Neural Computing and Applications
https://doi.org/10.1007/s00521-020-04945-0

ORIGINAL ARTICLE

A novel hybrid multi-verse optimizer with K-means for text documents clustering

Ammar Kamal Abasi[1] · Ahamad Tajudin Khader[1] · Mohammed Azmi Al-Betar[2] · Syibrah Naim[3] · Zaid Abdi Alkareem Alyasseri[1,4] · Sharif Naser Makhadmeh[1]

Received: 26 July 2019 / Accepted: 15 April 2020

Abstract

Text clustering has been widely utilized with the aim of partitioning specific document collection into different subsets using homogeneity/heterogeneity criteria. It has also become a very complicated area of research, including pattern recognition, information retrieval, and text mining. Metaheuristics are typically used as efficient approaches for the text clustering problem. The multi-verse optimizer algorithm (MVO) involves a stochastic population-based algorithm. It has been recently proposed and successfully utilized to tackle many hard optimization problems. However, a recently applied research trend involves hybridizing two or more algorithms with the aim of obtaining a superior solution regarding the problems of optimization. In this paper, a new hybrid of MVO algorithm with the K-means clustering algorithm is proposed, i.e., the H-MVO algorithm with the aims of enhancing the quality of initial candidate solutions, as well as the best solution, which is produced by MVO at each iteration. This hybrid algorithm aims at improving the global (diversification) ability of the search and finding a better cluster partition. The proposed H-MVO effectiveness was tested on five standard datasets, which are used in the domain of data clustering, as well as six standard text datasets, which are utilized in the domain of text document clustering, in addition to two scientific articles' datasets. The experiments showed that K-means hybridized MVO improves the results in terms of high convergence rate, accuracy, error rate, purity, entropy, recall, precision, and F-measure criteria. In general, H-MVO has outperformed or at least proven to be highly competitive compared to the original MVO algorithm and with well-known optimization algorithms like KHA, HS, PSO, GA, H-PSO, and H-GA and the clustering techniques like K-mean, K-mean++, DBSCAN, agglomerative, and spectral clustering techniques.

Keywords Multi-verse optimizer (MVO) · Hybridization · Text clustering · k-Means clustering

✉ Ammar Kamal Abasi
ammar_abasi@student.usm.my

Ahamad Tajudin Khader
tajudin@usm.my

Mohammed Azmi Al-Betar
mohbetar@bau.edu.jo

Syibrah Naim
syibrah@wsu.ac.kr

Zaid Abdi Alkareem Alyasseri
zaid.alyasseri@uokufa.edu.iq

Sharif Naser Makhadmeh
m_shareef_cs@yahoo.com

1 School of Computer Sciences, Universiti Sains Malaysia (USM), 11800 Pulau Pinang, Malaysia

2 Department of Information Technology, Al-Huson University College, Al-Huson, Irbid, Jordan

3 Technology Department, Endicott College of International Studies (ECIS), Woosong University, Daejeon, Korea

4 ECE Department, Faculty of Engineering, University of Kufa, Najaf, Iraq

Published online: 11 May 2020

1 Introduction

In the current modern era, text document clustering (TDC) represents a significant research area that has been rapidly increasing [1]. It can be implemented in many applications of analysis with the aim of gathering a specific set of text documents into coherent classes or clusters' subsets [2]. Substantial efforts were exerted by many techniques so that documents within a specific cluster possess high intra-similarity, as well as low inter-similarity with other clusters [3]. In other words, similar documents are allocated for a similar cluster, whereas dissimilar documents are allocated for varied clusters [4]. The similarities and dissimilarities are evaluated based on the attribute values that describe the documents.

TDC is the essential problem of unsupervised learning, as it discusses the partition of the data in an unknown area. This text mining field allows for large amounts of textual data to be organized. It also forms the basis for any further unsupervised learning [2]. The TDC methods are used to classify the documents into related groups of some categories or specific topics. Nonetheless, these categories are not considered a priori. Since examples from the related groups of the documents are not given beforehand (e.g., politics), the focus of this paper on partition clustering procedures as this clustering algorithm aims to partition a cretin object of dataset into another subset consisting of similar clusters based on minimization or maximization of the objective function regardless of their hierarchical structure.

The clustering techniques in machine learning are applied in many application fields such as pattern recognition [5], information retrieval [6], data mining [7], and text mining [8]. The traditional clustering algorithms are categorized into two key categories, including hierarchical and partitional categories [1, 9, 10]. Hierarchical strategies can be categorized as divisive (top-down) and agglomerative (bottom-up) approaches. Firstly, divisive clustering begins with all the dataset documents in the one cluster and then tries to separate them into smaller clusters that are heterogeneous. In comparison, agglomerative clustering treats each document as an isolated cluster and then collect the homogeneous clusters until no further fusion is feasible [11]. Partitional clustering (i.e., flat partition) methods aim to divide text documents into sets of different clusters.

The K-means algorithm is a prevalent and widespread partitioning clustering algorithm due to the algorithm's superior feasibility, as well as efficiency to deal with gigantic amounts of data [12]. The most shared clustering objective in the K-means involves minimizing the number of similarities among all the objects, as well as the matching cluster centers. The K-means algorithm, however, performed inefficiently. In particular, it is vulnerable to outliers because, in addition to its sensitivity to the initial centroids, it can fall in the local regions in the search space [13].

Over a period of around two decades, the metaheuristic algorithms have provided successful solutions to solve the TDC problem [1]. These algorithms are categorized into population-based and single-based algorithms based on the solutions number that are provided in each iteration [14] such as ray optimization algorithm (ROA) [15], harmony search (HS) [16], grey wolf optimizer (GWO) [17], cuckoo search (CS) [18], salp swarm algorithm (SSA) [19], fruit fly optimization algorithm (FFOA) [20], dragonfly algorithm (DA) [21], krill herd algorithm (KHA) [22], teaching-learning-based optimization (TLBO) [23], dolphin echolocation (DE), ant lion optimizer (ALO) [24, 25], particle swarm optimization (PSO) [26], and ant colony optimization (ACO) [27].

The multi-verse optimizer (MVO) method represents that an evolutionary population-based algorithm is inspired by the multi-verse theory [28]. The MVO algorithm is part of an expanded swarm intelligence group developed by Mirjalili et al. [29]. In the MVO algorithm, each of the universes is modified in a search space based on receiving good attributes from better universes. The MVO algorithm has several advantages. During a specific search, it is smoothly balancing exploration, as well as exploitation. Besides, a few parameters are set in the initial stage because there is no mathematical derivation required like other algorithms. It is also flexible, adaptable, simple, sound-and-complete, and scalable. MVO has been used to address NP hard problems, e.g., clustering problems [30], unmanned aerial vehicle path planning [31], neural networks [32], oil recovery [33], feature selection [34, 35], and optimizing SVM parameters [35].

Many researchers, who investigated text clustering, have applied the metaheuristic algorithms successfully to solve the problem of TDC. However, a key drawback of such methods manifests that these methods often start with the creation of a random solutions' group, and then, the initial solutions are moved, evolved, or combined over the iterations or generations during the execution. Until today, random has been the standard method to create the initial population [36]. Therefore, the obtained results of the algorithm depend (among other factors) on the solutions' quality in an initial population [2, 37]. Besides, other problems can involve unsatisfactory results, e.g., inaccurate clusters, as well as the behavior of the algorithms, which were chosen unlike the problem of the text document clustering. Besides, the best solution in MVO plays a key role in the exploration and exploitation phases. In this situation, improving the best solution at each iteration is helpful to find better solutions. Two and more algorithms are hybridized with the aim of obtaining the best solution to tackle the optimization problems [38].

In this paper, two versions of the hybrid strategy of the MVO algorithm are proposed called hybrid multi-verse optimizer algorithm (H-MVO) with the aim of enhancing the initial candidate solutions' quality, as well as the outcomes that are produced by the basic MVO algorithm for TDC. In the first version, the hybridization occurs in the ordering of the initialization phase, where the initial population process is invoked. The k-means clustering algorithm uses these solutions as input to improve each solution. In the second version, the hybridization occurs for the best solution, which is produced by MVO at each iteration, and it is considered as the initial state of k-means, including the first version. Based on the K-means and MVO principle, these two approaches are merged with the aim of proposing a novel hybrid MVO to solve the MVO shortcomings so that the value of the optimal fitness function is obtained. This proposed method is assessed on the commonly used standard benchmark datasets that are utilized in the data and text clustering domain. The results are compared with the existing comparative techniques and algorithms. The experimental results demonstrated that the H-MVO performed more accurately and efficiently than the well-known clustering techniques (i.e., K-mean, K-mean++, DBSCAN, agglomerative, and spectral), and the original, as well as the hybrid optimization algorithms (i.e., KHA, GA, PSO, HS, MVO, H-PSO, and H-GA).

The rest of the paper is divided into six sections. The second section discusses related works and previous studies. The third section introduces and discusses the TDC problem in detail. The MVO algorithm is addressed by the fourth section, while the fifth section explains the proposed methodology. The sixth section includes the dataset used, the experimental findings, and the proposed method's significance. The conclusion, along with guidelines for further studies, is provided in the seventh section of the current paper.

2 Related works

This section addresses that the related works of non-metaheuristics and metaheuristics algorithms for TDC are reviewed.

2.1 Non-metaheuristics for TDC problem

K-means [39], K-medoids [40], as well as fuzzy c-means clustering [41] represent the conventionally traditional partitional clustering algorithm's examples. All the conventional partitional clustering algorithms are scalable easily to larger datasets. These algorithms, however, do not target a global convergence because they extremely depend on the cluster centers' initial position, and they often converge to the nearest local optimum solution in a search space from a starting search position. Also, the algorithm's multiple runs cannot solve the problem to achieve the local optimum solution.

