Expert Systems With Applications 178 (2021) 115040

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

A hybrid approach for text document clustering using Jaya optimization algorithm

Karpagalingam Thirumoorthy *, Karuppaiah Muneeswaran

Department of Computer Science and Engineering, Mepco Schlenk Engineering College, Sivakasi 626005, Tamilnadu, India

ARTICLE INFO

Keywords:
Document clustering
Jaya optimization
Cosine similarity
Crossover
Mutation

ABSTRACT

In this digital era, millions of Internet users are contributing vast amounts of data in the form of unstructured text documents. Organizing this material is a tedious task. The clustering of text document plays a vital role for organizing these unstructured text documents. In our paper, we make use of Hybrid Jaya Optimization algorithm (HJO) for text Document Clustering (DC), referred to as HJO-DC. We have used the Silhouette index as a metric to measure the quality of a solution. The proposed work is compared with partitioning techniques such as K-Means and K-Medoids and metaheuristic techniques such as Genetic algorithm, Cuckoo Search, Particle Swarm Optimizer, Firefly and Grey Wolf Optimizer. Remarkably, the proposed algorithm achieves the highest quality clustering in all benchmark examples.

1. Introduction

Currently, an exponential growth of data is being shared by millions of Internet users in the form of text: personal blogs, forum, chats, emails, tweets, product reviews, movie reviews, etc. These huge amounts of unstructured text data are stored in the digital form. It is a very challenging task to retrieve the most relevant document from the vast collection of text corpus in the WWW repository (Manning, Raghavan, & Schütze, 2008). The text categorization, text summarization, text clustering, text feature selection, sentiment analysis, topic modelling are the some of the research works being carried out on the unstructured text documents. Clustering is an unsupervised technique, which organizes the data into groups based on the similarities between the data. The data objects within each cluster must have a high degree of similarity. The data objects in different clusters must have a high degree of dissimilarity. In text document clustering, the unstructured text documents are grouped based on the similarity between the documents in the text corpus. The recent literature studies show that text document clustering is used in various applications such as Web Search Engine Retrieval System (Yang, 2010), finding similar documents (Al-Anazi, Almahmoud, & Al-Turaiki, 2016), duplicate content detection (Pamulaparty, Rao, & Rao, 2014) and automatic clustering of newspapers (Afonso & Gottschalg-Duque, 2014).

Han, Kamber, and Pei (2012) presented the taxonomy of the clustering as: (i) Partitioning methods, (ii) Hierarchical methods (iii) Density based methods and (iv) Grid based methods. The partitioning method splits an entire dataset of n objects into k clusters ($k < n$). The well-known partitioning methods are k-means, K-medoids, Clustering Large Applications (CLARA). Hierarchical methods seek to form a hierarchy of nested cluster. It forms the tree views of cluster as dendrogram. Some of the popular hierarchical methods are AGENS, DIANA, BIRCH, and ROCK. Density-based methods split the entire dataset into the arbitrarily shaped clusters of objects and they are based on the cluster density. Some of the popular density-based methods are CLIQUE, DBSCAN, DENCLUE and OPTICS. Some of the grid-based methods which are based on multi-resolution grid data structures are STING, GRIDCLUST and WaveCluster. In spite of the many studies published in literature, good quality clustering of compound data documents remains an open issue, which has motivated the present research.

The rest of this article is organized as follows. In Section 2, various clustering techniques are reviewed. The text document representation and similarity metrics used in our work are illustrated in Section 3. Section 4 describes the proposed hybrid Jaya optimization-based clustering algorithm. Section 5 presents the details of dataset and cluster quality evaluation metrics. The results obtained in this study are presented and discussed in Section 6. Finally, Section 7 concludes the article.

* Corresponding author.
E-mail address: kthirumoorthy@mepcoeng.ac.in (K. Thirumoorthy).

https://doi.org/10.1016/j.eswa.2021.115040
Received 19 June 2020; Received in revised form 18 September 2020; Accepted 11 April 2021
Available online 16 April 2021

2. Related Work

Manual grouping is feasible if the size of document repository is tiny. The size of document repository becomes larger by the exponential growth of data. This makes it difficult for organizing the documents in a proper way. In order to organize these documents, the role of document clustering is more important. Section 2.1 through 2.3 outline some of recent work carried out in the field of document clustering.

2.1. Partitioning methods

Lydia, Govindasamy, Lakshmanaprabu, and Ramya (2018) used the K-Means algorithm for document clustering. They have used Euclidean distance measure for finding distance between documents. Bouras and Tsogkas (2012) proposed Wordnet enabled W-K means clustering algorithm for the news articles clustering techniques. Their work improved the performance of standard K-means clustering algorithm. In order to detect the crime patterns, Bsoul, Salim, and Zakaria (2013) proposed an Intelligent document clustering approach based on the affinity propagation algorithm. The Enhanced k-Means Clustering algorithm was proposed by Haraty, Dimishkieh, and Masud (2015) fordiscovering the pattern in the healthcare.

The author Mahdavi and Abolhassani (2009) proposed a novel Harmony K-means algorithm for document clustering; they enriched the K-means algorithm performance by Harmony Search (HS) optimization. Jha (2015) used K-Medoids clustering algorithm for document clustering and summarization. In order to cluster the heterogeneous data, Harikumar and Surya (2015) used K-Medoids clustering algorithm. They proposed a triplet form similarity measure to find the distance between two heterogeneous data objects. They evaluated the quality of cluster using purity index and Davies Bouldin index.

Fahad (2016) modified the K-Means clustering algorithm, which reduced the time complexity; the focus was on setting the initial centroid for the cluster. K-Means Hadoop MapReduce algorithm was proposed by Sreedhar, Kasiviswanath, and Reddy (2017). They implemented the standard K-Means algorithm by MapReduce program to produce the quality cluster in terms of inter-cluster and intra-cluster distance. The optimized k-means algorithm was proposed by Samir, Ahmed, and Mansour (2014) for image segmentation.

2.2. Hierarchical methods

Divisive hierarchical bisecting min max clustering algorithm was proposed by Kamat (2017). The method was tested on the BBC dataset and results were compared with those of the K-means and agglomerative clustering algorithm. Kotouza and Mitkas (2019) proposed the framework for hierarchical document clustering. They have utilized the cloud computing infrastructures. Their hierarchical clustering framework uses the binary tree construction and topic modeling to form the hierarchy of documents. The minimum spanning tree based hierarchical clustering algorithm was proposed by Agarwal and Roul (2018) that used the Kruskal's algorithm. Roul, Asthana, and Sahay (2015) proposed a novel approach for document indexing via hierarchical clustering. The new algorithm was tested on the 20Newsgroup dataset. Dasgupta (2016) proposed a cost function to find similarities between data points for hierarchical clustering. Ahmadi, Gholampour, and Tabandeh (2017) presented a new document clustering based on Sparse Topical Coding. They discovered the latent clusters in documents collection. Nguyen and Shin (2019) introduced a noval density based clustering algorithm using spatio-textual information. This approach outperformed DBSCAN with heterogeneous inputs.

2.3. Evolutionary algorithms

An evolutionary optimization approach for document clustering based on genetic algorithm was proposed by Akter and Chung (2013). They used the Cosine similarity metrics for finding the similarity between the documents. The proposed clustering algorithm outperformed the K-Means clustering approach. Lubna Alhenak (2019) presented a new Genetic-Frog-Leaping Algorithm for text document clustering by combining the genetic algorithm and shuffled frog-leaping algorithm. The new algorithm was successfully tested on the 20Newsgroup dataset and clearly outperformed the other existing methods. The cognitive-inspired multi-objective automatic document clustering technique was used by Saini, Saha, and Bhattacharyya (2018). This is a combination of self-organizing map and multi-objective differential evolution approach. Comparative results in terms of DB index and Dunn index clearly demonstrated the superior performance of the proposed algorithm. Abualigah, Khader, and Hanandeh (2018) appliedthe particle swarm optimization algorithm based feature selection method to improve the text clustering. They reduced the dimension of feature space by selecting informative feature. Using the reduced subset of informative features, improved the performance of the document clustering. Metre, Popat, and Deshmukh (2017) proposed a new document clustering technique using Universal Networking Language (UNL) generative feature vector and Boundary Restricted Particle Swarm Optimization algorithm. The quantum chaotic cuckoo search(QCCS) algorithm for data clustering was proposed by Boushaki, Kamel, and Bendjeghaba (2018). The initial randomness entailed by generation of initial population was solved by using a chaotic map, which improved the convergence speed. The results obtained on six benchmark dataset proved the significant superiority of the proposed QCCS. Bouyer, Ghafarzadeh, and Tarkhaneh (2015) presented a hybrid algorithm combining the cuckoo search and differential evolution algorithms for data clustering. Filefly algorithm based document clustering was proposed by Mohammed, Yusof, and Husni (2015). They presented Weight- based Firefly Algorithm (WFA) and Weight-based Firefly Algorithm II (WFAII) for document clustering. They evaluated the performance of the proposed algorithms on 20Newsgroup dataset. The obtained results were better than those of the existing methods. A Grey Wolf optimization algorithm for text document clustering was developed by Rashaideh et al. (2018). They defined the fitness function as minimizing the average distance of documents from the cluster centroid (ADDC). Vidyadhari, Sandhya, and Premchand (2019) proposed the Particle Grey Wolf Optimizer for text document clustering. The new algorithm combined the PSO and GWO algorithm. They evaluated the performance of the new algorithm on 20Newsgroup and Reuter dataset finding better results than those of the existing methods.

The Jaya optimization algorithm was developed by Venkata Rao (2016). The Jaya algorithm is a metaheuristic optimization method that generates new trial designs always trying to approach the current best solution and to escape from the current worst solution of the population. Algorithm-specific control parameters are required for optimization methods like Genetic Algorithm(GA), particle swarm optimization (PSO), Cuckoo Search (CS), Firefly optimization (FFY) and Grey Wolf Optimizer (GWO). The proper adjustment (tuning) of the algorithm-specific parameters is essential for moving the search process towards the global optimal solution. The computational cost will be increased if

Table 1
Document term matrix.

