{m}_i, \vec{m}_j)\}]}{\frac{1}{k}\sum_{i=1}^{K}[\frac{1}{n_i}\sum_{\vec{d}_j\in C_i}\cos(\vec{d}_j, \vec{m}_i)]}$$

$$= \frac{\sum_{i=1}^{K}[\max_{j\in K, j\neq i}\{\cos(\vec{m}_i, \vec{m}_j)\}]}{\sum_{i=1}^{K}[\frac{1}{n_i}\sum_{\vec{d}_j\in C_i}\cos(\vec{d}_j, \vec{m}_i)]} \qquad (12)$$

This cluster validity index is inspired by the work reported in [24] and has been suitably modified for text document clustering problems. It simultaneously takes care of the cohesion and separation factors into account while dealing with complex structure data sets. In subsequent sections, we will refer to the index as 'CS measure'. Finally, the fitness function for the evolutionary algorithms is given as,

$$fitness(\vec{X}_i) = \frac{1}{CS_i(k) + eps} \qquad (13)$$

where CS_i (k) is the function given in (11) of the ith chromosome and *eps* is a very small valued constant (here it is 0.0002). Therefore, maximization of this function actually means the minimization of the index defined in (11). This measure is a function of the ratio of the sum of within-cluster scatter to between-cluster separation.

C. Pseudo Code

The pseudo code for the complete algorithm for document clustering is given below:

```
Step 1: Initialize each chromosome to contain k different
        document vectors from the document collection as
        the initial cluster centroid vectors.
Step 2: For each chromosome in the population do
        (a) Assign each document vector in the document
            set to the closest centroid vector.
        (b) Using the crossover and mutation schemes
            described in (8), (9) and generate offspring of the
            current chromosome.
        (c) Calculate the fitness of the offspring and the
            parent chromosome using equation (12) and (13)
            and replace the parent from the current population
            if its fitness is lower.
Step 3: Repeat step (2) until a predefined maximum number
        of fitness function evaluation is exceeded.
Step 4: Report as the final solution the cluster centers and
        the partition obtained by the best chromosome
        (one yielding the highest value of the fitness
        function) at time $t = t_{max}$.
```

D. The Hybrid K-means-DE Clustering Algorithm

From our initial experiments, we found that while clustering large document datasets traditional evolutionary algorithms like the real-coded GA, PSO or DE take a large number of fitness *Function Evaluations* (FE) to produce acceptable solutions. Although the *k*-means is faster, it falls short of accuracy due to trapping in local optima. Real world IR applications, however, require categorization of large document collections at a rapid pace besides maintaining the quality of clustering. In order to make use of evolutionary techniques feasible with practical text mining problems we need a trade off between speed and accuracy of the clustering algorithm. We propose a synergism of the modified DE scheme with the classical *k*-means clustering technique to take the advantages of both techniques. The hybrid algorithm includes two modules, the DE module and the *k*-means module. At the initial stage, the DE module is executed for a short period (for 20,000 FEs) to discover the vicinity of the optimal solution by a global search and at the same time to minimize high computation. The result from the DE module is used as the initial seed of the *k*-means module. The k-means algorithm is further applied for refining and generating the results. The process can be summarized as follows:

```
Step 1: Start the DE clustering process until the maximum
        number of fitness function evaluation is exceeded
Step 2: Inherit clustering result from the DE module as the
        initial centroid vectors of k-means module.
Step 3: Start k-means process until the mean difference of
        the cluster centers between two successive
        iterations falls below a predefined threshold.
```

V. Experiment Setup

A. Datasets Used and Data Preprocessing

We used five different document collections [25-26] to compare the performance of all the algorithms as summarized in Table 1. The *hitech* dataset contained documents about computers, electronics, health, medical, research, and technology; while the dataset *wap* is from the WebACE project. The i*nspec1* dataset was derived from a scientific database and the documents are on the topics of back-propagation, fuzzy control, and pattern classification. The *fbis* dataset is from the Foreign Broadcast Information Service data of TREC-5, and finally the la1 dataset was obtained from articles of the Los Angeles Times that was used in TREC-5. We use the bag-of-words representation and define each term as a distinct word in the set of words of our test-document collection. To obtain the document vectors, each document is parsed, non-alpha characters and mark-up tags are discarded, case-folding is performed, i.e. all characters are converted to the same case (to lower-case) and stop words, i.e. words such as "an", "the" and "they" that are very frequent and do not have discriminating power are eliminated. The list of 571 stop words used in the Smart system is used [16, 17]. In order to define words that are in the same context with the same term and consequently to reduce dimensionality, we stemmed the words by using Porter's Stemming Algorithm [9], which is a commonly used algorithm for word stemming in English.