The main objective of the K-means is to update the clusters centroid characterized by the center of data points. The process continues until the iterative calculation fulfills a specific convergence criterion. Chen [42] used K-means with other clustering algorithms after proposing a novel scheme for the distance-based term weighting to encode the term weights through considering distances among the news terms and whether terms have occurred. The proposed work presents the potential to improve clustering performance.

Hussain and Haris [43] embed exploiting the statistical information of the data into k-means algorithm instead of utilizing them as an external distance measure and presenting a specific integrated framework, namely the k-means-based co-clustering (kCC) algorithm. Also, the initialization step is modified to involve multiple points with the aim of representing every cluster center like the points within a specific cluster that are close altogether; however, they are far from the points that represent other clusters. Furthermore, the neighborhood walk statistics are suggested as the semantic similarity approach for the cluster assignment, as well as the center re-estimation in an iterative process. The evaluation of the combined approach was carried out on some standard datasets. The proposed approach (i.e., kCC) outperformed the k-means, k-means++, ICC k-means, hierarchical ensemble clustering, and SSID k-means traditional algorithms.

The K-medoids text clustering algorithm functions in a similar way as the K-means text clustering. It begins by selecting k documents randomly as initial medoids with the aim of representing k clusters. Other documents that are close to the medoid are involved in the cluster. Subsequently, a novel medoid is chosen, which better represents the cluster. The documents are assigned to the clusters, which have the closest medoid. The medoids modify their location in each iteration. This method works on minimizing the number of dissimilarities among documents and their corresponding medoid. The cycle is repetitive until no medoid modifies its placement. The process ends here, and the final clusters along with the medoids are defined. The formed K clusters are centroid on the medoids. All the members of the documents are put in the most appropriate cluster depending on the nearest medoid [44].

Balabantaray et al. [45] compare the K-means clustering with the K-medoids clustering. K-means was carried out using both Euclidean and Manhattan distance on WEKA tool, and K-medoids was carried out through Java programming. Finally, it was observed that K-means yielded a better result than K-medoids. The use of the k-medoid clustering algorithm suffers from a couple of disadvantages. First, it needs many repetitions so that convergence is reached in addition to the slow implementation. It is because each of the iterations needs similarity computation or distance measures. Second, the k-medoid clustering algorithms cannot be compatible with the sparse text collection. Also, in the large division, as well as the documents' non-uniform distribution, a text does not include several words in common; the similarity value is quite small among these document pairs [46].

2.2 Metaheuristics algorithms for TDC problem

Over two decades, some nature-inspired metaheuristics were proposed to be applied in several real-life applications. Recently, to solve many unsupervised optimization problems, metaheuristic algorithms have been successfully implemented. In the current stage of any problem associated with unsupervised optimization, the user can select an appropriate metaheuristic algorithm easily to solve it. The obtained solution can ensure optimality as the population-based algorithms discover all the search space of the progress in the generations [47]. The next subsections outline the recently conducted works that investigated the partitional clustering.

2.2.1 Particle swarm optimization algorithm

The particle swarm optimization (PSO) is a well-known swarm-based algorithm [48]. This algorithm has been inspired by the birds' social behavior, which simulates the birds' collective intelligence [49].

The PSO's key characteristic involves the simple way of sharing information among the agents according to a few equations. The agent is referred to as a particle, and a swarm is formed by a group of these particles. The location of the particles is in a multidimensional search space; they change positions based on the best position accomplished so far (i.e., self-experience), as well as positions of the remaining swarm (i.e., collective experience). A fitness function is usually utilized in evaluating the agents' quality. The group of these steps permits a complex global behavior's emergence [50].

In the optimization problems, the positions of the particles signify the candidate solution to solve the given problem. In clustering, however, it relies on codification [51].

2.2.2 Genetic algorithm

GA-based clustering signifies an evolutionary algorithm, which is inspired by the natural selection process. The method that is based on the GA can be applied to the TC utilizing the ontology strategy, as well as the strategy of thesaurus [52]. Also, two-hybrid methods are applied utilizing different similarity measures. As a result, the GA, which is associated with the suggested similarity measures, has enhanced the TC method's performance.

The novel TC technique is based on the partitioning of a specific dataset into further subset groups and the application of the GA separately to each of the clusters instead of the whole dataset [53]. The separate application of the GA to the partitions with the aim of avoiding the local minima involves the key problem when the GA is used. The introduced GA has achieved better performance in comparison with previous approaches.

A novel approach has been recently enhanced to tackle the text data structure, in other words, MEDLINE abstract dataset, which relies on combining the GA with the VSM, as well as the agglomerative algorithm [54]. The experiments included a subset of MEDLINE dataset that is utilized in real applications. Accordingly, this proposed method is applicable to any of the text dataset to boost information retrieval.

2.2.3 Artificial bee colony algorithm

The artificial bee colony algorithm (ABC) is enhanced by Karaboga [55] and inspired by intelligent foraging of real honey bees' behavior with the aim of locating food sources. Honey bees can be classified into three categories: employed bees, onlooker bees, and scout bees. Moreover, half of the bee colony is occupied by employed bees, whereas the other half is occupied by onlooker bees. Inside the colony, the employed bees work on searching and exploiting the existing food sources around the hive, as well as sharing information of the nectar quantity of the sources of food with the onlooker bees.

The onlooker bees work on selecting and exploiting the food sources based on the information, which is shared by the employed bees. Also, the more nectar the food source has, the higher the possibility that the onlooker bees select it.

In turn, the employed bees that have exhausted food sources become the scout bees. The scout bees' occurrence is controlled via a parameter, namely (limit). The whole search space is explored by the scout bees and, therefore, new food sources are randomly generated instead of the food source, which is exhausted. Each of the food sources matches a specific solution to the given problem. The nectar quantity, which is related to the food source, matches the solution fitness.

Like the existing nature-inspired algorithms, the ABC algorithm begins with initializing few parameters, as well as arbitrarily generated solutions. This algorithm aims to optimize a given objective function with the employed bees' repeated number of cycles, the onlooker bees, as well as phases of the scout bees in a sequence [56].

2.2.4 Krill herd algorithm

The herding individual krill behaviors inspired the krill herd algorithm (KHA). This algorithm is a new evolutionary algorithm, which is based on swarm intelligence, as well as the bacterial foraging algorithms. KHA was presented by Gandomi and Alavi [22]; it attracted the researchers, and it was widely utilized to solve many optimization problems due to the simple idea, as well as the concept it presents, in addition to its easy implementation, and its suitable behavior for the clustering techniques [22].

2.2.5 Harmony search algorithm (HS)

Another optimization algorithm, which is powerful in the exploration search, is the harmony search (HS); however, it can, sometimes, be trapped in the local optima [57] and, therefore, conducting a global search efficiently is very difficult. Based on the HS algorithm, the search mechanism entirely depends on random and, thus, to reach fast convergence is quite hard. Forsati et al. [16] have developed HSCLUST to solve TDC problem. An efficient enhancement is expected regarding the proposed HSCLUST as it includes the K-means algorithm to devise three different hybrid methods in order to find a near-optimal partition.

2.3 Hybrid metaheuristic methods for text document clustering problem

A hybrid approach, which is based on the PSO algorithm, as well as the K-harmonic mean algorithm is suggested for the technique of data clustering. The proposed approach completely utilizes the merits of the two algorithms. This proposed hybridization algorithm helps the K-harmonic mean clustering in escaping from the local optimum solution, and it wins over the limitations via tuning the PSO algorithm's convergence speed. The proposed hybridization algorithm's performance is examined when seven datasets are used for the data clustering technique from the UCI Machine Learning Repository. It is also compared against the swarm optimization and the K-harmonic mean clustering standalone. The experimental results showed that this hybridization technique is superior [58].

Four hybridization versions of the ABC and the PSO are utilized, including sequence and parallel, as well as the sequence with an enlarged pheromone-particle table, in addition to the global best exchange approaches, were introduced with the aim of improving the technique of data clustering. The hybrid versions have been examined via the problem of data clustering. The experimental results revealed that the proposed methods' performance is superior in comparison with the existing standalone algorithms. The experiments were carried out utilizing standard datasets from the UCI Machine Learning Repository. Thus, among the hybridization versions, the sequence approach outperformed all the existing approaches. This is because growth diversifies throughout the new solutions' generation and, therefore, prevents from being stuck in local optimum [59].

The hybrid ABC algorithm is introduced to enhance the technique of data clustering. The key goal of this hybrid algorithm involves enhancing social learning among bees by the addition of the genetic algorithm's crossover operator to the artificial bee colony. Consequently, ten benchmark functions, as well as six datasets, were utilized to examine the technique of the data clustering from the Machine Learning Repository (UCI). In conclusion, the results showed that the introduced algorithm performed in a better manner compared to the existing algorithms. Also, better results were obtained in the technique of the data clustering [59].

A novel hybrid algorithm called differential evolution KH (DEKH) was introduced to solve the function optimization with the aim of overcoming the KH algorithm's poor intensification [60]. Such an improvement was made via appending the hybrid differential evolution (HDE) into the KH to tackle complex optimization issues in a more efficient way. The HDA works on inspiring intensification and encouraging the krill so that the intensification search within the defined region is performed. The experiments were carried out on 26 optimization functions, and the results showed that the suggested DEKH is suitable for finding an accurate solution compared to the KH, as well as the existing comparative methods. Furthermore, the DEKH method's robustness and the initial population volume's control over convergence, in addition to its effectiveness, were tested by conducting a group of experiments.