Doc	t_1	t_2	$\ldots t_i \ldots$	t_m
d_1	w_{11}	w_{12}	w_{1i}	w_{1m}
d_2	w_{21}	w_{22}	w_{2i}	w_{2m}
⋮				
d_n	w_{n1}	w_{n2}	w_{ni}	w_{nm}

we not handle the algorithm-specific parameters in a proper way. The Jaya optimization algorithm does not require any algorithm-specific parameters. The recent literature studies show that Jaya optimization algorithm is used in various domain of engineering applications with promising results such as classification systems (Kurada & Pavan, 2016), facial emotion recognition (Wang, Phillips, Dong, & Zhang, 2018), flow shop scheduling (Buddala & Mahapatra, 2018), currency exchange prediction (Das, Mishra, & Rout, 2020), structural optimization (Degertekin, Lamberti, & Ugur, 2019), structural damage identification (Du, Vinh, Trung, Quyen, & Trung, 2018) and clustering (Suraj & Ghosh, 2016).

Although many studies on document clustering were proposed in the literature, the definition of high-quality cluster remains the most important issue. This resulted in a few studies proposing new meta-heuristic algorithms. In this work, we used the hybrid Jaya optimization algorithm for text document clustering.

3. Preliminaries

This section reviews some common aspects such as text preprocessing, document representation and similarity measures, which are shared by most of the text clustering algorithms.

3.1. Text pre-processing

The text mining process can be fast and effective by text pre-processing. The pre-processing includes lowercasing, removing numbers, punctuations, stop words, and finally stemming.

Stop words Removal: Commonly used terms (stop words) in a language are present in almost all the text documents which never help us to distinguish the text. The stop word removal process removes the uninformative stop words from the text corpus.

Numbers, Punctuation Removal: In the text mining process, the numbers and punctuation symbols in the unstructured text documents are useless, so they are removed from the text corpus.

Stemming: Stemming is the process to obtain the words to their root form. In this work, we have used the Porter Stemmer for stemming process.

3.2. Document representation

The entire text corpus is represented by a vector space model. The text corpus D is a set of unstructured text documents and is denoted as $D = \{d_1, d_2, d_3, \ldots, d_n\}$ where n is the total number of documents in the text corpus. Features are tokenized from the text corpus. They are denoted as $T = [t_1, t_2, t_3, \ldots, t_m]$ where m is the total number of terms in the text corpus. Each document $d_i, 1 \leqslant i \leqslant n$ is represented by a feature vector. Each element of the feature vector represents the weight value of the term which is extracted from the vocabulary of given text corpus and is denoted as $< w_{i1}, w_{i2}, w_{i3}, \ldots, w_{im} >$ where w_{ij} denotes the weight value

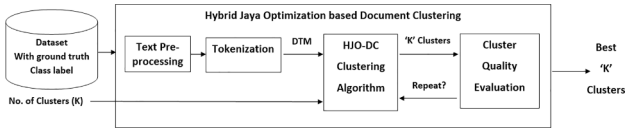

Fig. 1. Architecture design of proposed document clustering approach.

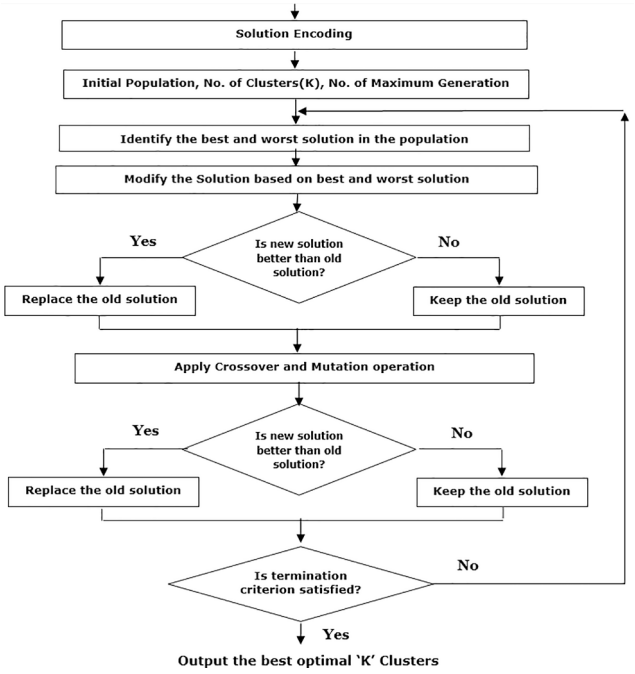

Fig. 2. Flowchart of the Hybrid Jaya Optimization algorithm.

of the term t_j in the document d_i. The document term matrix is shown in Table 1.

Term Frequency – Inverse Document Frequency (TF-IDF) is used to compute the weight of the term t_j in the document d_i.

$$TF - IDF(d_i, t_j, D) = TF(d_i, t_j) * IDF(t_j, D) \tag{1}$$

$$TF(d_i, t_j) = \frac{tf_{ij}}{\sum_{t=1}^{m} tf_{it}} \tag{2}$$

$$IDF(t_j, D) = log_2\left(\frac{n}{DF(t_j)}\right) \tag{3}$$

where DF – document frequency count. $DF(t_j)$ – number of documents containing the term t_j in text corpus.

3.3. Similarity measures

The key objective of the text document clustering is to unite the most relevant documents into the same group. Every document is represented by a TF-IDF based feature vector. Similarity between the documents is calculated by cosine similarity measure. The similarity score between the two documents is computed within the range 0 to 1. If similarity score between the two documents is 1 then that documents are similar. If similarity score between the two documents is 0 then that documents are

Table 2
Preliminary notation.

Notation	Description
n	number of documents in text corpus
N	Population Size
K	Number of clusters
k	cluster index
i	solution index
j	design variable index
t	iteration/generation index
T_{max}	Maximum number of iteration/generation
Ψ_i	i^{th} Solution
$\Psi_{i,j}$	j^{th} position(design variable index) of Solution Ψ_i
$\Psi_i^{(t)}$	i^{th} solution Ψ_i*of Iteration/Generation t*
$\Psi_i^{fitness}$	*fitness* value of solution Ψ_i
Ψ_{best}	*best* solution
$\Psi_{best}^{fitness}$	*fitness* value of best solution
Ψ_{worst}	*worst* solution
$\Psi_{worst}^{fitness}$	*fitness* value of worst solution
α, β	random integers between [1,K]

Table 3
Solution encoding description.

Solution	1	3	3	1	2	3	2	2	1	1
Document	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9	d_{10}
Cluster	C_1	C_3	C_3	C_1	C_2	C_3	C_2	C_2	C_1	C_1

completely different. ***Cosine similarity*** (Manning et al. (2008, 2012)) is the standard measure, calculate the cosine angle between the two feature vectors. Let $d_1 = (x_1, x_2, \ldots, x_n)$, *and* $d_2 = (y_1, y_2, \ldots, y_n)$ be two document vectors. The cosine similarity is defined as

$$CosineSim(d_1, d_2) = \frac{\sum_{i=1}^{n} x_i * y_i}{\sqrt{\sum_{i=1}^{n} x_i^2}\sqrt{\sum_{i=1}^{n} y_i^2}} \tag{4}$$

4. Proposed work

Let the text corpus D contain n documents and it must be grouped into K clusters, then we can coin the K^n number of possible solutions. If we work with all the solutions, increase the computation cost. In this information era, huge numbers of research works have utilized the optimization techniques in order to find a global optimal solution, optimization techniques are applied. In the proposed work, we have adapted them in the text document clustering problem domain. Fig. 1 illustrates the architecture of the proposed work. We use the Jaya optimization algorithm to form the quality clusters in the given text corpus.

4.1. Hybrid Jaya optimization algorithm

In order to find the optimal solution, the searching process of Jaya optimization algorithm is moving towards to the best solution, away from the worst solution. It has been used in different kinds of engineering problems. The working principles of Jaya optimization algorithm are shown in Fig. 2. A brief summary of the preliminary notations used in this work is shown in Table 2.

Table 4
One-point crossover operation (Cutting point: 3) – Parent Solutions.

Parent Solution 1	1	3	3	1	2	3	2	2	1	1
Parent Solution 2	2	1	2	1	3	3	2	1	3	3

4.2. Solution encoding

The candidate solution is encoded as an integer vector. The number of text documents in the text corpus decides the size of solution vector. Each element in the vector represents an index of the cluster. Let the text corpus contain 10 documents $\{d_1, d_2, \ldots, d_{10}\}$ and need to be grouped into 3 clusters $\{C_1, C_2, C_3\}$. Let the sample solution vector is $\Psi_i = [1,3,3,1,2,3,2,2,1,1]$. It represents that documents $\{d_1, d_4, d_9, d_{10}\}$ are grouped into cluster C_1, documents $\{d_5, d_7, d_8\}$ are grouped into cluster C_2 and documents $\{d_2, d_3, d_6\}$ are grouped into cluster C_3. The description of the solution vector is shown in Table 3.

4.3. Initial population

Population is the subset of solutions in the global solution space of optimization problem. Let N be the population size. Randomly, we generate N solutions with random positive numbers between 1 and K (K is the number of clusters). After that, the validation process is applied to verify each solution. Validation process ensures that each cluster contains more than one documents. If a cluster does not have more than one document then validation process rejects that solution and generates the new solution randomly.

4.4. Fitness function

Fitness function plays a crucial role in the optimization problems. The fitness function calculates a single positive integer to depict how good the solution is. We have used the Silhouette Index as a maximizing objective function to measure the fitness of each individual solution in the population, which is formulated in Eqs. (5)–(7).

Silhouette Coefficient (Rousseeuw (1987, 2012)) is an intrinsic metric. The Silhouette index evaluates the clustering by inspecting how well the clusters are isolated from other clusters and examining how the objects are closely coupled within the cluster. The Silhouette Coefficient is formulated by using inter-cluster distance and intra-cluster distance. The Silhouette Coefficient is calculated for each document. The Silhouette Coefficient for a document d_i is formulated as

$$Silhouette\_Score(d_i) = \frac{\Delta_i - \chi_i}{max(\Delta_i, \chi_i)} \tag{5}$$

χ_i defines the intra-cluster distance. It is the mean distance between the document d_i and all other documents in the cluster to which d_i belongs.

Table 5
One-point crossover operation (Cutting point: 3) – Child Solutions.

Child Solution 1	1	3	3	1	3	3	2	1	3	3
Child Solution 2	2	1	2	1	2	3	2	2	1	1

Table 6
Two-point Swap mutation operation (Swap points: 3 & 9).