TABLE I.
DATASETS USED FOR EXPERIMENTS

Dataset	Source	Number of documents	Number of terms per document	Number of clusters
hitech	San Jose Mercury (TREC)	767	7499	6
inspec1	Scientific Database	920	11803	3
Wap	WebACE	780	7131	20
fbis	FBIS (TREC)	821	1997	17
la1	LA Times (TREC)	801	8449	6

TABLE II.
PARAMETER SETUP FOR DIFFERENT ALGORITHMS

G3 with PCX		HPSO-TVAC		Modified DE	
Parameter	Value	Parameter	Value	Parameter	Value
Popsize	100	Popsize	40	Pop_size	10*dim
σ_η	0.1	Inertia weight	0.794	CR_{max}	1.0
σ_ξ	0.1	C_1	Linearly varying 0.35→2.4	CR_{min}	0.5
μ	3	C_2	Linearly varying 2.4→0.35	Scale factor R	Uniformly distributed random number between 0.5 and 1.0 with mean value 0.75
λ	2	V_{max}	3.00		
		Re-initialization velocity	Linearly decaying from V_{max} to 0.1 V_{max}		

B. Algorithms Compared

In this paper, we compared the performance of the celebrated *K*-means clustering, a real coded GA based document clustering and a PSO based clustering algorithm with the method proposed by us. As a real coded GA we have used the Generalized Generation Gap (G3) model with Parent Centric Recombination (PCX) [26]. We have also employed a state of the art version of PSO known as Self Organizing Hierarchical PSO with Time Varying Accelaration Coefficients (HPSO-TVAC) [27]. Table 2 lists all the parameter settings used for all the algorithms.

C. Evaluating the Quality of Clustering

Apart from the modified CS measure, we also used entropy to measure of quality of the final partitioning (with the idea that the best entropy is obtained when each cluster contains exactly one data point) yielded by the competitor algorithms. Entropy is calculated over the results obtained after each run of an optimization algorithm is terminated. 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 (14)

$$E_j = -\sum_i p_{ij} \log(p_{ij}) \quad (14)$$

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

$$E_{CS} = \sum_{j=1}^{m} \frac{n_j * E_j}{n} \quad (15)$$

D. Simulation Strategy

All the optimization algorithms used here, are stochastic in nature. Hence, for each problem they have been run several times. The results have been stated in terms of the mean values and standard deviations over 20 runs in each case. Since the population size for DE, PSO and GA are markedly different; we choose number of fitness evaluations as a measure of computation time instead of 'generations' or 'iterations'. Number of Fitness Evaluations (FE) roughly equals the product of the population size and the number of generations. Here the three optimization-based algorithms were allowed to run for 50,000 FEs. Only for the k-means, we terminate a run as soon as the mean change of the cluster centers between two successive iterations falls below a specified threshold value of 0.001. For the hybrid algorithm, at first the modified DE is run for 20, 000 FEs and then the k-means is started taking the initial seeds from the previous DE run. The k-means is terminated following the same stopping condition discussed above. Finally, we would like to point out that all the algorithms were developed from scratch in Visual C++ platform on a Pentium IV, 2.2 GHz PC, with 512 KB cache and 2 GB of main memory in Windows Server 2003 environment.

VI. EXPERIMENT RESULTS

Table III summarizes the clustering results for five datasets in terms of the average of 'best of the run' CS measures obtained after the termination of 20 independent runs. Each run was continued up to 50,000 FE. Table IV indicates the mean entropy (of 20 runs) obtained over the final clustering results given by each of the algorithm. A comparison of these two tables shows that if an algorithm yields a higher fitness function value then its entropy is also better (i.e. smaller). Hence, the fitness function we have designed with a modified CS measure provides an acceptable measure of the clustering quality. The best entries of tables III and IV have been marked in boldface.