A novel hybrid strategy called cuckoo search and krill herd (CSKH) was introduced to make the KH works in a more efficient way [61]. The CSKH involves krill updating (KU), as well as abandoning (KA) operator, which is introduced from CS throughout the process when the krill position has been updating with the aim of enhancing its performance significantly, as well as its reliability in dealing with problems of function optimization. The KU operator works on encouraging the exploitation, as well as allowing krill individuals to carry out a careful search. However, the KA operator can be applied to further enhance the CSKH's exploration search instead of poor krill at the end of each iteration. This strategy's performance is evaluated via 14 standard optimization functions. The results revealed that the suggested hybrid strategy of the CSKH algorithms can be more powerful, as well as efficient compared to the basic KH and existing comparative methods.

A new hybrid of the KH algorithm, associated with the harmony search (HS) algorithm called H-KHA, was proposed to enhance the ability of the global (diversification) search. The improvement involves the addition of the global search operator (a new solution is improvised) of HS algorithm to KH algorithm to enhance the ability of the exploration search via a new factor of probability called distance factor, whereby krill individuals are moved toward the best global solution. The H-KHA's performance outperformed or at least it was found to be highly competitive compared to the original KH algorithm, the clustering techniques that are well-known, as well as the existing comparative optimization algorithms [10].

The obtained results, as well as the analysis of these studies, were successful to some degree in improving the clustering performance; they have overcome the local methods such as K-Means and k-medoids. However, from a practical point of view, there is no right balance between exploitation and exploration during the optimization process in metaheuristic algorithms. This means that moving toward the global optimum solution is not guaranteed [29]. Besides, they usually start by creating a set of random solutions; these initial solutions are then moved, evolved, or combined over the iterations or generations during the execution. Until today, random has been the standard method to create the initial population. Therefore, the obtained results of the algorithm depend (among other factors) on the quality of the solutions in the initial population [2]. To solve the TDC problem using metaheuristic algorithms, therefore, needs further and in-depth investigation to overcome the existing weaknesses.

3 Text document clustering problem

The TDC represents the NP-complete problem to find clusters in the heterogeneous documents via minimizing the objective ($\min f(\boldsymbol{x})$ equals the minimizing of the Euclidean distance function in the present paper). The TDC problem formulation, the text document preprocessing, the clustering algorithm, similarity measures, and the objective function are described in this section of the paper.

3.1 Problem formulation

The TDC problem is described at a high level as follows: Given a set of d documents, in order to divide the ($Docs$) into a predetermined number (K) clusters, where ($Docs$) represents a vector of documents ($Docs = (d_1, d_2, \ldots, d_i, \ldots, d_n)$). Doc_i means the number of document i and Doc_n signifies all the number of documents in $Docs$. Each of the clusters possesses a cluster centroid (K_{cent}) that is defined as a vector of terms length weights f ($k_{cnt} = (k_{cnt1}, k_{cnt2}, \ldots, k_{cntj}, \ldots, k_{cntf})$), where k_{cnt} is the centroid of the k_{th} cluster, k_{cnt1} signifies the value of position 1 in the cluster centroid k, and k_{cntf} signifies all the exceptional centroid features‘ number (terms) [40, 62].

In order to determine a partition $k_{cnt} = (k_{cnt1}, k_{cnt2}, \ldots, k_{cntj}, \ldots, k_{cntf}$ fulfilling the conditions:

1. $k_{cnt} \neq \emptyset$. each cluster must not be empty (i.e., each centroid must attract at least one document).
2. $k_{cnt} \bigcap k_{cnt'} = \emptyset \quad if \quad K \neq K', \quad \bigcup_{k=1}^{K} k_{cnt} = 0$ each cluster must contain unique documents (i.e., Hard clustering).
3. The objects that belong to a similar cluster bear a high resemblance to each other, whereas the objects that belong to varied clusters are not like each other.

3.2 Text document preprocessing

Before the clustering algorithm is applied, the steps of the standard preprocessing are utilized with the aim of preprocessing the text documents, which include tokenization, stop words removal, stemming, and term weighting steps. The text documents are converted into numerical format or matrix by the steps of the preprocessing. A brief description of the standard preprocessing steps is given in the following sections.

1. Tokenization

   In such a process, the text data are split into basic independent units sequence (i.e., words or terms) called tokens.
2. Stop words removal

   The process of the stop words removal works on removing specific common words, which are most frequently occurred. Examples of these words include ‘is, ‘are’, ‘am’, ‘that,’ and ‘an’. A list of 571 words can be obtained from (http://www.unine.ch/Info/clef/), that contains 571 words.
3. Stemming

   Stemming involves converting words into their roots to achieve simplicity, i.e., ‘introduce’ and ‘introduction’ are treated like ‘introduce.’ Also, words like, ‘computer,’ ‘computes,’ ‘computation,’ and ‘computing’ are treated like ‘compute.’
4. Term weighting

   Term weighting aims at converting text data into a particular numerical format. There exist many schemes of term weighting in the literature. For text document representation, term frequency-inverse document frequency scheme (TF-IDF) is calculated in the vector space model (VSM)[63]. In the VSM, each document is shown with a vector, and the values of the cells are weighted as follows:

$$d_i = \{F_{i,1}, F_{i,2}, \ldots, F_{i,j}, \ldots, F_{i,t}\},$$

where t refers to the features number and F_{ij} refers to the feature j weight in document i [64]. For calculating the weighting of the feature, the following weighting scheme is used:

$$F_{i,j} = \ TF - IDF(i,j) = TF(i,j) \times \left(\log \frac{d}{DF(j)}\right) \quad (1)$$

where $TF(i, j)$ the feature j frequency in document i, and $DF(j)$ is all the documents containing feature j.

Matrix of size $m \times n$ is used to represent the VSM as follows:

$$VSM = \begin{pmatrix} F_{1,1} & F_{1,2} & F_{1,(t-1)} & F_{1,t} \\ \vdots & \vdots & \ddots & \vdots \\ \cdots & \cdots & \cdots & \cdots \\ F_{(m-1),1} & \cdots & \cdots & F_{(m-1),t} \\ F_{m,1} & F_{m,2} & \cdots & F_{m,t} \end{pmatrix}$$

3.2.1 Solution representation

The solution within the population means a independence candidate solution that aims to solve the TDC problem as a vector $\boldsymbol{x} = (x_1, x_2, \ldots, x_d)$, where d signifies the number of documents and the values of the cells are assigned by any cluster number k as represented in Fig. 1. Based on Fig. 1, the solution X has twenty dimensions, documents are divided into five clusters, for instance, document one is in cluster five, and cluster four has four documents $\{13, 14, 17, 18\}$.

3.2.2 Objective function

The distance/similarity is used in the clustering methods as an objective function to assess the degree of closeness between the cluster centroid and the documents. In this paper, The Euclidian distance measure is used. This measure is commonly used as a standard measure for the same purpose [65, 66].

For instance, given document $d_1 = (F_{11}, F_{12}, \ldots, F_{1t})$ and $k_{cntj} = (k_{cntj1}, k_{cntj2}, \ldots, k_{cntjj}, \ldots, k_{cntf})$, the Euclidean distance is as follows:

$$distance(d_1, k_{cntj}) = \left(\sum_{i=1}^{t} |F_{1,i} - k_{cntj,i}|^2\right)^{1/2}, \quad (2)$$

whereby $F_{1,i}$ signifies the weight of feature i in document 1, and $k_{cntj,i}$ signifies the feature weight i in cluster centroid j. It is worth noting that the distance value is within (0, 1), where the most appropriate value is 0, and the worst value is 1. For each cluster, the centroid is recalculated using Eq. (3) as follows:

$$k_{cnti} = \frac{\sum_{j=1}^{d} (a_{ij}) d_{jF}}{\sum_{j=1}^{d} a_{ij}}, \quad (3)$$

where k_{cntj} signifies the centroid cluster of j, d_{jt} signifies to the t_{th} feature weight of document i, a_{ij} signifies a binary dimension matrix $d \times k$ as follows:

$$a_{ij} = \begin{cases} 1 & \text{cluster } j \text{ contains the document} i, \\ 0 & \text{otherwise} \end{cases} \quad (4)$$

Eventually, for every $\boldsymbol{x}$ solution the $f(\boldsymbol{x})$ must be formulated of the proposed algorithm . As shown in Eq. 5, the average distance of documents to the cluster centroid (ADDC) is used.

$$\min f(\boldsymbol{x}) = \frac{\sum_{i=1}^{k} \left(\frac{1}{d_i} \sum_{j=1}^{d_i} distance(d_j, k_{cnti})\right)}{k}, \quad (5)$$

whereby $f(\boldsymbol{x})$ signifies the cost function, and d_i signifies the documents number in cluster i.

4 Original multi-verse optimizer

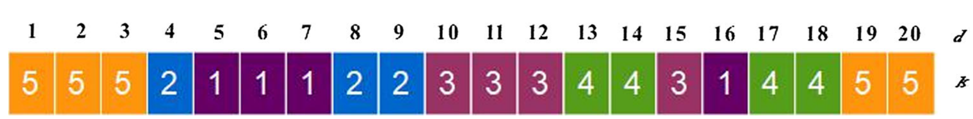
This section discusses the inspiration of MVO in Sect. 4.1, as well as the mathematical model in Sect. 4.2.

4.1 Inspiration

The inspiration might be from natural phenomena, animals, or social activity behavior. The algorithm is based on various potential mathematical models following the inspiration. MVO was enthusiastic mainly by the multi-verse theory in physics science [28].