Before Mutation	1	3	3	1	2	3	2	2	1	1
After Mutation	1	3	1	1	2	3	2	2	3	1

Δ_i defines the inter-cluster distance. It is the mean distance between the document d_i and all other documents in the cluster to which d_i does not belong. The distance between two documents d_i and d_j is computed from Cosine similarity value.

$$distance(d_i, d_j) = 1 - CosineSim(d_i, d_j) \tag{6}$$

The Silhouette Coefficient value is bounded between −1 and +1. The highest quality clustering process must have the Silhouette Coefficient value equal to +1. The fitness score of solution Ψ is defined as follows,

$$fitness(\Psi) = \frac{1}{n}\sum_{i=1}^{n} Silhouette\_Score(d_i) \tag{7}$$

4.5. Generating new solutions

The best and worst solutions in the generation 't' are used to generate the new solution. The best solution has the highest fitness score (+1) and worst solution has the lowest fitness score (-1) in the generation 't'. The best solution of iteration't' is represented as $\Psi_{best}^{(t)}$ and the worst solution of iteration 't' is represented as $\Psi_{worst}^{(t)}$. By considering the best and worst solutions of generation 't', the j^{th} variable of old solution $\Psi_{i,j}^{(t)}$ is modified at generation (t + 1) into $\Psi_{i,j}^{(t+1)}$ as described by Eq. (8):

$$\Psi_{i,j}^{(t+1)} = \Psi_{i,j}^{(t)} + abs\left(\alpha\left|\Psi_{best,j}^{(t)} - \Psi_{i,j}^{(t)}\right| - \beta\left|\Psi_{worst,j}^{(t)} - \Psi_{i,j}^{(t)}\right|\right) \tag{8}$$

Where α, β are two random positive integer numbers between 1 and K (K is the number of clusters). After that, each design variable of the candidate solution is validated as follows,

$$\Psi_{i,j}^{(t+1)} = \begin{cases} \mathrm{mod}\left(\Psi_{i,j}^{(t+1)}, K\right) + 1, & if\ \Psi_{i,j}^{(t+1)} > K \\ \Psi_{i,j}^{(t+1)}, & otherwise \end{cases} \tag{9}$$

4.6. Genetic operations on populations

Crossover: In evolutionary algorithms, the crossover is a genetic operator used to generate new solutions by combining the two parent solutions. In this work, we used the one-point crossover techniques to generate the new child solutions. *One-point crossover:* Each parent solution is divided into two parts based on cutting point. Cutting point is a random positive integer between 1 and K. After that, the second part of the solution is swapped between the two parents. For example, two parent solutions are shown in Table 4. After the crossover, the new children are shown in Table 5.

Mutation: In evolutionary algorithms, the mutation is a genetic operator which is used to maintain the genetic diversity from one generation to the next generation. In this work, we used the swap mutation technique to mutate the solution. Randomly, choose the two positions in a solution and swap the cluster index value of these positions. For example, Table 6 describes the process of two point swap mutation. The Jaya algorithm does not require any algorithm specific control parameter. In this work, we hybridize the Jaya algorithm by adding the genetic operations such as crossover and mutation. In order to maintain the basic principles of Jaya algorithm, it is forced to avoid algorithm specific control parameter (crossover probability, mutation probability, selection strategy) from the proposed hybrid Jaya optimization algorithm. Also, the Jaya algorithm search process only depends on the best and worst candidate solution. Hence, in our proposed work, we perform the genetic operation only on the best and worst candidate solution. The following scenarios are possible while applying the one-point crossover between the best and worst candidate solutions. Let the Parent 1 be the Ψ_{best} candidate solution and Parent 2 be the Ψ_{worst} candidate solution. Based on the one-point crossover, the best candidate solution is divided into two parts as.

Ψ_{best}= Part $1^{\Psi_{best}}$ best genens Part $2^{\Psi_{best}}$ best genes

Case 1: Based on the one-point crossover, the worst candidate solution is divided into two parts as.

Ψ_{worst}= Part $1^{\Psi_{worst}}$ best genens Part $2^{\Psi_{worst}}$ worst genes.

After crossover the new child solutions are as follows,

C1= Part $1^{\Psi_{best}}$ best genens Part $2^{\Psi_{worst}}$ worst genes

C2= Part $1^{\Psi_{worst}}$ best genens Part $2^{\Psi_{best}}$ best genes

According to case 1, child-2 (C2) is more possibility to be a good solution.

Case 2: Based on the one-point crossover, the worst candidate solution is divided into two parts as.

Ψ_{worst}= Part $1^{\Psi_{worst}}$ worst genens Part $2^{\Psi_{worst}}$ best genes.

After crossover the new child solutions are as follows,

C1= Part $1^{\Psi_{best}}$ best genens Part $2^{\Psi_{worst}}$ best genes

C2= Part $1^{\Psi_{worst}}$ worst genens Part $2^{\Psi_{best}}$ best genes

According to case 2, child-1 (C1) is more possibility to be a good solution.

Case 3: Based on the one-point crossover, the worst candidate solution is divided into two parts as.

Ψ_{worst}= Part $1^{\Psi_{worst}}$ worst genens Part $2^{\Psi_{worst}}$ worst genes.

After crossover the new child solutions are as follows,

C1= Part $1^{\Psi_{best}}$ best genens Part $2^{\Psi_{worst}}$ worst genes

C2= Part $1^{\Psi_{worst}}$ worst genens Part $2^{\Psi_{best}}$ best genes

According to case 3, child-1 and child-2 could not be a good solution and it should be ignored. Similarly, the mutation strategy is applied to the worst candidate solution to find the possibility of best solution.

4.7. Pseudo-code of proposed method

Algorithm 1 describes the iterative process of the proposed method. Line 1: pre-process the text corpus dataset. Line 2: tokenize the text corpus to find the bag of words. Line 3: compute the TF-IDF based Document Term Matrix. Line 5: compute the distance matrix. Lines 6–11: generate the initial population table which contains 'N' solutions. Lines 12–36 show the searchprocess of hybrid Jaya optimization algorithm. Line 37 returns the best optimized solution.

Algorithm 1. Document Clustering Using HJO-DC

Input:
D : Dataset with ground truth class label;
K : **Number of clusters** to be formed;
N : Population Size;
T_{max} : Maximum number of **iterations/generations**
Output: Ψ_{best}: best optimal 'K' cluster

D ← textPreProcessing(D)
$T : [t_1, t_2, t_3, ..., t_m] \leftarrow Tokenizer(D)$ // bag of words in D
$DTM \leftarrow TFIDF\_Tokenizer(D)$ // DTM is a |D|×|T| matrix
$n = |D|$ // number of documents in D
$\Gamma \leftarrow computeDistanceBetweenDoc(DTM, n)$ // Γ is a |D|×|D| matrix
// Inital Population construction
foreach *solution* Ψ_i *i=1 to N* **do**
  foreach ***design variable*** *j of solution* Ψ_i*; j=1 to n* **do**
    $\Psi_{i,j} = randomInteger(1, K)$ // random value between 1 and K
  end
  $\Psi_i^{fitness} = computeFitness\left(\Psi_i, \Gamma\right)$
end
$\Psi_{best} = findBestSolution(); \Psi_{worst} = findWorstSolution()$
// Iterative Process
foreach *iteration t 1 to* T_{max} **do**
  foreach *solution* Ψ_i *i=1 to N* **do**
    foreach ***design variable*** *j of solution* Ψ_i*; j=1 to* n_f **do**
      $\alpha = randomInteger(1, K); \beta = randomInteger(1, K)$
      $\Phi_{i,j} = \Psi_{i,j} + abs\left(\alpha \left|\Psi_{best,j} - \Psi_{i,j}\right| - \beta \left|\Psi_{worst,j} - \Psi_{i,j}\right|\right)$ // Φ_i - temporary solution
      $\Phi_{i,j} = validate(\Phi_{i,j})$ // Validate the design variable using Eq.(9)
    end
    $\Phi_i^{fitness} = computeFitness\left(\Phi_i, \Gamma\right)$
    if $\Phi_i^{fitness} > \Psi_i^{fitness}$ **then**
      $\Psi_i = \Phi_i$ // replace the old solution by new (temporary) solution
    end
  end
  $\Psi_{best} \leftarrow findBestSolution(); \Psi_{worst} \leftarrow findWorstSolution()$
  $\Psi_{child1}, \Psi_{child2} \leftarrow Crossover(\Psi_{best}, \Psi_{worst})$
  if $\Psi_{child1}^{fitness} > \Psi_{best}^{fitness}$ **then**
    $\Psi_{best} = \Psi_{child1}$ // replace the best by child1 solution
  end
  if $\Psi_{child2}^{fitness} > \Psi_{best}^{fitness}$ **then**
    $\Psi_{best} = \Psi_{child2}$ // replace the best by child2 solution
  end
  $\Psi_{mutated} \leftarrow MutateSolution(\Psi_{worst})$
  if $\Psi_{mutated}^{fitness} > \Psi_{best}^{fitness}$ **then**
    $\Psi_{best} = \Psi_{mutated}$ // replace the best by mutated solution
  end
end
return Ψ_{best}

Algorithm 2 describes the distance matrix computation process. The size of distance matrix is —D— × —D—. It uses the cosine similarity metrics to find the distance between two documents.

Algorithm 2. Compute distance between documents

Input:
$DTM - DocumentTermMatrix$
$n -$ *number of documents in D*
Output: $\Gamma - DistanceMatrix$
Function $computeDistanceBetweenDoc(DTM, n)$
  for *i=1 to n* **do**
    $d_i \leftarrow DTM_i$ // copy the i^{th} document vector from DTM
    for *j=1 to n* **do**
      $d_j \leftarrow DTM_j$ // copy the j^{th} document vector from DTM
      $\Gamma_{i,j} \leftarrow 1 - cosineSim(d_i, d_j)$ // distance=1-similarity
    end
  end
  return Γ
End Function

Algorithm 3 computes the fitness score of each individual solution. Line 1: indicates the input for the *computeFitness* function. Lines 2–4: form the cluster from the input solution Ψ_{input}. Lines 5–9: compute the silhouette score of each document. Line 10: compute the fitness_score of given input solution Ψ_{input}.

Algorithm 3. Compute Fitness score of Solution

Input:
$\Psi_{input} - Solution$
$\Gamma - Distance\ Matrix$
$n -$ *number of documents in D*
$K -$ *number of clusters to be formed*
Output: $fitness\_score$
Function $computeFitness(\Psi_{input}, \Gamma, n, K)$
  for *k=1 to K* **do**
    $C_k \leftarrow documentIndexOf\ (\Psi_{input}, k)$ // form the cluster from input solution
  end
  // compute Silhouette score for each document in D
  for *i=1 to n* **do**
    $\Delta_i \leftarrow interClusterDistance\ (\Gamma, clusterIndex(d_i), C_k)$
    $\chi_i \leftarrow intraClusterDistance\ (\Gamma, clusterIndex(d_i), C_k)$
    $Silhouette\_Score(d_i) = \frac{\Delta_i - \chi_i}{max(\Delta_i, \chi_i)}$
  end
  // compute Silhouette Index of entire clustered document based on Ψ_{input} solution
  $fitness\_score \leftarrow \frac{1}{n} \sum_{i=1}^{n} Silhouette\_Score(d_i)$
  return $fitness\_score$
End Function

Algorithm 4 performs the crossover genetic operation on two parent solutions. It uses the one-point crossover technique. Line 2: randomly choose the cutting point. Lines 3–4: generate the new child solutions from the input solution.