TABLE III.
CS MEASURE: PERFORMANCE COMPARISON

Method	Mean CS Value *(standard Deviation)*				
	inspec1	hitech	Wap	fbis	la1
K-means	0.5422 (0.0029)	0.8554 (0.0003)	0.6421 (0.0006)	0.9492 (0.0039)	0.9302 (0.0631)
Modified DE	**0.1445 (0.0022)**	**0.3443 (0.0366)**	**0.1844 (0.0017)**	0.3372 (0.0012)	**0.5343 (0.0058)**
HPSO-TVAC	0.7489 (0.0071)	0.7754 (0.0093)	0.6978 (0.0656)	0.6205 (0.0078)	0.7493 (0.0482)
G3 with PCX	0.5364 (0.0262)	0.6485 (0.0093)	0.3776 (0.0078)	**0.2879 (0.0043)**	0.7383 (0.0749)
K-means with DE	0.4905 (0.0386)	0.3832 (0.0039)	0.2284 (0.0071)	0.3551 (0.0006)	0.8035 (0.0089)

TABLE IV
ENTROPY MEASURE: PERFORMANCE COMPARISON

Method	Final Entropy Value (standard deviation)				
	inspec1	hitech	Wap	fbis	la1
K-means	0.2994 (0.0038)	0.9432 (0.0495)	0.5294 (0.0049)	0.9932 (0.0034)	0.4986 (0.0037)
Modified DE	**0.2110 (0.0091)**	**0.5478 (0.0075)**	**0.4110 (0.0029)**	0.6336 (0.0819)	**0.3772 (0.0119)**
HPSO-TVAC	0.2584 (0.0043)	0.7863 (0.0037)	0.4984 (0.0048)	0.7123 (0.0046)	0.5108 (0.0065)
G3 with PCX	0.2321 (0.0110)	0.7303 (0.0362)	0.4621 (0.0210)	**0.6145 (0.0057)**	0.4281 (0.0070)
K-means with DE	0.2294 (0.0029)	0.6002 (0.0011)	0.4328 (0.0871)	0.6930 (0.0076)	0.4082 (0.0096)

TABLE V
RESULTS OF UNPAIRED T-TESTS

Data set	Std. Err	*t*	95% Conf. interval	Two-tailed *P*	Significance
inspec1	0.002	8.6156	-0.0227 to -0.0143	< 0.0001	**Extremely significant**
hitech	0.002	30.9146	-0.05583 to -0.0489	< 0.0001	**Extremely significant**
Wap	0.008	2.6198	-0.0386 to -0.0049	0.0126	**Significant**
fbis	0.018	1.0404	-0.216 to -0.210	0.3047	**Not Significant**
la1	0.003	9.0674	-0.0379 to -0.0240	< 0.0001	**Extremely significant**

Table V illustrates the results of unpaired *t*-tests taken based on the fitness value between the best algorithm and the second best in each case (standard error of difference of the 2 means, 95% confidence interval of this difference, the *t* value, and the two-tailed *P* value). For all cases in Table IV, sample size = 20 and degrees of freedom = 98 and is a summary of 20 independent runs. Since all the datasets used here have their nominal partitions known to the user, the present work also computes the mean number of correctly classified documents for each of the algorithms. This is the average number of documents that were assigned to clusters according to the nominal classification. Figure 1 illustrates the average classification accuracy obtained over 20 runs for the five datasets by all the algorithms. Figures 2, 3 and 4 illustrate the performance of the other clustering methods over the *hitech*, *Wap* and *la1* datasets. Table VI depicts the mean execution time for 20 independent runs (with standard deviations) for each algorithm applied to the benchmark datasets.

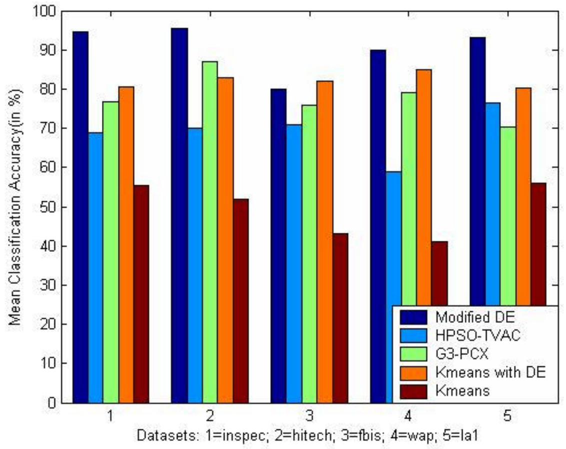

Figure 1. Mean classification accuracy obtained using all the algorithms

We also tested the evolutionary text document clustering algorithms on the results from Web search engines. We downloaded documents returned from the Google search engine. After that, we clustered the documents using our algorithm and displayed the results. Due to lack of space, we have to skip the details of the entire results. Here we illustrate some clusters that were obtained by employing the *k*-means with DE based clustering scheme on the top 500 URLs returned from searching the term "Madonna". The categories correspond to the common usage of the word "Madonna" in documents over the Web (name of an eminent pop artist, artworks, Virgin Mary, Catholic religion etc). Table VII shows some of the clusters formed using the hybrid of K-means and DE.