The main components of the multi-verse theory include the wormhole, the black, and the white holes. These are mathematically designed for MVO. Based on this theory, physicists believed that there exists more than a single big bang and each bang has caused the birth of the universe.

Fig. 1 Solution representation



The theory of multi-verse conceptual models, as well as the big bang theory, is illustrated in Fig. 2.

A multi-version approach is a contradictory approach to the universe. This reveals that in addition to ours, there is another universe where we live according to the multi-verse principle. Three critical MVO-based principles are used in MVO's inspiration algorithm (i.e., wormhole, black, and white holes). Physicists thought the big bang is the white hole, which is a critical element of the universe's creation. The black holes behave differently from the white holes. They have a compelling gravitational force and can attract everything, even light beams. Based on the inspiration process, every universe cab can expand space by using the inflation rate. For white holes and stars as well as asteroids and planets, the universe's inflation rate is essential. Besides black holes and laws, the exact inspiration of MVO is illustrated in this scenario.

In terms of optimization, both the white holes and black holes definitions are used for the exploration phase. In multi-version theory, each universe corresponds to the candidate solution and each object in the universe corresponds to the decision variable. This means that the wormhole concept ensures the exploitation phase.

4.2 Mathematical model and algorithm

To explore the search spaces, MVO uses the concepts of the white holes and the black holes. In contrast, it uses wormholes to exploit search spaces. As any population-based algorithm, the optimization process often begins with the creation of a population of unique solutions and aims to develop solutions over the number of iterations predefined. On the basis of the MVO algorithm, individual improvements can be achieved in each population based on one of the theories of the potential existence of the multiple universes. By these ideas, each optimization solution is a single universe, and each object in this universe is a decision variable of a given problem. MVO thus accompanies such moves during the optimization process. In Fig. 3, the MVO flowchart is shown and the implementation of the MVO is provided in algorithm 1.

1. A higher inflation rate is the higher probability of white hole.
2. A higher inflation rate is the lower probability of black hole.
3. The objects in the universe transfer from a white hole into a black hole.
4. Randomly in all universes, the objects travel toward the best universe.

The universes that have a lower inflation rate (i.e., black hole) receive the objects from the universe that have a higher inflation rate (i.e., white hole). In this case, the average inflation rates of the universe are improved via iterations. The universes are arranged according to their inflation rates in every iteration; the white hole is assigned to the universe by the roulette wheel selection. This procedure has been formulated in a matrix (6).

$$\mathrm{U} = \begin{bmatrix} x_1^1 & x_1^2 & \cdots & x_1^d \\ x_2^1 & x_2^2 & \cdots & x_2^d \\ \vdots & \vdots & \vdots & \vdots \\ x_n^1 & x_n^2 & \cdots & x_n^d \end{bmatrix}, \tag{6}$$

where U is $n \times d$ matrix of universes called population, n signifies the number of universes, and d signifies the number of dimensions. Usually, the decision variable (dimension) j in solution i is generated randomly as follows:

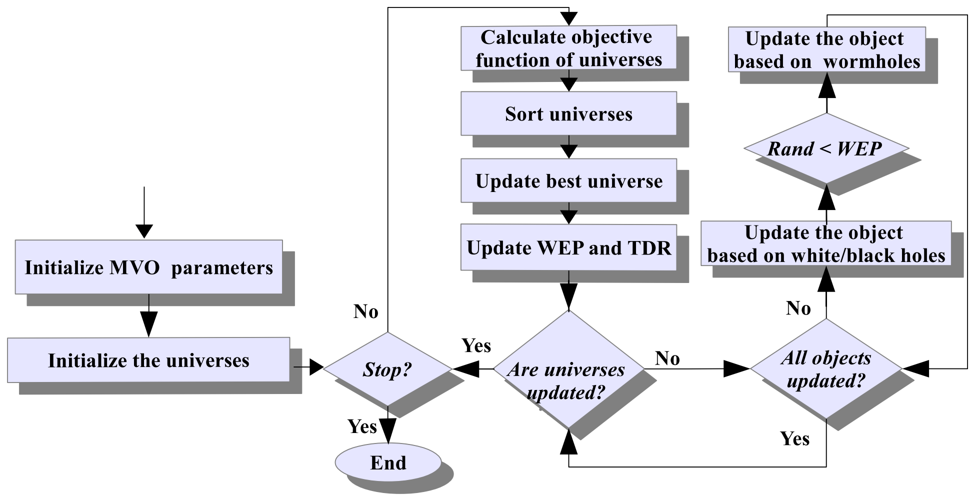
$$\begin{aligned} x_i^j &= lb_j + rand()\%((ub_j - lb_j) + 1) \\ &\forall i \in (1, 2, \ldots, n) \wedge \forall j \in (1, 2, \ldots, d), \end{aligned} \tag{7}$$

where $[lb_j, ub_j]$ signifies the limitation of decision variable j (lower and upper bound) and $rand()$ denotes a function to generate a random number in the range (0, 1).

In each of the iterations, every decision variable j in the solution i that has the black hole (i.e., x_i^j) changed a value using two options. First, from the historical solutions value like $x_i^j \in (x_i^1, x_i^2, \ldots, x_i^{j-1})$. Second, the value remains the same. This is as described in Eq. (8).

Fig. 2 **a** Big bang theory conceptual model [67]. **b** Multi-verse theory conceptual model [68]

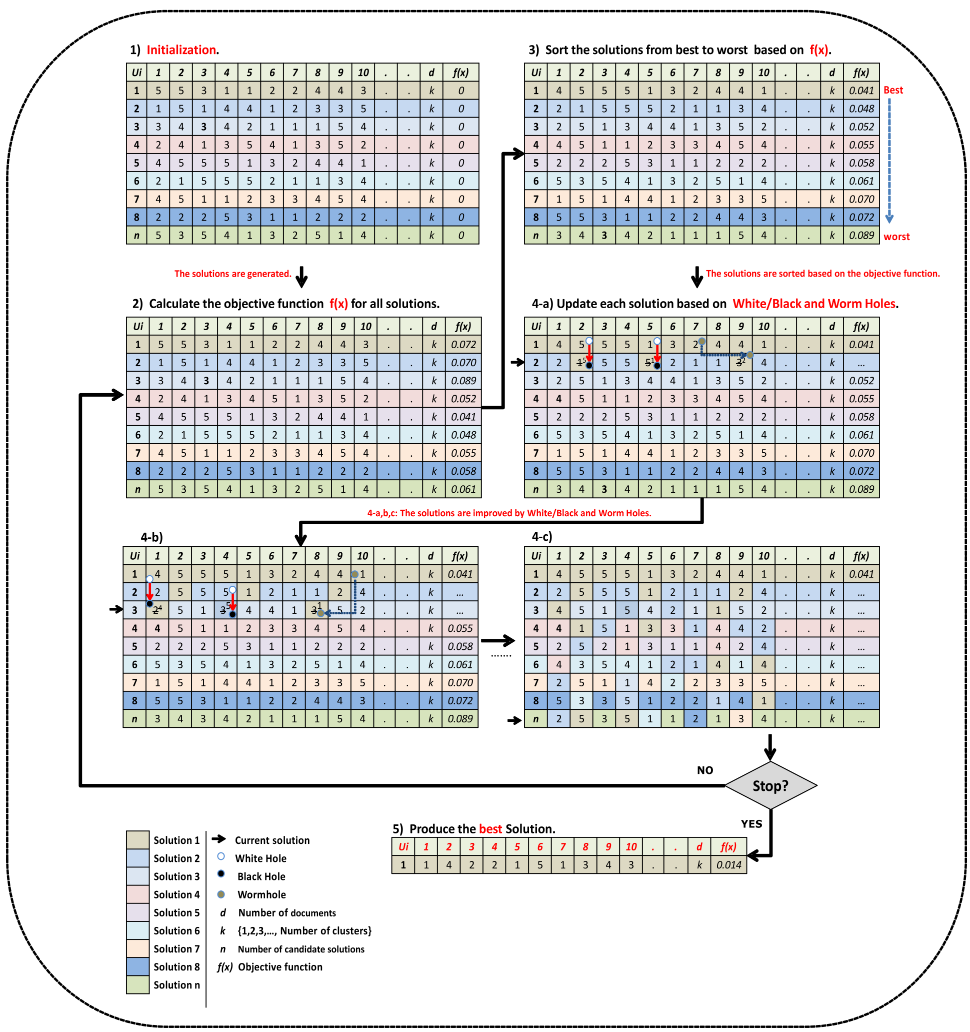
Fig. 3 Flowchart of MVO algorithm



$$x_i^j = \begin{cases} x_m^j & rand_1 < Norm(U_i), \\ x_i^j & rand_1 \geq Norm(U_i) \end{cases}, \tag{8}$$

whereby x_i^j shows the jth variable of ith universe, U_i indicates the ith universe, $Norm(U_i)$ denotes the normalized inflation rate of the ith universe, $rand_1$ denotes a randomly chosen number in the range [0,1], and x_m^j is the jth variable of mth universe, which is chosen by using the roulette wheel. In order to increase the diversity of the solutions, the wormhole is utilized, assuming that the

Fig. 4 Selecting the best solution using MVO



wormhole exists in the solutions randomly. Such a process is as follows:

$$x_i^j = \begin{cases} \begin{cases} x_j + TDR \times ((ub_j - lb_j) \times r_4 + lb_j) & r_3 < 0.5, \\ x_j - TDR \times ((ub_j - lb_j) \times r_4 + lb_j) & r_3 \geq 0.5 \end{cases} & r_2 < WEP \\ x_i^j & r_2 \geq WEP \end{cases}, \tag{9}$$

where x_j shows jth the fittest universe variable that is generated until now, lb_j signifies the minimum limit of jth parameter, ub_j signifies the maximum limit of jth parameter, x_j i is the jth parameter of ith universe, and $r2$, $r3$, $r4$ represent random numbers from [0, 1]. Therefore, it was concluded, based on this formulation, that the wormhole existence probability (WEP), as well as the traveling distance rate (TDR), represents the main coefficients and the coefficients' formula can be given by:

$$WEP = min + ci \times \left(\frac{max - min}{mi} \right) \tag{10}$$

$$TDR = 1 - \frac{ci^{1/p}}{mi^{1/p}}, \tag{11}$$

where min and max represent predefined constant values, mi is the maximum number of iterations, ci denotes current iteration, and p is the exploitation accuracy.