Table 7
WebKB dataset.

S.No	Category	Total Docs
1	project	504
2	course	930
3	faculty	1124
4	student	1641
	Total	4199

Table 8
SMS dataset.

S.No	Category	Total Docs
1	Spam	747
2	Ham	4827
	Total	5574

Algorithm 4. Crossover between parents $\Psi_{best}, \Psi_{worst}$

Input:
Ψ_{best} - Best Solution
Ψ_{worst} - Worst Solution
Output: $(\Psi_{child1}, \Psi_{child2})$- child solution
1 **Function** $Crossover(\Psi_{best}, \Psi_{worst})$
2 $cuttingPoint \leftarrow randomInteger\left(1, length(\Psi_{best})\right)$
3 $\Psi_{child1} \leftarrow \Psi_{best}[: cuttingPoint] + \Psi_{worst}[cuttingPoint :]$
4 $\Psi_{child2} \leftarrow \Psi_{worst}[: cuttingPoint] + \Psi_{best}[cuttingPoint :]$
5 **return** $(\Psi_{child1}, \Psi_{child2})$
6 **End Function**

Algorithm 5 applies the mutation genetic operation on trail solution. It uses the swap mutation technique. Lines 2–3: randomly choose the swapping point. Line 5: create the new mutated solution from the input solution.

Algorithm 5. Mutation of Ψ_{worst} solution

Input:
Ψ_{worst} - Worst Solution
Output: $\Psi_{mutated}$- mutated solution
1 **Function** $MutateSolution(\Psi_{worst})$
2 $swapPoint1 \leftarrow randomInteger\left(1, length(\Psi_{worst})\right)$
3 $swapPoint2 \leftarrow randomInteger\left(1, length(\Psi_{worst})\right)$
4 $\Psi_{mutated} \leftarrow \Psi_{worst}$
5 $\Psi_{mutated,swapPoint1}, \Psi_{mutated,swapPoint2} \leftarrow \Psi_{mutated,swapPoint2}, \Psi_{mutated,swapPoint1}$
6 **return** $\Psi_{mutated}$
7 **End Function**

5. Implementation details

This section describes the datasets selected as benchmark examples and the evaluation metric chosen for assessing the performance of the proposed text document clustering algorithm.

5.1. Dataset

In this work, five distinct datasets (WebKB, SMS, BBC News, 10Newsgroups, DMOZ) were used for evaluating the performance of the proposed clustering algorithm. WebKB[1] is a collection of web pages collected by the World Wide Knowledge Base. These pages were collected from computer science departments of various universities in 1997. The web pages are classified into various classes: student, faculty, staff, department, course, project, and other. For the experimental works we have chosen the class label (project, course, faculty, student) documents. Table 7 describes the properties of WebKB dataset. SMS[2] Dataset is a collection of SMS, which is labeled as Spam or Ham. It contains 5574 labeled SMS message. Description of the SMS dataset is shown in Table 8. BBC[3] Dataset contains 2225 text documents from the BBC news website, which are classified into five categories (business, entertainment, politics, sport, tech). Table 9 shows the document distribution of BBC dataset. 20Newsgroups[4] dataset contains approximately 15000 news documents, which are manually classified into 20 groups. In this work we considered ten categories. News document distribution among the selected categories is shown in Table 10. DMOZ[5] directory contains url classification dataset. It contains more than 100000 url description. Here, we considered 4 categories with selected documents. The DMOZ dataset distribution among the selected categories is shown in Table 11.

5.2. Cluster quality evaluation metrics

The extrinsic and intrinsic methods are used to measure the quality of clustering. If the ground truth (cluster label) of the dataset is available, extrinsic methods are used, otherwise intrinsic methods are used. There are four standard extrinsic evaluation measures namely accuracy, precision, recall and F-Score, that are used to evaluate the performance of the proposed clustering methods. The clustering accuracy is defined as the percentage of documents accurately placed into the actual ground truth cluster label.

$$Accuracy = \frac{n_{correct}}{n} * 100 \tag{10}$$

where $n_{correct}$ – number of documents that are correctly placed in the

Table 9
BBC dataset.

S.No	Category	Total Docs
1	Business	510
2	Entertainment	386
3	Politics	417
4	Sport	511
5	Tech	401
	Total	2225

[1] WebKb: http://ana.cachopo.org/datasets-for-single-label-text-categorization.
[2] SMS: https://www.kaggle.com/uciml/sms-spam-collection-dataset.
[3] BBC: https://storage.googleapis.com/dataset-uploader/bbc/bbc-text.csv.
[4] 20Ng: http://ana.cachopo.org/datasets-for-single-label-text-categorization.
[5] DMOZ: https://www.kaggle.com/shawon10/url-classification-dataset-dmoz

Table 10
10Newsgroup dataset.

S.No	Category	Total Docs
1	rec.autos	989
2	rec.motorcycles	996
3	rec.sport.baseball	994
4	rec.sport.hockey	999
5	sci.crypt	991
6	sci.electronics	984
7	sci.med	990
8	sci.space	987
9	soc.religion.christian	996
10	talk.politics.guns	909
	Total	9835

Table 11
DMOZ dataset.

S.No	Category	Total Docs
1	Arts	500
2	Business	500
3	Health	500
4	Science	500
	Total	2000

Table 12
The parameters settings for Genetic algorithm.

GA parameters	Values
Selection strategy	Roulette wheel selection
Crossover operator	one-point crossover
Crossover probability	0.5
Mutation operator	Swap
Mutation probability	0.3

Table 13
The parameters settings for PSO algorithm.

PSO parameters	Values
W – inertia weight	0.75
Wdamp – inertia weight damping ratio	0.7
C1 – personal learning coefficient	1.5
C2 – global learning coefficient	1.5

corresponding cluster as per the actual ground truth of cluster label. The Precision (P) is a measure of exactness. It is defined as the percentage of documents correctly placed in actual ground truth cluster label.

$$Precision = \frac{1}{K}\sum_{k=1}^{K}\frac{nk_{correct}}{|C_k|} \tag{11}$$

where $nk_{correct}$ is number of documents correctly placed in the cluster C_k. $|C_k|$ is the total number of documents placed in cluster C_k.

The Recall (R) is a measure of completeness. It is the ratio between the number of documents correctly placed in cluster C_k and the number of documents actually available in the ground truth cluster label C_k.

$$Recall = \frac{nk_{correct}}{|GTC_k|} \tag{12}$$

Table 14
The parameters settings for CS algorithm.

CS parameters	Values
α – Step size	1
λ – Levy distribution co-efficient	1.5
P_a – Probability of discovery of alien egg	0.25

Table 15
The parameters settings for FFY algorithm.

FFY parameters	Values
α – randomization parameter	0.1
B_0 – Base attraction	1
γ – Absorption coefficient	1

Table 16
The parameters settings for GWO algorithm.

GWO parameter	Values
a – controlling parameter	$a \in [2, 0]$

where $|GTC_k|$ is the number of documents actually available in the ground truth cluster label C_k.

The F-Score (F) is defined as the weighted harmonic mean of the precision and recall.

$$F = \frac{2 * Precision * Recall}{Precision + Recall} \tag{13}$$

6. Results and discussion

In this section, the HJO-DC document clustering algorithm is extensively compared with partitioning techniques such as k-Means, K-medoids and metaheuristic algorithms such as Genetic Algorithm (GA), Cuckoo search (CS), Particle Swam Optimizer (PSO), Grey Wolf Optimizer (GWO) and Firefly algorithm (FFY) in terms of accuracy, precision, recall and F-Score. The algorithm specific parameter values of GA, PSO, CS, FFY and GWO are shown in Tables 12–16.

Computations were carried out on a computer equipped with 2.30 GHz, Intel Core i5 processor and 16 GB RAM memory. We have used Python 3.7.3 for programming and matplotlib library to plot the performance graph. We set the population size N as 50 and maximum iteration T_{max} as 200. In most cases, the proposed method shows better accuracy than its competitors. The proposed HJO-DC based document

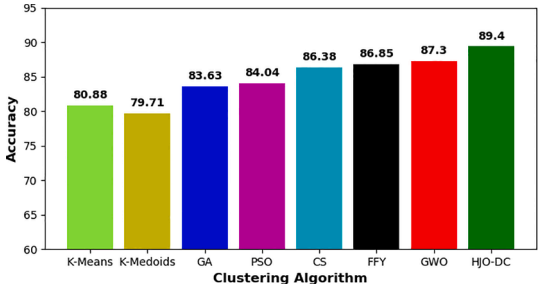

Fig. 3. Accuracy comparison of clustering algorithms for WebKB dataset.

Table 17
Performance comparison of clustering algorithms for WebKB dataset.