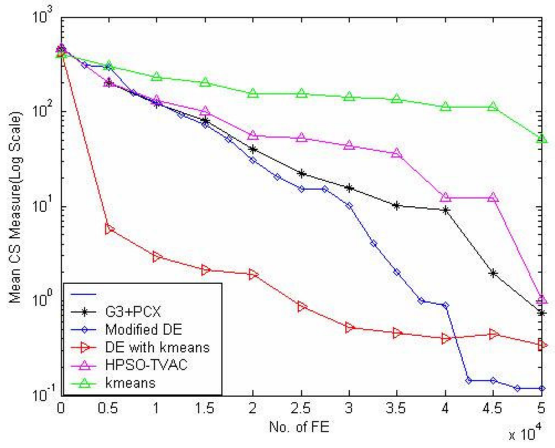

Figure 2. Convergence of the different algorithms for 'hitech' dataset

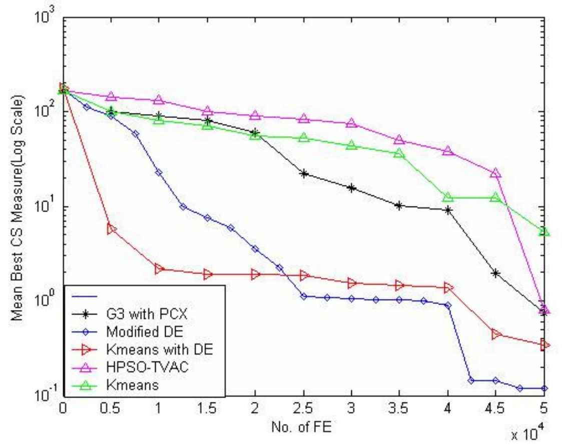

Figure 3. Convergence of the different algorithms for 'Wap' dataset

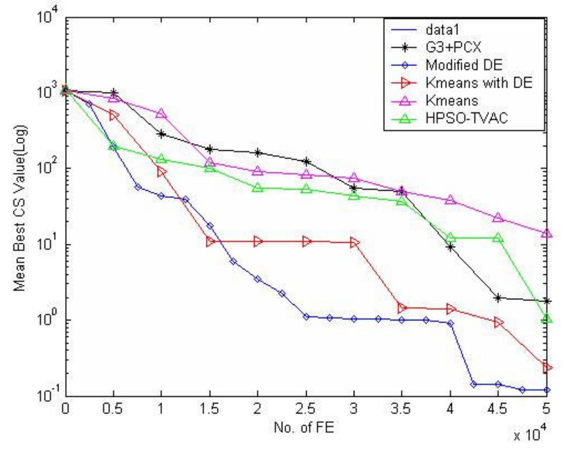

Figure 4. Convergence of the different algorithms for 'la1' dataset

VII. DISCUSSIONS ON THE RESULTS

A scrutiny of Tables III and IV reveals that on the majority of cases the modified DE based clustering algorithm can outperform the classical K-means, the G3 model with Parent centric Recombination and the HPSO-TVAC algorithm when applied to document categorization problem. We note that only in one case (for 'fbis' dataset) the modified DE method's mean entropy is numerically larger (i.e., worse) than the mean of the GA based method, but as Table V shows, this difference is *not* statistically significant. It is interesting to see from Tables V that the proposed DE based method outperformed all the algorithms in a statistically meaningful way for all the remaining cases. The performance of the real coded GA based method has been found to be superior to the HPSO-TVAC algorithm. Perhaps the performance of the later algorithm can be improved by more tuning of its parameters. From Tables III and IV we observe that the performance of the classical *k*-means is 'worst of the lot' in terms of accuracy. Frequent trapping in locally optimal portions of the search space can be one reason for the loss of quality in *k*-means clustering. But at the same time, it can be observed from Table VI; *k*-means converges fastest and is computationally the least expensive. The hybrid DE--*K*-means algorithm combines the ability of global searching by the modified DE algorithm and the fast convergence of the *k*-means algorithm and avoids the drawback of both algorithms. The result from the DE based global search is used as the initial seed of the *k*-means module and the *k*-means algorithm is applied for refining and generating the final results. Our experimental results illustrate that using this hybrid algorithm can generate higher compact clustering than using *k*-means alone while maintaining its speed close to that of the *k*-means. It provides a reasonable trade-off between the accuracy and computational time so as to make the scheme feasible for real world applications.