Figure 4 exemplifies the selection of the best solution using MVO. An optimal solution is the final result of that phase $\boldsymbol{x} = (x'_1, x'_2, \ldots, x'_d)$, which is the optimal solution of TDC problem. The following are the MVO steps:

1. Several TDC solutions are initialized (i.e., the number of clusters, documents, etc.).
2. Calculate the Euclidian distance (i.e., objective function $f(x)$) for all solutions.
3. According to the objective function, sort all solutions from the best to the worst.
4. Using the worm and black/white holes to update each solution.

   For the decision variable j of solution (x_i):

   - random value $r1$ is generated in order to compare against the (x_i) objective function value and exchange j by:

     Case 1: The value is replaced from better solutions.
     Case 2: The value remains the same.

   - Random value $r2$ is generated $r2$ in order to compare against the WEP and exchange i by:

     Case1: The value is replaced from the best solution after adding TDR value.
     Case2: The value is replaced from the best solution after subtracting the TDR value.

   Repeat step 4 and steps from 2 to 4 until the end criterion is satisfied.

5. Best solution is produced.

Algorithm 1 Pseudocodes and general steps of MVO algorithm

```
Initialize MVO parameters (Min, Max, Best solution = 0 , LB, UB, Number of solutions, Number of dimensions
(Dim) , Number of iterations ).
Create MVO population.
while the end criterion is not satisfied do
    For all solutions evaluate the objective function.
    The solutions are Sort_s=Sorted from the best to worst based on objective value .
    Normalize=Normalize the objective function for all solutions.
    Update the Best solution vector.
    for each solutions indexed by i except the best solution do
        Calculate WEP and TDR using Equations ( 10 and 11 ).
        Black_hole_index=i.
        for each object indexed by j do
            r1 = random([0, 1]).
            if r1 < Normalize(S_i) then
                White_hole_index = Roulette_Wheel_Selection .
                S(Black_hole_index,j)= Sort_s(White_hole_index,j).
            end if
            r2 = random([0, 1]).
            if r2 < Wormhole_existance_probability then
                r3 = random([0, 1]).
                r4 = random([0, 1]).
                if r3 < 0.5 then
                    Update the Position of Solution using Equation 9 case one.
                else
                    Update the Position of Solution using Equation 9 case two.
                end if
            end if

        end for
    end for
end while
Produce the best Solution.
```

5 Methodology

The details of the proposed method (i.e., H-MVO) is presented in this section. A proposed hybrid strategy of the MVO algorithm comprises two key stages. The first stage involves enhancing the quality of the initial candidate solutions. The second stage involves enhancing the best solution, which is produced by MVO at each iteration. Figure 6 shows the proposed hybrid strategy steps. Each phase is also described as follows.

5.1 K-means clustering algorithm

The K-means algorithm represents the most popular clustering algorithm. K-means is a prominent technique of partitional clustering. It was introduced over 50 years ago [1]. The K-means algorithm has been commonly utilized with the aim of dealing with huge databases due to its simplicity. Also, it is easily implemented, and it enjoys low computational complexity, as well as fast convergence [69]. The process involves two main steps' iterative. In this process, the entire dataset is classified into clusters that are heterogeneous. Over the years, many visions of improvement have been developed with the aim of enhancing its performance like the Kernel K-means [69], K-harmonic-means [7], and K-Medoids [69, 70].

By using the K-means algorithm, the data are split into K groups, which are characterized by their centroids (a cluster centroid typically represents the points' mean in the cluster), which arbitrarily generated artificial data to signify the whole group [71]. The algorithms' steps are originated by calculating the distances between samples and centroids. Every sample in the data is allocated to the closest centroid; each of the points' collection is allocated to a centroid that forms a cluster. Then, each cluster's centroid is updated in accordance with the points allocated to the cluster. The process is repeated until no point can change the clusters. Figure 5 exemplifies the way this method functions, in which the circles represent data and crosses represent the centroids [71]. The steps of the K-means algorithm are given in Algorithm 2.

5.2 Enhancing the quality of initial candidate solutions using K-means (H-MVO1)

The MVO algorithm is too sensitive to the initial condition solutions. Consequently, the algorithm's starting point significantly influences the quality of the final optimal solution [72].

Two or more algorithms were hybridized in recent studies with the aim of obtaining the best solutions to tackle the optimization problems [73, 74]. The key hybridization drivers involve avoiding the local optimization and enhancing the initial solutions, which leads to improving the global (diversification) search ability. This becomes more prominent in multidimensional problems. Based on such a perspective, the proposed hybrid algorithm combines the main features of the local search algorithm to enhance the initial solution, and then, the result can be passed as an initial solution for the MVO.

After the preprocessing step, the H-MVO starts with initializing a few parameters, and it randomly generates the solutions. Each solution is linked to one universe, i.e., the number of solutions and the number of universes are equal. It is worth mentioning that the initialization phase of the original MVO (Sect. 4 is adopted for clustering with some modifications. These modifications are related to the nature of the problem's variables, given that the clustering problem is discrete, and the MVO algorithm has initially been used for tackling continuous structural optimization problems [29]. The MVO is proposed to deal with discrete values of the decision variables of each TDC solution by using a rounding function to convert from continuous values to discrete values. The modifications are presented as follows:

1. The generation function of the initial solution Eq. 7 is modified:

$$x_i^j = \lfloor rand()\%(ub_j - lb_j))Bigr\rfloor \quad \forall i \in (1,2,\ldots,n) \wedge \forall j \in (1,2,\ldots,d), \tag{12}$$

   where n is the candidate number of solutions, d is the number of documents, $lb_j = 1$, ub_j is the number of

---

Algorithm 2 K-means clustering algorithm

Input: number of clusters K and a collection of objects D.
Output: distribute objects D to clusters K.
Steps:

Step1: select an initial objects as centroids clusters K .

Step2: **while** *the end criterion is not achieved* **do**
    Compute new cluster centroid for each k cluster.
    Assigning the objects to its closest cluster (less distance between the object and clusters centroids).
end

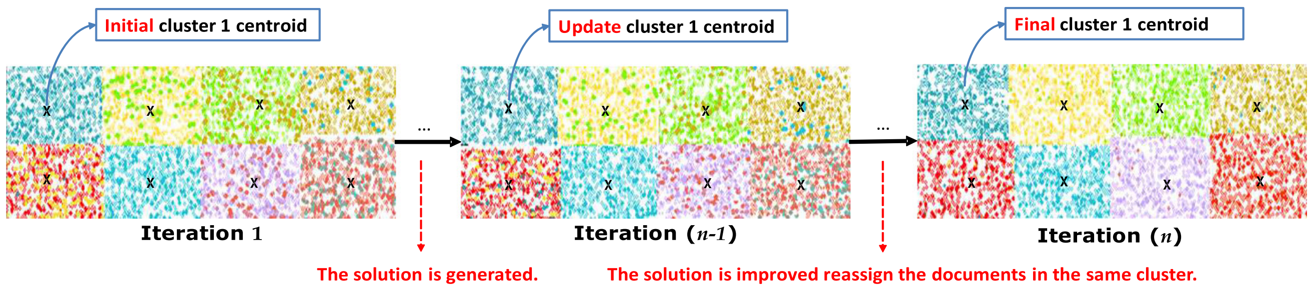


Fig. 5 Illustrative example of K-means operations

clusters, *Bigr* signifies a function, which rounds the value of x_i^j to the nearest integer, which is less than or equal to x_i^j, and *rand*() is a function to generate number in the range (0, 1).

2. The equations of the wormhole Eq. 9 are changed:

$$x_i^j = \begin{cases} \begin{cases} \lfloor x_j + TDR \times ((ub_j - lb_j) \times r_4 + lb_j) Bigr \rfloor & r_3 < 0.5, \\ \lfloor x_j - TDR \times ((ub_j - lb_j) \times r_4 + lb_j) Bigr \rfloor & r_3 \geq 0.5 \end{cases} & r_2 < WEP \\ x_i^j & r_2 \geq WEP \end{cases}, \tag{13}$$

Algorithm 3 Enhancing the quality of initial candidate solutions algorithm

Input: Population matrix U of size n solutions
Output: Improving Population matrix U of size n solutions by K-means
for *Each solution x_i in Population matrix U* **do**
 Step1: calculate the solution x_i quality using Eq.5
 Step2: make the solution x_i instead of the first step in Algorithm 2.
 Step3: run Algorithm 2
 Step4: calculate the solution x_i quality using Eq.5

 if *The solution x_i improved* **then**
  Step5a: replace the solution x_i with the old solution.
 end
 else
  Step5b: the solution x_i remains unchanged.
 end
end

where x_j represents the j_{th} index of the best solution, which has been, so far, created, the rate of the traveling distance (*TDR*), and the wormhole existence probability (*WEP*) represent coefficient parameters, $lb_j = 1$, ub_j signifies the number of clusters, $r2$, $r3$, $r4$ signify random numbers between (0, 1), and *Bigr* denotes a function that rounds the value of x_i^j to the nearest integers that are less than or equal to x_i^j.