Category	K-Means			K-Medoids			GA			PSO			CS			FFY			GWO			HJO-DC		
	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F
project	0.62	0.81	0.7	0.62	0.8	0.7	0.68	0.84	0.75	0.68	0.84	0.75	0.7	0.86	0.77	0.74	0.87	0.8	0.74	0.87	0.8	0.77	0.89	0.83
course	0.78	0.81	0.8	0.75	0.8	0.77	0.81	0.84	0.82	0.81	0.84	0.83	0.84	0.86	0.85	0.85	0.87	0.86	0.86	0.87	0.86	0.87	0.89	0.88
faculty	0.82	0.81	0.82	0.81	0.8	0.81	0.85	0.84	0.84	0.86	0.84	0.85	0.88	0.86	0.87	0.89	0.87	0.88	0.88	0.87	0.88	0.91	0.89	0.90
student	0.9	0.81	0.85	0.89	0.8	0.84	0.9	0.84	0.87	0.91	0.84	0.87	0.94	0.86	0.9	0.92	0.87	0.89	0.93	0.87	0.9	0.94	0.89	0.92
macro avg	0.78	0.81	0.79	0.77	0.8	0.78	0.81	0.84	0.82	0.82	0.84	0.83	0.84	0.86	0.85	0.85	0.87	0.86	0.85	0.87	0.86	0.87	0.89	0.88
weighted avg	0.82	0.81	0.81	0.81	0.8	0.8	0.84	0.84	0.84	0.85	0.84	0.84	0.87	0.86	0.87	0.87	0.87	0.87	0.88	0.87	0.87	0.9	0.89	0.89
Total No. of Docs		4199			4199			4199			4199			4199			4199			4199			4199	
Correctly clustered		3396(80.88%)			3347(79.71%)			3512(83.64%)			3529(84.04%)			3627(86.38%)			3647(86.85%)			3666(87.3%)			3754(89.4%)	
Incorrectly clustered		803(19.12%)			852(20.29%)			687(16.36%)			670(15.96%)			572(13.62%)			552(13.15%)			533(12.7%)			445(10.6%)	

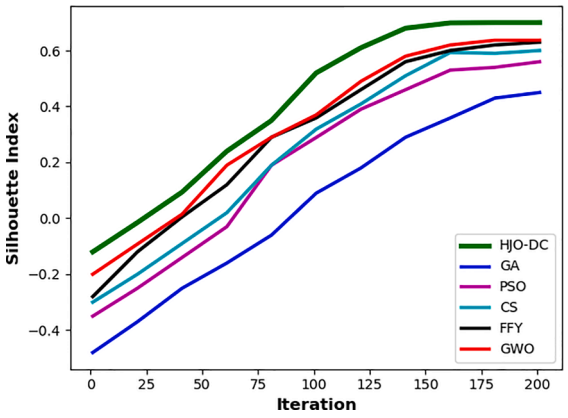

Fig. 4. Comparison of convergence behavior of clustering algorithms for WebKB dataset.

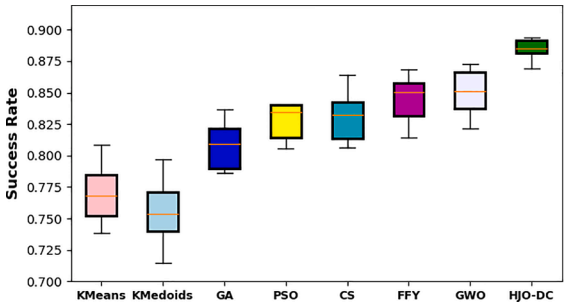

Fig. 5. The boxplots of the clustering accuracy results for WebKB dataset.

clustering algorithm is executed 20 times on the datasets in Tables 7–11 and the clustering accuracy of best, worst, average and standard deviation are compared to each other algorithm.

6.1. Performance comparisons on the WebKB Dataset

The WebKB dataset documents are grouped into 4 clusters as per the ground truth class label of dataset. The data plotted in Fig. 3 show that the proposed HJO-DC algorithm achieved a higher clustering accuracy on the WebKB dataset than the other comparative methods. For the WebKB dataset, the proposed HJO-DC algorithm achieved 89.4% of accuracy. Table 17 compares the clustering results in terms of Precision, Recall and F-Score for the WebKb dataset. It can be seen that the HJO-DC method has a larger number of instances correctly clustered (3754 instances over 4199) than the other existing techniques. The Precision, Recall and F-Score of proposed HJO-DC are 0.9,0.89 and 0.89,

Table 18
Results of the repeatability test for WebKB dataset.

Algorithms	Clustering accuracy			
	Worst	Best	Average	STDEV
K-Means	0.7383	0.8088	0.768	0.0218
K-Medoids	0.7147	0.7971	0.7569	0.0242
GA	0.7859	0.8364	0.8085	0.0185
PSO	0.8054	0.8404	0.8282	0.0141
CS	0.8061	0.8638	0.8318	0.0192
FFY	0.814	0.8685	0.8436	0.0179
GWO	0.8216	0.8731	0.8506	0.0166
HJO-DC	0.8693	0.894	0.884	0.0081

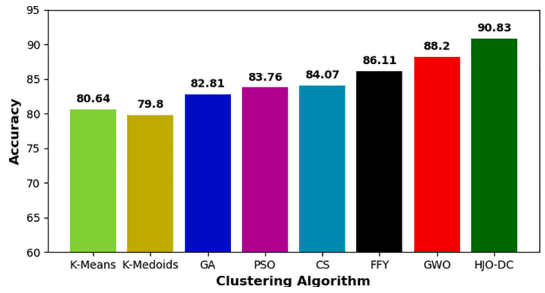

Fig. 6. Accuracy comparison of clustering algorithms for SMS dataset.

respectively. According to the F-measure value, the proposed HJO-DC algorithm outperforms all other methods. Fig. 4 compares the convergence curves of the different algorithms. Remarkably, HJO-DC generated intermediate solutions with higher Silhouette index values than its competitors over the whole optimization process of the WebKB dataset example. The boxplots in the Fig. 5 shows the distribution of the clustering accuracy gained from 20 runs of proposed algorithm and other algorithms for WebKB dataset examples. It can be seen that there is a less gap between the best, median, and the worst results achieved by the proposed HJO-DC clustering algorithm. It indicates that the performanace of proposed algorithm is robust. The results of the proposed algorithm and other algorithms in terms of the best, worst, average and the standard deviation of the clustering accuracy are presented in Table 18. As show in Table 18, it can be concluded that the proposed algorithm's performance is better than other algorithms.

6.2. Performance comparisons on the SMS dataset

The SMS dataset documents are grouped into 2 clusters as per the ground truth class label of dataset(spam,ham). Fig. 6 shows that the clustering accuracy on SMS dataset for the proposed HJO-DC is better than for other existing methods. The proposed HJO-DC algorithm achieved 90.83% of clustering accuracy for the SMS dataset. Table 19 compares the clustering results in terms of Precision, Recall and F-Score on the SMS dataset. It can be seen that the HJO-DC method has a larger number of instances correctly clustered (5063 instances over 5574) than the other existing techniques. The Precision, Recall and F-Score of proposed HJO-DC are 0.93, 0.91 and 0.92, respectively. According to the F-measure value, the proposed HJO-DC outperforms all other methods. The comparison of convergence curves plotted in Fig. 7 confirms the superiority of HJO-DC, which obtained higher values of the Silhouette index throughout optimization process also for the SMS dataset example. In order to check the reliability, each algorithm is executed 20 times on the SMS dataset. Fig. 8 shows the distribution of the clustering accuracy. It is quite evident that the proposed HJO-DC based document clustering algorithm is more reliable than the other methods. Table 20 summarizes the best, worst, average and the standard deviation of the results in terms of clustering accuracy by the proposed algorithm and other algorithms. In terms of STDEV, the difference between HJO-DC and the other algorithm was significant. The proposed algorithm achieved highest average of clustering accuracy than other algorithms.

6.3. Performance comparisons on the BBC dataset

The BBC News dataset documents are grouped into 5 clusters as per the ground truth class label of dataset. Fig. 9 shows that the clustering accuracy on BBC dataset the proposed HJO-DC algorithm is higher than for other methods. For the BBC dataset, the proposed HJO-DC algorithm achieved 89.62% of clustering accuracy. Table 21 compares the clustering results in terms of Precision, Recall and F-Score on the Webkb dataset. It can be seen that the HJO-DC method has a larger number of instances correctly clustered (1994 instances over 2225) than the other

Table 19
Performance comparison of clustering algorithms for SMS dataset.

Category	K-Means			K-Medoids			GA			PSO			CS			FFY			GWO			HJO-DC		
	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F
Spam	0.39	0.81	0.53	0.37	0.79	0.51	0.43	0.83	0.56	0.44	0.84	0.58	0.45	0.84	0.59	0.49	0.86	0.62	0.54	0.88	0.67	0.61	0.91	0.73
Ham	0.96	0.81	0.88	0.96	0.79	0.87	0.97	0.83	0.89	0.97	0.84	0.9	0.97	0.84	0.9	0.98	0.86	0.91	0.98	0.88	0.93	0.98	0.91	0.94
macro avg	0.68	0.81	0.7	0.67	0.79	0.69	0.7	0.83	0.73	0.71	0.84	0.74	0.71	0.84	0.74	0.73	0.86	0.77	0.76	0.88	0.8	0.79	0.91	0.84
weighted avg	0.89	0.81	0.83	0.88	0.79	0.82	0.9	0.83	0.85	0.9	0.84	0.86	0.9	0.84	0.86	0.91	0.86	0.88	0.92	0.88	0.89	0.93	0.91	0.92
Total No. of Docs	5574			5574			5574			5574			5574			5574			5574			5574		
Correctly clustered	4495(80.64%)			4421(79.31%)			4616(82.81%)			4669(83.76%)			4686(84.1%)			4800(86.11%)			4916(88.2%)			5063(90.83%)		
Incorrectly clustered	1079(19.36%)			1153(20.69%)			958(17.19%)			905(16.24%)			888(15.9%)			774(13.89%)			658(11.8%)			511(9.17%)		

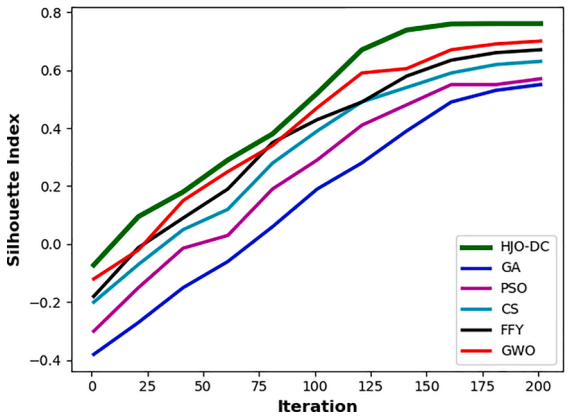

Fig. 7. Comparison of convergence behavior of clustering algorithms for SMS dataset.

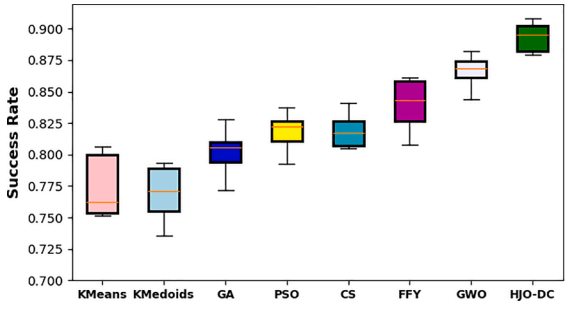

Fig. 8. The boxplots of the clustering accuracy results for SMS dataset.

Table 20
Results of the repeatability test for SMS dataset.