TABLE VI.
EXECUTION TIME PERFORMANCE

Dataset	Algorithm	Mean Time taken for a single iteration (milliseconds)	Mean time for one experiment (sec)	Time for k-means (sec)	Time for k-means with DE (sec)
hitech	Modified DE	56.95 (0.3229)	78.655 (1.2339)	16.585 (0.0029)	20.235 (0.0483)
	HPSO-TVAC	229.85 (0.9451)	183.705 (0.3821)		
	G3 with PCX	246.35 (0.6129)	228.75 (1.5164)		
inspec 1	Modified DE	24.85 (1.2529)	39.850 (2.8937)	9.775 (0.0048)	13.875 (0.0493)
	HPSO-TVAC	330.95 (0.2738)	365.70 (0.2326)		
	G3 with PCX	769.65 (4.2849)	407.85 (2.8391)		
Wap	Modified DE	76.95 (0.5821)	94.005 (1.3928)	40.875 (0.0493)	42.785 (0.8329)
	HPSO-TVAC	90.65 (0.4182)	397.405 (2.5732)		
	G3 with PCX	94.75 (5.3010)	436.915 (2.3029)		
fbis	Modified DE	97.65 (1.5738)	508.305 (2.1937)	50.980 (0.0329)	56.395 (0.0291)
	HPSO-TVAC	568.45 (3.2912)	618.150 (1.8738)		
	G3 with PCX	776.80 (4.3939)	646.815 (2.9489)		
la1	Modified DE	39.50 (0.8229)	58.015 (0.2239)	21.750 (0.0073)	28.130 (0.9821)
	HPSO-TVAC	159.85 (0.2451)	123.515 (0.9782)		
	G3 with PCX	142.85 (0.9829)	128.675 (0.5982)		

TABLE VII.
CLUSTER RESULTS FOR "MADONNA"

Cluster results: keywords and sample documents	Related topic
Filmography, Awards, Agent, Photos, Fan club, Biography, Functions 1. Madonna – the queen of pop 2. I'll Never be an Angel	Madonna – the eminent pop artist and singer
Virgin, Mary, Mother, Joseph, Lourdes, Fatima, Catholic, Marian Statues, blessed 1. Purification of Mary 2. Virgin of the charity of Cobre	Madonna - Virgin Mary
House, Gospels, Cardinal, Angels, Saint John, Publications 1. Black Madonna pilgrimage 2. The Irish Madonna of Hungary	Madonna related to Catholic religion
Collectables, Rare items, Posters, Raffaelo, Paintings, Statues, Sculpture, Museum, Michelangelo - Madonna tribe: two madonna artworks by Krzysztof - Religious stained glass: Madonna and child	Madonna related to artworks and sculptures

VIII. CONCLUSIONS

This paper proposed a modified mutation scheme for the classical DE (DE/rand/1/bin) scheme to improve its convergence properties. We used the modified DE algorithm to categorize a few well-chosen document collections, which represent many complexities of the practical information retrieval problems. We also have proposed a new validity index especially suited for high dimensional document clustering problems by modifying the CS measure. We have compared our algorithm with the classical *k*-means and state of the art versions of the real coded GA and PSO algorithms using a five document-dataset test suite, on the following performance metrics: (a) clustering quality, (b) speed of convergence, (c) misclassification error, and (d) robustness. Finally, to exploit the accuracy provided by the modified DE based global search and the low computational cost of the *k*-means algorithm; we have proposed a hybrid *k*-means algorithm, which refines on the initial seeds resulting from a short-lived DE based search. The hybrid scheme is seen to yield accuracy close to a stand-alone DE based clustering while maintaining the speed comparable to a stand-alone K-means algorithm.