A set of solutions is randomly generated in the initialization phase utilizing Eq. 7, and the solutions fitness is computed utilizing Eq. 5. The k-means clustering algorithm considers the solutions, in this phase, as input for enhancing each of the solutions and recalculating the quality of the solution using Eq. 5. If the generated solution quality outperforms the old solution quality, it memorizes the generated solution; otherwise, it continues with the old solution. The process will be repeated for all solutions in the population (see, Algorithm 3).

5.3 Enhancing the best solution of MVO using K-means (H-MVO2)

Like other metaheuristic algorithms, the MVO algorithm is designed so that two phases are accomplished, including the exploration phase and the exploitation phase, in which, the algorithm should be armed with mechanisms for an extensive search of the search space. In exploration phase, the search space's promising regions are identified. The exploitation phase, on the other hand, emphasizes the local search, as well as convergence toward the promising areas that are obtained in the exploration phase. The key

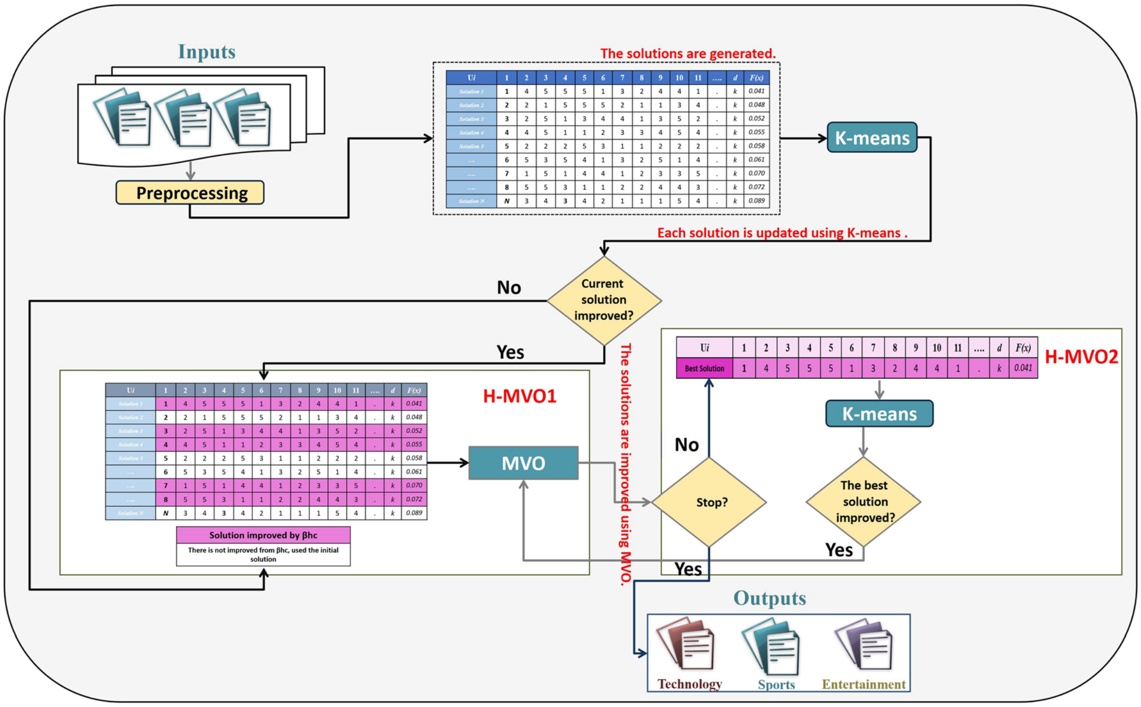


Fig. 6 Proposed methodology

solution, which is adequate in the two phases, represents the best solution. The possibility of solutions to learn from the best solution is very high in the exploration phase. Besides, it only focuses on the best solution in the exploitation phase. The liberal bias to improve the best solution might be sharing good attributes in the path of searching for global optima and leads the search to find better solutions. At each iteration, the best solution, which is produced by H-MVO1, is regarded as the k-means initial state. This stage increases the existence probability of the useful attributes in the best solution and, therefore, the best solution sends these to other solutions and assists them to improve their fitness value. The process of enhancing the best solution is detailed in Algorithm 4). The proposed method's optimization framework is shown in Fig. 6 and pseudocode in Algorithm 5.

Algorithm 4 Enhance the best solution of MVO

Input: The best solution $best_{SI}$
Output: Improving solution $best_{SI}$ by K-means
Step1: make the solution $best_{SI}$ instead of the first step in Algorithm 2.
Step2: run Algorithm 2
Step3: calculate the solution $best_{SI}$ quality using Eq.5
if *The solution* $best_{SI}$ *improved* **then**
| Step4a: replace the solution $best_{SI}$ with the old solution.
end
else
| Step4b: the solution $best_{SI}$ remains unchanged.
end

Algorithm 5 Pseudocodes and general steps of H-MVO algorithm

```
Initialize MVO parameters (Min, Max, Best solution = 0 , LB, UB, Number of solutions, Number of dimensions
(Dim) , Number of iterations ).
Create MVO population.
run Algorithm 3
while the end criterion is not satisfied do
    Evaluate the objective function for all solutions.
    Sort_s=Sorted Solutions based on objective value from the best to worst.
    Normalize=Normalize the objective function for all solutions.
    run Algorithm 4
    Update the Best solution vector.
    for each solutions indexed by i except the best solution do
        Calculate WEP and TDR using Equations ( 10 and 11 ).
        Black_hole_index=i.
        for each object indexed by j do
            r1 = random([0, 1]).
            if r1 < Normalize(S_i) then
                White_hole_index = Roulette_Wheel_Selection .
                S(Black_hole_index,j)= Sort_s(White_hole_index,j).
            end if
            r2 = random([0, 1]).
            if r2 < Wormhole_existance_probability then
                r3 = random([0, 1]).
                r4 = random([0, 1]).
                if r3 < 0.5 then
                    Update the Position of Solution using Equation 13 case one.
                else
                    Update the Position of Solution using Equation 13 case two.
                end if
            end if

        end for
    end for
end while
Produce the best Solution.
```

It is worth noting that the proposed method can be utilized to solve other formulated optimization problems such as differential equations [75, 76], global optimization [77], time-fractional Schrodinger equations [78], second-order, two-point fuzzy boundary value problems [79], and fuzzy Fredholm–Volterra integrodifferential equations [75].

6 Results and discussion

The experimental design is given in this section, parameter setting and experimental benchmark datasets, state-of-the-art algorithms, as well as the evaluation measures of the proposed method. Statistical analysis also provides, besides the complexity of the HMVO, convergence rate and an overview of the findings. The Linux environment is used to perform the experiments. The program MATLAB was used in this paper with various public datasets that are described in the following section.

6.1 The experimental design

The evaluation, as well as the comparison of the proposed H-MVO method, along with other state-of-the-art techniques, is provided in this section. For providing better insights, three types of artificial data are used to test the proposed algorithm output, including text clustering datasets (TCD), data clustering datasets (DCD), and scientific articles datasets (SAD). Thirty times the non-metaheuristic and metaheuristic algorithms were tested using the same initial solutions with the entire datasets. The number of runs was selected for all the competing algorithms based on the literature to conduct a fair comparison. The non-metaheuristic methods for the clustering technique run 100 iterations every run time, 100 iterations can be experimentally suitable for the intensification search algorithm's convergence, and 1000 iterations are suitable for the diversification search algorithm's convergence for the population-based algorithms. As conventionally performed, seven external evaluation criteria are applied. These measures include error rate, purity, entropy, recall, precision, F-measure, and accuracy criteria, and the sum of the intra-cluster distances represents an internal quality measure. To conduct a comparative evaluation, the findings obtained through the evaluation measures have been compared against the results, which were obtained by using twelve state-of-the-art algorithms: K-mean, K-mean++, DBSCAN, Agglomerative, Spectral, KHA, HS, PSO, GA, H-PSO, H-GA, and MVO.

6.2 Experimental benchmark datasets and parameter setting

To demonstrate the proposed H-MVO algorithm's efficacy, three types of artificial datasets were chosen to test the problem of clustering. Five DCD, two SAD, and six TCD were selected for the comparison from the literature as testbeds. Such datasets reflect the clustering benchmark datasets. They are commonly implemented to examine the performance of the newly developed algorithms. The Machine Learning Repository (UCI) of the University of California provided the DCD, namely (CMC, Iris, Seeds, Glass, and Wine).

The TCD can be downloaded from the Laboratory of Computational Intelligence (LABIC) in a numerical form after term extraction. The attributes of the TCD are discussed below:

CSTR [1]The Centre for Speech Technology Research (CSTR) is a research center interdisciplinary, in which informatics, linguistics, and the English language are linked. The CSTR was established in 1984. It is concerned with research in several areas such as information access and speech recognition. This dataset contains 299 documents, which belong to four classes: theory, artificial intelligence, robotics, and systems.

20Newsgroups [2]The 20Newsgroups dataset contains 19,997 articles belonging to 20 classes that are collected from different Usenet newsgroups. The first 100 documents are selected from the top three classes in this dataset for our experiment: comp_windows_x, talk_politics_misc, and rec_autos classes.