Algorithms	Clustering accuracy			
	Worst	Best	Average	STDEV
K-Means	0.7517	0.8064	0.774	0.0229
K-Medoids	0.7356	0.7931	0.77	0.0199
GA	0.7714	0.8281	0.8033	0.0161
PSO	0.7926	0.8376	0.8194	0.0137
CS	0.8052	0.8407	0.8193	0.0134
FFY	0.8075	0.8611	0.8396	0.0197
GWO	0.8441	0.882	0.866	0.011
HJO-DC	0.8791	0.9083	0.8931	0.0106

existing techniques. The Precision, Recall and F-Score of proposed HJO-DC are 0.9, 0.9 and 0.9, respectively. According to the F-measure value, the proposed HJO-DC outperforms allother methods. The comparison of convergence curves plotted in Fig. 10 confirms the superiority of HJO-DC, which obtained higher values of the Silhouette index throughout optimization process also for the BBC dataset example. The boxplots in Fig. 11 shows the distribution of the clustering accuracy which are taken from 20 runs of proposed algorithm and other algorithm. Eventually, it can be seen that the performance of the proposed algorithm outperforms all other methods. Finally, in Table 22, the best, worst, average and the

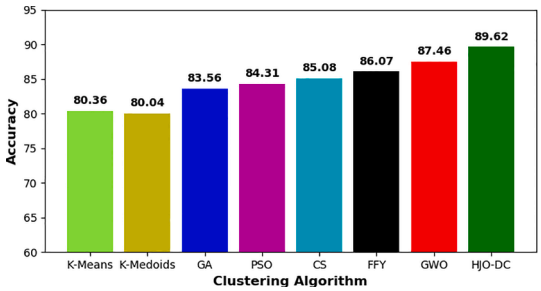

Fig. 9. Accuracy comparison of clustering algorithms for BBC dataset.

standard deviation of the results obtained by HJO-DC algorithm and other algorithms in terms of clustering accuracy are presented. It can be noted that, the proposed HJO-DC algorithm achieved the highest average of clustering accuracy (0.881) while comparing other algorithms.

6.4. Performance comparisons on the 10Newsgroup dataset

The 10Newsgroup dataset documents are grouped into 10 clusters as per the ground truth class label of dataset. Fig. 12 shows that the clustering accuracy on 10Newgroup dataset for the proposed HJO-DC is higher than for the other methods. For the 10Newsgroup dataset, the proposed HJO-DC algorithm achieved 86.27% of clustering accuracy. Table 23 compares the clustering results in terms of Precision, Recall and F-Score on the 10Newsgroup dataset. It can be seen that the HJO-DC method has a larger number of instances correctly clustered (8485 instances over 9835) than the other existing techniques. The Precision, Recall and F-Score of the proposed HJO-DC algorithm are 0.86, 0.86 and 0.86, respectively. According to the F-measure value, the proposed HJO-DC outperforms all other methods. The comparison of convergence curves plotted in Fig. 13 confirms the superiority of HJO-DC, which obtained higher values of the Silhouette index throughout optimization process also for the 10Newsgroup dataset example. The boxplots in the Fig. 14 shows the distribution of the clustering accuracy gained from 20 runs of proposed algorithm and other algorithms for 10Newsgroup dataset examples. It can be seen that there is a less gap between the best, median, and the worst results achieved by the proposed HJO-DC clustering algorithm. It indicates that the performanace of proposed algorithm is robust. Table 24 outlines the best, worst, average and standard deviation of the clustering accuracy obtained by the proposed algorithm and other algorithms. As shown in Table 24, the proposed algorithm achieved a better performamce while comparing other algorithms.

6.5. Performance comparisons on the DMOZ dataset

The DMOZ dataset documents are grouped into 4 clusters as per the ground truth class label of dataset. Fig. 15 shows that the clustering accuracy on DMOZ dataset for the proposed HJO-DC is higher than for other. For the DMOZ dataset, the proposed HJO-DC algorithm achieved 87.4% of clustering accuracy. Table 25 compares the clustering results in terms of Precision, Recall and F-Score on the DMOZ dataset. It can be seen that theHJO-DC method has a larger number of instances correctly clustered (1748 instances over 2000) than the other existing techniques. The Precision, Recall and F-Score of proposed HJO-DC algorithm are 0.87,0.87 and 0.87, respectively. According to the F-measure value, the proposed HJO-DC outperforms all other methods. The comparison of convergence curves plotted in Fig. 16 confirms the superiority of HJO-

Table 21
Performance comparison of clustering algorithms for BBC dataset.

Category	K-Means			K-Medoids			GA			PSO			CS			FFY			GWO			HJO-DC		
	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F
Business	0.83	0.8	0.82	0.82	0.8	0.81	0.88	0.84	0.86	0.87	0.84	0.85	0.88	0.85	0.86	0.88	0.86	0.87	0.91	0.87	0.89	0.91	0.9	0.90
Entertainment	0.78	0.8	0.79	0.76	0.8	0.78	0.82	0.84	0.83	0.81	0.84	0.82	0.84	0.85	0.85	0.85	0.86	0.85	0.84	0.87	0.85	0.88	0.9	0.89
Politics	0.78	0.8	0.79	0.76	0.8	0.78	0.83	0.84	0.83	0.8	0.84	0.82	0.84	0.85	0.84	0.85	0.86	0.85	0.86	0.88	0.87	0.9	0.9	0.90
Sport	0.85	0.8	0.83	0.85	0.8	0.82	0.84	0.84	0.84	0.9	0.84	0.87	0.87	0.85	0.86	0.87	0.86	0.87	0.9	0.87	0.89	0.93	0.9	0.91
Tech	0.77	0.8	0.79	0.8	0.8	0.8	0.8	0.84	0.82	0.83	0.84	0.84	0.82	0.85	0.83	0.85	0.86	0.85	0.86	0.88	0.87	0.86	0.9	0.88
macro avg	0.8	0.8	0.8	0.8	0.8	0.8	0.83	0.84	0.83	0.84	0.84	0.84	0.85	0.85	0.85	0.86	0.86	0.86	0.87	0.87	0.87	0.89	0.9	0.90
weighted avg	0.8	0.8	0.8	0.8	0.8	0.8	0.84	0.84	0.84	0.84	0.84	0.84	0.85	0.85	0.85	0.86	0.86	0.86	0.88	0.87	0.87	0.9	0.9	0.90
Total No. of Docs	2225			2225			2225			2225			2225			2225			2225			2225		
Correctly clustered	1788(80.36%)			1781(80.04%)			1860(83.6%)			1876(84.31%)			1893(85.08%)			1915(86.07%)			1946(87.46%)			1994(89.62%)		
Incorrectly clustered	437(19.64%)			444(19.96%)			365(16.4%)			349(15.69%)			332(14.92%)			310(13.93%)			279(12.54%)			231(10.38%)		

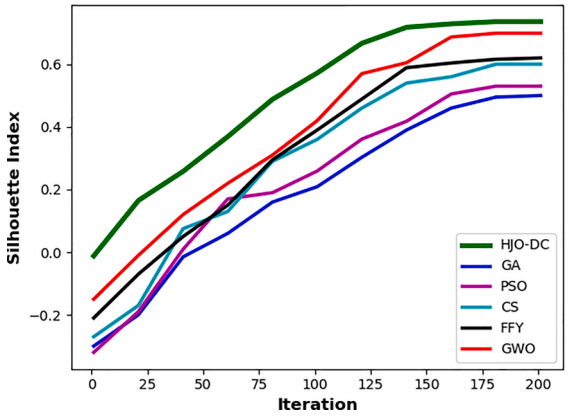

Fig. 10. Comparison of convergence behavior of clustering algorithms for BBC dataset.

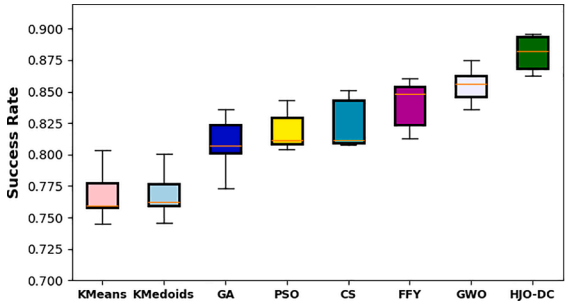

Fig. 11. The boxplots of the clustering accuracy results for BBC dataset.

DC, which obtained higher values of the Silhouette index throughout optimization process also for the DMOZ dataset example. The boxplots in Fig. 17 shows the distribution of the clustering accuracy which are taken from 20 runs of proposed algorithm and other algorithm. Eventually, it can be seen that the performance of the proposed algorithm outperforms all other methods. Finally, in Table 26, the best, worst, average and the standard deviation of the results obtained by HJO-DC algorithm and other algorithms in terms of clustering accuracy are presented. It can be noted that, the proposed HJO-DC algorithm achieved the highest average of clustering accuracy(0.8631) while comparing other algorithms.

Table 22
Results of the repeatability test for BBC dataset.

Algorithms	Clustering accuracy			
	Worst	Best	Average	STDEV
K-Means	0.7452	0.8036	0.7673	0.0189
K-Medoids	0.7461	0.8004	0.7683	0.0173
GA	0.773	0.836	0.8079	0.0193
PSO	0.8045	0.8431	0.818	0.0135
CS	0.8081	0.8508	0.8237	0.0179
FFY	0.813	0.8607	0.8409	0.0181
GWO	0.836	0.8746	0.8554	0.0114
HJO-DC	0.8629	0.8962	0.881	0.013

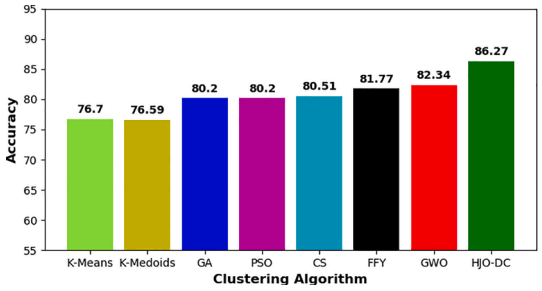

Fig. 12. Accuracy comparison of clustering algorithms for 10Newsgroup dataset.

7. Conclusion

In this paper, a hybrid Jaya optimization algorithm is proposed to cluster the text document. The proposed algorithm enhanced the global search procedure by including in the Jaya optimization with crossover and mutation. The performance of proposed HJO-DC was investigated against various well-known document clustering techniques. Computations were carried out on five well known benchmark datasets. The detailed analysis of numerical results indicated with no shadow of doubt that the hybrid jaya optimization based document clustering scheme is better than other clustering techniques.