ACKNOWLEDGEMENTS

This research was supported by the International Joint Research Grant of the IITA Institute of Information Technology Assessment) foreign professor invitation program of the MIC (Ministry of Information and Communication), Korea.

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: ACluster-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 WebDocuments, 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] A. K. Jain, M. N. Murty, and P. J. Flynn, 1999. *Data Clustering: A Review, ACM Computing Survey*, Vol. 31, No. 3, pp. 264-323.

[7] M. Steinbach, G. Karypis, V. Kumar, 2000. A Comparison of Document Clustering Techniques. TextMining Workshop, KDD.

[8] Y. Zhao and G. Karypis, 2004. Empirical and Theoretical Comparisons of Selected Criterion Functions for Document Clustering, Machine Learning, 55 (3): pp. 311-331.

[9] J. A. Hartigan 1975. *Clustering Algorithms*. John Wiley and Sons, Inc., New York, NY.

[10] C. Santimetvirul and P. Willett, 1995. Non-hierarchic document clustering using a genetic algorithm. Information Research, 1(1).

[11] V.V. Raghavan and K. Birchand, 1979. A clustering strategy based on a formalism of the reproductive process in a natural system. Proceedings of the Second International Conference on Information Storage and Retrieval, 10–22.

[12] D. Merkl, 2002. Text mining with self-organizing maps. Handbook of data mining and knowledge, pp. 903-910, Oxford University Press, Inc. New York.

[13] C. Xiaohui, T.E. Potok, P. Palathingal *Document Clustering using Particle Swarm Optimization,* IEEE Swarm Intelligence Symposium, The Westin Pasadena, Pasadena, California, 2005.

[14] R Storn, K Price, Differential evolution – A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces, Journal of Global Optimization, 11(4) (1997) 341–359.

[15] S. Das, A. Konar, U. K. Chakraborty, Two Improved Differential Evolution Schemes for Faster Global Search in *ACM-SIGEVO Proceedings of Genetic and Evolutionary Computation Conference (GECCO-2005)*, Washington DC, June, 2005.

[16] G. Salton and C. Buckley, 1988. Term-weighting approaches in automatic text retrieval. Information Processing and Management, 24 (5): pp. 513-523.

[17] *ftp://ftp.cs.cornell.edu/pub/smart/ (accessed on April 13, 2006)*

[18] B. Everitt,, 1980. Cluster Analysis. 2nd Edition. Halsted Press, USA.

[19] G. Salton, 1989. Automatic Text Processing. Addison-Wesley.

[20] S. Paterlini, and T. Minerva, 2003. *Evolutionary Approaches for Cluster Analysis.* In A. Bonarini, F. Masulli, G. Pasi (eds.) Soft Computing Applications. Springer-Verlag, Berlin. 167-178.

[21] T. Rogalsky, S. Kocabiyik, and R. Derksen,, "Differential evolution in aerodynamic optimization," *Canadian Aeronautics and Space Journal,* vol. 46, no. 4, pp. 183–190, 2000.

[22] G. Stumberger, D. Dolinar, U. Pahner and K. Hameyer, Optimization of radial active magnetic bearings using the finite element technique and differential evolution algorithm, *IEEE Transactions on Magnetics*, vol. 36, no. 4, pp.1009–1013, 2000.

[23] I. Zelinka and J. Lampinen, "An evolutionary learning algorithms for neural networks," Proc. of 5th International Conference on Soft Computing, MENDEL'99, pp.410–414, 1999.

[24] C.H Chou,, M.C Su, E. Lai,, 2004. A new cluster validity measure and its application to image compression. Pattern Analysis and Applications 7(2), 205-220.

[25] TREC. 1999. Text Retrieval Conference. http://trec.nist.gov (*(accessed on April 13, 2006)*

[26] J. Moore, E. Han, D. Boley, M. Gini, R. Gross, K. Hastings, G. Karypis, V. Kumar, and B. Mobasher. Web page categorization and feature selection using association rule and principal component clustering. In 7th Workshop on Information Technologies and Systems, 1997.

[27] K.Deb, A. Anand and D. Joshi (2002). A Computationally Efficient Evolutionary Algorithm for Real-Parameter Optimization, Evolutionary computation, 10(4), pp. 371 – 395.

[28] A. Ratnaweera, K, S. Halgamuge,: Self organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients. IEEE Trans. on Evolutionary Computation (2004) 8(3): 240-254.