Tr41, Tr12, and Wap [3]These datasets, including tr41,tr12, and Wap from Karypis lab, contain 878, 313, and 1560 documents that belong to 10, 8, and 20 classes, respectively. They can be downloaded from [5]

Classic4 [4]The Classic4 dataset contains 2000 documents, which belong to : (1) CISI, (2) CACM, (3) MED, and (4) CRAN (each classes has 500 documents) classes. Previously, it contains more than 7000 documents, which belong to the same number of classes.

The SAD involves scientific articles, which are published in international conferences, namely (NIPS 2015 and AAAI 2013). The features of the SAD are discussed as follows:

NIPS 2015 The dataset has been taken from kaggle site.[5] It contains 403 articles that were published in the Neural Information Processing Systems (NIPS) conference, which is an important core ranked conference in the machine learning domain. It has topics that range from deep learning, computer vision to cognitive science, and reinforcement learning. This dataset includes paper id, the title of the paper, event type (poster/oral/spotlight presentation), name of the pdf file, abstract, and paper text, out of which only, title, abstract, and paper text that are used during our experimentation. Here, most of the articles are related to machine learning and natural language processing.

AAAI 2013 The dataset has been taken from UCI repository[6] that comprises 150 articles, which are accepted from a different core ranked conference of AI domain, i.e., AAAI 2013. Each paper has information, including the paper title, the topics (author-selected low-level keywords from the conference-provided list), the keywords (author-generated keywords), the abstract, and the high-level keywords (author selected high-level keywords from the conference-provided list). Many articles are associated with artificial intelligence such as the multi-agent system, reasoning, and machine learning such as data mining and knowledge discovery.

Table 1 illustrates related information, which is given in each of the datasets. It includes the name and the number of the dataset, the number of clusters, the documents or objects number, and the features number. It is essentially important to identify the parameter values of the introduced H-MVO, as well as further comparative algorithms.

[1] http://archive.ics.uci.edu/ml/index.php/.

[2] https://www.kaggle.com/ammarabbasi/20newsgroups-300-articles.

[3] http://archive.ics.uci.edu/ml/index.php/.

[4] http://archive.ics.uci.edu/ml/index.php/.

[5] https://www.kaggle.com/benhamner/exploring-the-nips-2015-papers/data.

[6] https://www.archive.ics.uci.edu/ml.

Table 1 Description of the experimental datasets

Type	ID	Datasets	Number of objects/documents	Number of clusters (K)	Number of features (t)
DCD					
	DS1	CMC	1473	3	9
	DS2	Iris	150	3	4
	DS3	Seeds	210	3	7
	DS4	Glass	214	7	9
	DS5	Wine	178	3	13
TCD					
	DS6	CSTR	299	4	1725
	DS7	20Newsgroups	300	3	2275
	DS8	tr12	313	8	5329
	DS9	tr41	878	10	6743
	DS10	Wap	1560	20	7512
	DS11	Classic4	2000	4	6500
SAD					
	DS12	NIPS 2015	403	2	22,888
	DS13	AAAI 2013	150	4	1897

Table 2 Parameter setting for comparing all algorithms

Algorithm	Parameters	Value
All optimization algorithms	Population size	20
All optimization algorithms	Maximum number of iteration	1000
All Optimization algorithms	runs	30
proposed method (H-MVO)	WEP Max	1.00
proposed method (H-MVO)	WEP Min	0.20
proposed method (H-MVO)	p	6.00
GA	Crossover probability	0.80
GA	Mutation probability	0.02
PSO	Maximum inertia weight	0.90
PSO	Minimum inertia weight	0.20
PSO	C1	2.00
PSO	C2	2.00
KHA	*V f*	0.02
KHA	*D max*	0.002
KHA	*N Max*	0.05
HS	*PAR Min*	0.45
HS	*PAR Max*	0.90
HS	*bw Min*	0.10
HS	*bw Max*	1.00
HS	HMCR	0.90

Table 2 illustrates the comparison of the entire algorithms' algorithmic parameters.

6.3 Evaluation measures

Seven different evaluation measures have been conventionally used for evaluating the solutions of TDC. The measures included (i) accuracy (ii) error rate,(iii) entropy, (iv) recall, (v) precision, (vi) F-measure, (vii) purity, and [80]. The calculation details of these measures are:

6.3.1 Error rate (ER)

The external quality measure is characterized as the ER. The ER in TDC is used to measure the incorrect documents percentage to overall number of documents as shown in Eq. 14.

$$ER = \frac{number\ of\ misplaced\ objects\ or\ text\ documents}{size\ of\ test\ dataset} \times 100, \tag{14}$$

6.3.2 Accuracy

This measure is used to compute the percentage of documents assigned to the right clusters [81, 82] as shown in Eq. (15).

$$Ac = \frac{1}{n}\sum_{j=1}^{k} n_{ij}, \tag{15}$$

where n signifies the all documents, n_{ij} signifies the number of assigned documents correctly in cluster j of class, and i, k represents the number of clusters.

6.3.3 Precision

The ratio between the total documents in the cluster and correct text documents is provided by the precision measure [83].

$$P(i,j) = \frac{n_{i,j}}{n_j}, \tag{16}$$

where $n_{i,j}$ signifies the correctly assigned number of objects or the text documents of class i in cluster j and n_j signifies all cluster j documents.

6.3.4 Recall

The ratio between the total documents in the class and correct text documents is provided by the recall measure. The precision for class i in cluster j is calculated by Eq. (17).

$$R(i,j) = \frac{n_{i,j}}{n_i}, \tag{17}$$

where $n_{i,j}$ is the correctly assigned number of objects or the text documents of class i in cluster j and n_i is all documents in class i.

6.3.5 F-measure

The F-measure shows a harmonic combination between precision and recall. The best F-measure value is close to 1 [84]. Equation (18) is utilized to calculate the F-measure.

$$F(i,j) = \frac{2 \times P(i,j) \times R(i,j)}{P(i,j) + R(i,j)}, \tag{18}$$

where $P(i, j)$ is the precision of class i in cluster j and $R(i, j)$ is the recall of class i in cluster j. Calculating the F-measure of all the clusters is presented in Eq. (19).

$$F = \sum_{i=1}^{k} \frac{n_j}{n} \max F(i,j), \tag{19}$$

where $\max F(i,j)$ is the max value for one class over all clusters.

6.3.6 Purity measure

It is used to compute each cluster's percentage in the large class by assigning each of the clusters to the most frequent class [85]. It is the percent of the total number of documents that were classified correctly, in the unit range (0, 1). The best purity value is close to 1, given that the large-size class rate in each of the clusters is computed for concurrence with an estimated size of the cluster. Equation (20) calculates this measure for all the clusters.

$$purity = \frac{1}{n} \sum_{i=1}^{k} \max(i,j), \tag{20}$$

where $\max(i,j)$ represents the large class i size in cluster j, k signifies the number of clusters, and n signifies the entire number of documents in the dataset.

6.3.7 Entropy measure

This measure analyzes the documents distributed in clusters for a single class. The best entropy value is close to 0. The entropy evaluates for each class the documents distributed to the correct clusters. The entropy of the cluster is calculated by two steps: (1) The distribution of documents for clusters is determined for each class, and (2) all entropies by step (1) are utilized to measure cluster entropy. Calculating the cluster j entropy is presented in Eq. (21).

$$E(kj) = -\sum_{i=1} p(i,j) logp(i,j), \tag{21}$$

where $E(kj)$ signifies the entropy of cluster j and $p(i, j)$ represents the probability of the text documents or objects in cluster i belonging to class j. For all clusters, entropy is calculated via Eq. (22).

$$Entropy = -\sum_{i=1}^{K} \frac{n_i}{n} E(kj), \tag{22}$$

where K signifies the number of clusters, n_i represents all documents in cluster i, and n represents all the datasets documents.

6.4 Analysis of the results

For the sake of measuring the clusters, which were obtained by the proposed H-MVO algorithm and others, seven evaluation measures, namely ER, accuracy, precision, recall, F-measure, purity, and entropy, were used. Tables 3, 4, and 5 report the results of five DCD, two SAD, and six TCD, respectively. As illustrated in the tables, the hybridized versions of H-MVO1 and H-MVO2 highly competitive ER results were obtained. In addition, it outperformed other algorithms for both metaheuristic and hybrid metaheuristic methods. These improvements were achieved because H-MVO enhanced the quality of the initial candidate solutions of the MVO by enhancing the start point of these solutions and improving the best solution of the population by utilizing the local search strategy

Table 3 Results of accuracy, entropy, purity, ER, precision, recall, and F-measure for five DCD