Table 23
Performance comparison of clustering algorithms for 10Newsgroup dataset.

Category	K-Means			K-Medoids			GA			PSO			CS			FFY			GWO			HJO-DC		
	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F
rec.autos	0.76	0.77	0.76	0.78	0.77	0.77	0.8	0.8	0.8	0.78	0.8	0.79	0.81	0.8	0.81	0.83	0.82	0.82	0.82	0.82	0.82	0.86	0.86	0.86
rec.motorcycles	0.78	0.77	0.78	0.76	0.77	0.77	0.81	0.8	0.81	0.8	0.8	0.8	0.81	0.81	0.81	0.81	0.82	0.81	0.82	0.82	0.82	0.86	0.86	0.86
rec.sport.baseball	0.78	0.77	0.77	0.77	0.77	0.77	0.79	0.8	0.8	0.81	0.8	0.81	0.81	0.8	0.81	0.82	0.82	0.82	0.83	0.82	0.83	0.86	0.86	0.86
rec.sport.hockey	0.75	0.77	0.76	0.75	0.77	0.76	0.81	0.8	0.81	0.79	0.8	0.79	0.8	0.8	0.8	0.82	0.82	0.82	0.82	0.82	0.82	0.86	0.86	0.86
sci.crypt	0.78	0.77	0.77	0.76	0.77	0.76	0.8	0.8	0.8	0.8	0.8	0.8	0.79	0.81	0.8	0.82	0.82	0.82	0.83	0.82	0.83	0.85	0.86	0.86
sci.electronics	0.78	0.77	0.77	0.76	0.77	0.76	0.82	0.8	0.81	0.81	0.8	0.81	0.83	0.8	0.82	0.81	0.82	0.81	0.82	0.82	0.82	0.86	0.86	0.86
sci.med	0.77	0.77	0.77	0.78	0.77	0.77	0.83	0.8	0.81	0.8	0.8	0.8	0.8	0.81	0.8	0.81	0.82	0.81	0.81	0.82	0.82	0.86	0.86	0.86
sci.space	0.75	0.77	0.76	0.77	0.77	0.77	0.79	0.8	0.8	0.81	0.8	0.81	0.8	0.81	0.8	0.84	0.82	0.83	0.85	0.82	0.84	0.87	0.86	0.86
soc.religion.christian	0.77	0.77	0.77	0.75	0.77	0.76	0.81	0.8	0.8	0.83	0.8	0.81	0.82	0.81	0.81	0.82	0.82	0.82	0.83	0.82	0.83	0.87	0.86	0.87
talk.politics.guns	0.74	0.77	0.75	0.77	0.77	0.77	0.77	0.8	0.79	0.79	0.8	0.8	0.79	0.81	0.8	0.81	0.82	0.82	0.81	0.82	0.82	0.88	0.86	0.87
macro avg	0.77	0.77	0.77	0.77	0.77	0.77	0.8	0.8	0.8	0.8	0.8	0.8	0.81	0.81	0.81	0.82	0.82	0.82	0.82	0.82	0.82	0.86	0.86	0.86
weighted avg	0.77	0.77	0.77	0.77	0.77	0.77	0.8	0.8	0.8	0.8	0.8	0.8	0.81	0.81	0.81	0.82	0.82	0.82	0.82	0.82	0.82	0.86	0.86	0.86
Total No. of Docs		9835			9835			9835			9835			9835			9835			9835			9835	
Correctly clustered		7543(76.7%)			7533(76.59%)			7888(80.2%)			7888(80.2%)			7918(80.51%)			8042(81.77%)			8098(82.33%)			8485(86.27%)	
Incorrectly clustered		2292(23.3%)			2302(23.41%)			1947(19.8%)			1947(19.8%)			1917(19.49%)			1793(18.23%)			1737(17.66%)			1350(13.73%)	

Fig. 13. Comparison of convergence behavior of clustering algorithms for 10Newsgroup dataset.

Fig. 14. The boxplots of the clustering accuracy results for 10Newsgroup dataset.

Table 24
Results of the repeatability test for 10Newsgroup dataset.

Algorithms	Clustering accuracy			
	Worst	Best	Average	STDEV
K-Means	0.7117	0.767	0.7409	0.0188
K-Medoids	0.7128	0.7659	0.7413	0.0182
GA	0.767	0.802	0.7835	0.0136
PSO	0.7626	0.802	0.7829	0.0143
CS	0.7717	0.8051	0.7877	0.0111
FFY	0.7728	0.8177	0.7931	0.0139
GWO	0.785	0.8234	0.8075	0.0121
HJO-DC	0.8238	0.8627	0.8458	0.0119

Fig. 15. Accuracy comparison of clustering algorithms for DMOZ dataset.

Table 25
Performance comparison of clustering algorithms for DMOZ dataset.

Category	K-Means			K-Medoids			GA			PSO			CS			FFY			GWO			HJO-DC		
	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F
Arts	0.8	0.81	0.81	0.82	0.81	0.81	0.83	0.83	0.83	0.84	0.83	0.83	0.82	0.84	0.83	0.83	0.85	0.84	0.86	0.85	0.85	0.88	0.87	0.88
Business	0.81	0.81	0.81	0.79	0.81	0.8	0.85	0.83	0.84	0.83	0.83	0.83	0.86	0.84	0.85	0.85	0.85	0.85	0.87	0.85	0.86	0.88	0.87	0.88
Health	0.81	0.81	0.81	0.79	0.81	0.8	0.83	0.83	0.83	0.85	0.83	0.84	0.85	0.84	0.85	0.86	0.85	0.85	0.84	0.85	0.85	0.88	0.87	0.87
Science	0.81	0.81	0.81	0.82	0.81	0.81	0.83	0.83	0.83	0.81	0.83	0.82	0.84	0.84	0.84	0.86	0.85	0.85	0.84	0.85	0.85	0.86	0.87	0.87
macro avg	0.81	0.81	0.81	0.81	0.81	0.81	0.83	0.83	0.83	0.83	0.83	0.83	0.84	0.84	0.84	0.85	0.85	0.85	0.85	0.85	0.85	0.87	0.87	0.87
weighted avg	0.81	0.81	0.81	0.81	0.81	0.81	0.83	0.83	0.83	0.83	0.83	0.83	0.84	0.84	0.84	0.85	0.85	0.85	0.85	0.85	0.85	0.87	0.87	0.87
Total No. of Docs	2000			2000			2000			2000			2000			2000			2000			2000		
Correctly clustered	1616(80.8%)			1612(80.6%)			1668(83.4%)			1660(83%)			1684(34.2%)			1696(84.8%)			1704(85.2%)			1748(87.4%)		
Incorrectly clustered	384(19.2%)			388(19.4%)			332(16.6%)			340(17%)			316(15.8%)			304(15.2%)			296(14.8%)			252(12.6%)		

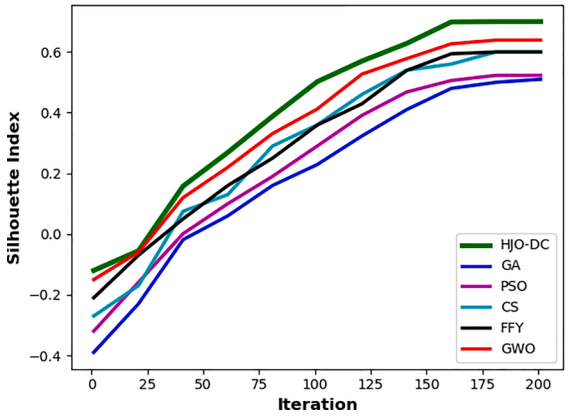

Fig. 16. Comparison of convergence behavior of clustering algorithms for DMOZ dataset.

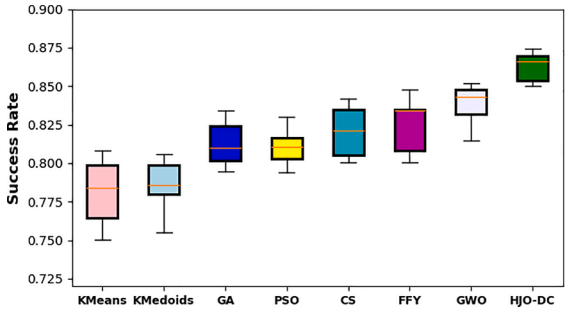

Fig. 17. The boxplots of the clustering accuracy results for DMOZ dataset.

Table 26
Results of the repeatability test for DMOZ dataset.

Algorithms	Clustering accuracy			
	Worst	Best	Average	STDEV
K-Means	0.7505	0.808	0.7815	0.0196
K-Medoids	0.755	0.806	0.7858	0.0148
GA	0.7945	0.834	0.8132	0.0131
PSO	0.794	0.83	0.8097	0.011
CS	0.8005	0.842	0.8207	0.0169
FFY	0.8005	0.848	0.8249	0.0167
GWO	0.815	0.852	0.839	0.0118
HJO-DC	0.85	0.874	0.8631	0.0091

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

We would like to thank the anonymous reviewers for their helpful comments and advice in improving this work. Also, we would like to thank the Management and Principal of Mepc Schlenk Engineering College (Autonomous), Sivakasi for providing us the state of art facilities to carry out this proposed research work in the Mepco Research Centre in collaboration with Anna University Chennai, Tamil Nadu, India.

References

Abualigah, L. M., Khader, A. T., & Hanandeh, E. S. (2018). A new feature selection method to improve the document clustering using particle swarm optimization algorithm. *Journal of Computational Science, 25*, 456–466. https://doi.org/10.1016/j.jocs.2017.07.018

Afonso, A., & Gottschalg-Duque, C. (2014). Automated text clustering of newspaper and scientific texts in brazilian portuguese: Analysis and comparison of methods. *JISTEM, 11*, 415–436. https://doi.org/10.4301/S1807-17752014000200011

Agarwal, A., & Roul, R. K. (2018). A novel hierarchical clustering algorithm for online resources. In P. K. Sa, S. Bakshi, I. K. Hatzilygeroudis, & M. N. Sahoo (Eds.), *Recent findings in intelligent computing techniques* (pp. 467–476). Singapore: Springer Singapore.

Ahmadi, P., Gholampour, I., & Tabandeh, M. (2017). Cluster-based sparse topical coding for topic mining and document clustering. *Advances in Data Analysis and Classification, 12*, 537–558. https://doi.org/10.1007/s11634-017-0280-3. https://doi.org/10.1007.

Akter, R., & Chung, Y. (2013). An evolutionary approach for document clustering. *IERI Procedia, 4*, 370–375. https://doi.org/10.1016/j.ieri.2013.11.053. URL: http://www.sciencedirect.com/science/article/pii/S2212667813000567.

Al-Anazi, S., Almahmoud, H., & Al-Turaiki, I. (2016). Finding similar documents using different clustering techniques. *Procedia Computer Science, 82*, 28–34. https://doi.org/10.1016/j.procs.2016.04.005

Bouras, C., & Tsogkas, V. (2012). A clustering technique for news articles using wordnet. *Knowledge-Based Systems, 36*, 115–128. https://doi.org/10.1016/j.knosys.2012.06.015. URL: http://www.sciencedirect.com/science/article/pii/S0950705112001864.

Boushaki, S., Kamel, N., & Bendjeghaba, O. (2018). A new quantum chaotic cuckoo search algorithm for data clustering. *Expert Systems with Applications, 96*, 358–372. https://doi.org/10.1016/j.eswa.2017.12.001. URL: http://www.sciencedirect.com/science/article/pii/S0957417417308217.

Bouyer, A., Ghafarzadeh, H., & Tarkhaneh, O. (2015). An efficient hybrid algorithm using cuckoo search and differential evolution for data clustering. *Indian Journal of Science and Technology, 8*, 1–12. https://doi.org/10.17485/ijst/2015/v8i24/60146

Bsoul, Q., Salim, J., & Zakaria, L. Q. (2013). An intelligent document clustering approach to detect crime patterns. *Procedia Technology, 11*, 1181–1187. URL: http://www.sciencedirect.com/science/article/pii/S2212017313004659, doi: 10.1016/j.protcy.2013.12.311. 4th International Conference on Electrical Engineering and Informatics, ICEEI 2013.

Buddala, R., & Mahapatra, S. S. (2018). Improved teaching–learning-based and jaya optimization algorithms for solving flexible flow shop scheduling problems. *Journal of Industrial Engineering International, 14*, 555–570.

Das, S. R., Mishra, D., & Rout, M. (2020). A hybridized elm-jaya forecasting model for currency exchange prediction. *Journal of King Saud University – Computer and*

Information Sciences, 32, 345–366. https://doi.org/10.1016/j.jksuci.2017.09.006. URL: http://www.sciencedirect.com/science/article/pii/S131915781730160X.
Dasgupta, S. (2016). *A cost function for similarity-based hierarchical clustering* (pp. 118–127). New York, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/2897518.2897527
Degertekin, S., Lamberti, L., & Ugur, I. (2019). Discrete sizing/layout/topology optimization of truss structures with an advanced jaya algorithm. *Applied Soft Computing, 79*, 363–390. https://doi.org/10.1016/j.asoc.2019.03.058
Du, D. C., Vinh, H. H., Trung, V. D., Quyen, N. T. H., & Trung, N. T. (2018). Efficiency of jaya algorithm for solving the optimization-based structural damage identification problem based on a hybrid objective function. *Engineering Optimization, 50*, 1233–1251. https://doi.org/10.1080/0305215X.2017.1367392. arXiv:https://doi.org/10.1080/0305215X.2017.1367392.
Fahad, S. (2016). A modified k-means algorithm for big data clustering. *International Journal of Computer Science Engineering and Technology, 6*, 129–132.
Han, J., Kamber, M., & Pei, J. (2012). *Data mining concepts and techniques* (3rd ed.). Waltham, Mass: Morgan Kaufmann Publishers.
Haraty, R. A., Dimishkieh, M., & Masud, M. (2015). An enhanced k-means clustering algorithm for pattern discovery in healthcare data. *International Journal of Distributed Sensor Networks, 11*, 615–740. https://doi.org/10.1155/2015/615740
Harikumar, S., & Surya, P. V. (2015). K-medoid clustering for heterogeneous datasets. *Procedia Computer Science, 70*, 226–237. https://doi.org/10.1016/j.procs.2015.10.077. URL: http://www.sciencedirect.com/science/article/pii/S187705091503241X.
Jha, M. (2015). Document clustering using k-medoids. *International Journal on Advanced Computer Theory and Engineering, 4*, 54–58. https://arxiv.org/ftp/arxiv/papers/1504/1504.01183.pdf.
Kamat, P., Chodankar, R., Harmalkar, R., Naik, G., & Narulkar, P. (2017). Document clustering using divisive hierarchical bisecting min max clustering algorithm. *IOSR Journal of Computer Engineering, 19*, 66–70. https://doi.org/10.9790/0661-1903066670
Kotouza, F. E. P., & Mitkas, P. A. (2019). A dockerized framework for hierarchical frequency-based document clustering on cloud computing infrastructures. *Journal of Cloud Computing, 9*, 1–12. https://doi.org/10.1186/s13677-019-0150-y
Kurada, R. R., & Pavan, K. (2016). Automatic unsupervised data classification using jaya evolutionary algorithm. *Advanced Computational Intelligence: An International Journal (ACII), 3*, 35–42. https://doi.org/10.5121/acii.2016.3204
Lubna Alhenak, M. H. (2019). Genetic-frog-leaping algorithm for text document clustering. *Computers, Materials & Continua 61*, 1045–1074. URL: http://www.techscience.com/cmc/v61n3/35288, 10.32604/cmc.2019.08355.
Lydia, L., Govindasamy, P., Lakshmanaprabu, S., & Ramya, D. (2018). Document clustering based on text mining k-means algorithm using euclidean distance similarity. *Journal of Advanced Research in Dynamical and Control Systems, 10*, 208–214.
Mahdavi, M., & Abolhassani, H. (2009). Harmony k-means algorithm for document clustering. *Data Mining and Knowledge Discovery, 18*, 370–391. https://doi.org/10.1007/s10618-008-0123-0
Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to information retrieval*. USA: Cambridge University Press.
Metre, V. A., Popat, S. K., & Deshmukh, P. B. (2017). Optimization of document clustering using unl document vector generation and swarm intelligence. In *2017 International conference on computing, communication, control and automation (ICCUBEA)* (pp. 1–6). https://doi.org/10.1109/ICCUBEA.2017.8463860
Mohammed, A., Yusof, Y., & Husni, H. (2015). Document clustering based on firefly algorithm. *Journal of Computer Science, 11*, 453–465. https://doi.org/10.3844/jcssp.2015.453.465
Nguyen, M. D., & Shin, W. (2019). An improved density-based approach to spatio-textual clustering on social media. *IEEE Access, 7*, 27217–27230.
Pamulaparty, L., Rao, C., & Rao, D. (2014). A near-duplicate detection algorithm to facilitate document clustering. *International Journal of Data Mining and Knowledge Management Process (IJDKP), 4*, 39–49. https://doi.org/10.5121/ijdkp.2014.4604
Rashaideh, H., Sawaie, A., Al-Betar, M. A., Abualigah, L. M., Al-laham, M. M., Al-Khatib, R. M., & Braik, M. (2018). A grey wolf optimizer for text document clustering. *Journal of Intelligent Systems, 29*, 814–830. https://doi.org/10.1515/jisys-2018-0194
Roul, R., Asthana, S., & Sahay, S. (2015). Automated document indexing via intelligent hierarchical clustering: A novel approach. In *2014 International conference on high performance computing and applications ICHPCA 2014*. https://doi.org/10.1109/ICHPCA.2014.7045347
Rousseeuw, P. J. (1987). Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics, 20*, 53–65. https://doi.org/10.1016/0377-0427(87)90125-7. URL: http://www.sciencedirect.com/science/article/pii/0377042787901257.
Saini, N., Saha, S., & Bhattacharyya, P. (2018). Automatic scientific document clustering using self-organized multi-objective differential evolution. *Cognitive Computation, 11*, 271–293. https://doi.org/10.1007/s12559-018-9611-8
Samir, B., Ahmed, S., & Mansour, S. (2014). Optimized k-means algorithm. *Mathematical Problems in Engineering, 2014*, 1–14. https://doi.org/10.1155/2014/506480
Sreedhar, C., Kasiviswanath, N., & Reddy, P. (2017). Clustering large datasets using k-means modified inter and intra clustering (km-i2c) in hadoop. *Journal of Big Data, 4*, 1–19. https://doi.org/10.1186/s40537-017-0087-2
Suraj, Sinha, & Ghosh S., R. K. (2016). Classification of two class motor imagery task using jaya based k-means clustering. In *2016 International conference on global trends in signal processing, information computing and communication (ICGTSPICC)* (pp. 175–179).
Venkata Rao, R. (2016). Jaya: A simple and new optimization algorithm for solving constrained and unconstrained optimization problems. *International Journal of Industrial Engineering Computations, 7*, 19–34. https://doi.org/10.5267/j.ijiec.2015.8.004
Vidyadhari, C., Sandhya, N., & Premchand, P. (2019). Particle grey wolf optimizer (pgwo) algorithm and semantic word processing for automatic text clustering. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 27*, 201–223. https://doi.org/10.1142/S0218488519500090
Wang, S. H., Phillips, P., Dong, Z. C., & Zhang, Y. D. (2018). Intelligent facial emotion recognition based on stationary wavelet entropy and jaya algorithm. *Neurocomputing, 272*, 668–676. https://doi.org/10.1016/j.neucom.2017.08.015. URL: http://www.sciencedirect.com/science/article/pii/S0925231217313644.
Yang, H. (2010). A document clustering algorithm for web search engine retrieval system. In *2010 International Conference on e-Education, e-Business, e-Management and e-Learning* (pp. 383–386). doi: 10.1109/IC4E.2010.72.

Karpagalingam Thirumoorthy received his M.E. degree from Arulmigu Kalasalingam College of Engineering, Krishnankoil, Tamilnadu, India and B.E. degree from Kamaraj College of Engineering and Technology, India. He worked as an Assistant Professor in Computer Science and Engineering department of Mepco Schlenk Engineering College, Sivakasi for 11 years and currently pursuing Ph.D in Mepco Schlenk Engineering College, Sivakasi, India under Anna University, Chennai, India. His research areas include Text mining and Data mining.

Karuppaiah Munneswaran is presently the Senior Professor in the Department of Computer Science and Engineering at Mepco Schlenk Engineering College, Sivakasi. He completed his Doctorate in M.S University, Tirunelveli, India and his Post Graduate Degree from PSG College of Technology, Coimbatore, India and Bachelor Degree from Thiagarajar College of Engineering, Madurai, India. He has nearly 33 years of teaching experience. He has published nearly 52 papers in reputed International Journals with 531 citations and 95 papers in International and National Conferences. He has published a book entitled 'Compiler Design' in Oxford University Press.