Dataset	Measure	Non-metaheuristic methods					Metaheuristic methods					Hybrid metaheuristic methods			
		K-mean	K-mean++	DBSCAN	Agglomerative	Spectral	KHA	HS	PSO	GA	MVO	H-PSO	H-GA	H-MVO1	H-MVO2
DS1	ER	0.5547	0.5626	0.5654	0.5494	0.5512	0.5606	0.56	0.559	0.567	0.5307	0.542	0.5514	0.5267	**0.5067**
	Accuracy	0.6412	0.5937	0.6167	0.6464	0.5956	0.6244	0.5904	0.6190	0.5990	0.6500	0.6774	0.6064	0.6730	**0.6960**
	Precision	0.7281	0.6596	0.6266	0.6934	0.6226	0.6984	0.6193	0.6709	0.6599	0.6929	0.6514	0.6673	0.7289	**0.7359**
	Recall	0.6593	0.6178	0.5848	0.6855	0.5797	**0.6905**	0.5865	0.6381	0.6411	0.6750	0.6315	0.6025	0.6601	0.6801
	F-measure	0.6789	0.6213	0.5893	0.6739	0.5909	**0.6847**	0.5864	0.6440	0.6314	0.6678	0.6687	0.6674	0.6590	0.6624
	Purity	0.7004	0.6454	0.6124	0.7017	0.6054	0.7049	0.6138	0.6631	0.6631	0.6917	0.6529	0.6228	0.7081	**0.7111**
	Entropy	0.5087	0.504	0.4723	0.517	0.5029	0.5119	0.5276	0.4237	0.5094	0.4553	0.4549	0.4216	0.3997	**0.3814**
	p value	–	–	–	–	–	–	–	–	–	–	–	–	< 0.0001	1
	Rank	4	11	12	5	13	6	14	8	10	3	7	9	2	1
DS2	ER	0.2147	0.2098	0.1631	0.1854	0.1746	0.2266	0.2105	0.1587	0.2165	0.2344	**0.158**	0.211	0.1854	0.1644
	Accuracy	0.6738	0.6677	0.6866	0.7242	0.6768	0.6993	0.6908	0.6722	0.7190	0.7204	0.7203	0.7278	0.7332	**0.7420**
	Precision	0.6940	0.6919	0.7055	0.7703	0.7353	0.7747	0.7481	0.7343	0.8056	0.7624	0.7417	0.7741	0.7923	**0.8086**
	Recall	0.6501	0.6639	0.6584	0.7558	0.7140	0.7676	0.7154	0.6921	0.7372	0.7456	0.7476	0.7624	0.7771	**0.7972**
	F-measure	0.6613	0.6562	0.6659	0.7476	0.7094	0.7549	0.7215	0.6946	0.7530	0.7447	0.7629	**0.7965**	0.7856	0.7605
	Purity	0.6813	0.6835	0.6854	0.7768	0.7410	0.7772	0.7395	0.7153	0.7723	0.7681	0.7442	0.7065	0.7643	**0.7813**
	Entropy	0.4188	0.4163	0.3918	0.5573	0.5284	0.5467	0.528	0.4628	0.4778	0.4916	0.5337	0.463	0.3738	**0.3578**
	p value	–	–	–	–	–	–	–	–	–	–	–	–	< 0.0001	1.0000
	Rank	14	13	12	5	11	8	10	9	4	7	6	3	2	1
DS3	ER	0.1264	0.112	0.1221	0.1302	0.113	0.136	0.1302	0.1526	0.21	0.1485	0.1388	0.1346	0.1165	**0.1055**
	Accuracy	0.7247	0.7020	0.6026	0.7779	0.6975	0.7328	0.6298	0.7231	0.7021	0.7577	0.7858	0.7588	0.7771	**0.7961**
	Precision	0.7290	0.7604	0.6973	0.8146	0.6786	0.7976	0.7196	0.7721	0.7658	0.7921	0.7996	0.7816	0.8131	**0.8388**
	Recall	0.6922	0.7461	0.6927	**0.8267**	0.6860	0.7958	0.6876	0.7420	0.7180	0.7783	0.7518	0.7516	0.8030	0.8100
	F-measure	0.6884	0.7373	0.6738	0.8145	0.6557	0.7842	0.6881	0.7501	0.7215	0.7719	0.7722	0.7713	0.6481	**0.8235**
	Purity	0.7152	0.7682	0.7013	0.8339	0.7138	0.8101	0.7171	0.7697	0.7521	0.7916	0.8001	0.7081	0.7787	**0.8506**
	Entropy	0.4680	0.4818	0.4474	0.4119	0.5053	0.4757	0.4638	0.4148	0.4328	0.5326	0.3697	0.4028	0.3648	**0.3548**
	p value	–	–	–	< 0.00001	–	–	–	–	–	–	–	–	–	1
	Rank	11	9	14	2	13	5	12	8	10	7	3	6	4	1
DS4	ER	0.4615	0.4457	0.4498	0.4322	0.4661	0.4393	0.4205	0.4626	0.5103	0.419	0.4762	0.4422	0.363	**0.329**
	Accuracy	0.5191	0.5424	0.5499	0.5826	0.4966	0.5230	0.5056	0.5066	0.4964	0.5345	0.5490	0.5446	0.5706	**0.5914**
	Precision	0.5366	0.6310	0.6028	0.6108	0.5620	0.5812	0.5296	0.5689	0.5220	0.6024	**0.6032**	0.6716	0.6489	**0.6720**
	Recall	0.4901	0.5652	0.5938	0.6341	0.5231	0.5438	0.4820	0.5432	0.4970	0.5922	0.6228	0.6340	0.6452	**0.6790**
	F-measure	0.4912	0.5859	0.5827	**0.6149**	0.5302	0.5472	0.4967	0.5392	0.4914	0.5865	0.5902	0.5337	0.5932	0.6104
	Purity	0.5179	0.6072	0.6105	0.6392	0.5554	0.5664	0.5122	0.5659	0.5198	0.6090	0.6214	0.6562	0.6109	**0.6758**
	Entropy	0.4189	0.5202	0.5036	0.4592	0.4114	0.4704	0.3782	0.4389	0.4148	0.5661	0.4364	0.4262	0.3659	**0.3428**

Table 3 continued

Dataset	Measure	Non-metaheuristic methods					Metaheuristic methods					Hybrid metaheuristic methods			
		K-mean	K-mean++	DBSCAN	Agglomerative	Spectral	KHA	HS	PSO	GA	MVO	H-PSO	H-GA	H-MVO1	H-MVO2
	p value	–	–	–	–	–	–	–	–	–	–	–	–	< 0.001	1.0000
	Rank	13	7	6	3	11	9	12	10	14	8	5	4	2	1
DS5	ER	0.3239	0.3184	0.3349	0.3415	0.3359	0.323	0.3257	0.3205	0.3427	0.3371	0.3087	0.3099	0.3021	**0.2741**
	Accuracy	0.4755	0.5016	0.5107	0.4877	0.4496	0.4782	0.4645	0.4638	0.4481	0.5345	0.5672	0.5055	0.5458	**0.5921**
	Precision	0.4899	0.5912	0.5570	0.5633	0.5169	0.5415	0.4829	0.5245	0.4799	0.5665	0.5415	0.5629	0.5895	**0.6309**
	Recall	0.4463	0.5194	0.5452	0.5504	0.4744	0.5014	0.4355	0.4957	0.4550	**0.5897**	0.5734	0.5125	0.5497	0.5810
	F-measure	0.4512	0.5445	0.5388	0.5465	0.4823	0.5015	0.4573	0.4925	0.4444	0.5662	0.5545	0.5393	0.5695	**0.6074**
	Purity	0.4786	0.5667	0.5681	0.5644	0.5122	0.5267	0.4719	0.5230	0.4752	0.5998	0.5667	0.5679	0.5830	**0.6242**
	Entropy	0.5186	0.6067	0.6071	0.5664	0.5222	0.5877	0.5109	0.5970	0.5032	0.6188	0.5607	0.5269	0.5040	**0.4912**
	p value	–	–	–	–	–	–	–	–	–	–	–	–	< 0.001	1.0000
	Rank	12	7	8	6	11	9	13	10	14	4	3	5	2	1
Average		10.8	9.4	10.4	4.2	11.8	7.4	12.2	9	10.4	5.8	4.8	5.4	2.4	1
Final rank		12	9	10	3	13	7	14	8	10	6	4	5	2	1

Table 4 Results of accuracy, entropy, purity, ER, precision, recall, and F-measure for six TCD

Dataset	Measure	Non-metaheuristic methods					Metaheuristic methods					Hybrid metaheuristic methods			
		K-mean	K-mean++	DBSCAN	Agglomerative	Spectral	KHA	HS	PSO	GA	MVO	H-PSO	H-GA	H-MVO1	H-MVO2
DS6	ER	0.4029	0.3678	0.4244	0.3672	0.3591	0.3328	0.3431	0.3374	0.3948	0.3198	0.3028	0.3078	0.2678	**0.2358**
	Accuracy	0.3573	0.4355	0.4005	0.4360	0.4319	0.3649	0.4464	0.4356	0.3399	0.4593	0.5494	0.5056	0.5683	**0.5779**
	Precision	0.4091	0.3953	0.3393	0.4423	0.3597	0.4213	0.4235	0.5340	0.4417	0.5715	0.6065	0.6130	0.6395	**0.6497**
	Recall	0.3092	0.4076	0.4256	0.4666	0.4925	0.5355	0.5060	0.4360	0.3418	0.4829	**0.5600**	0.4950	0.5079	0.5568
	F-measure	0.3460	0.3546	0.3046	0.3266	0.3971	0.4139	0.3377	0.4819	0.3886	0.5244	0.5577	0.5859	0.5774	**0.5936**
	Purity	0.3525	0.4096	0.4076	0.4816	0.4485	0.3874	0.4355	0.4953	0.4050	0.5685	**0.6135**	0.5743	0.5015	0.5980
	Entropy	0.8201	0.5246	0.4586	0.5076	0.4893	0.4344	0.4786	0.6199	0.7170	0.5207	0.5076	0.5104	0.4577	**0.4010**
	p value										–	< 0.001	–	–	1
	Rank	14	11	12	10	9	7	8	6	13	5	2	4	3	1
DS7	ER	0.3205	0.3257	0.384	0.3313	0.3143	0.2962	0.3021	0.2967	0.3605	0.2804	0.2744	0.2804	0.2864	**0.2324**
	Accuracy	0.3180	0.3784	0.3038	0.4055	0.3633	0.3216	0.3122	0.3498	0.3676	0.4044	0.4442	0.4308	0.5174	**0.5326**
	Precision	0.3121	0.3652	0.3094	0.3990	0.3424	0.3829	0.3601	0.4134	0.4209	0.4392	0.4821	0.4814	0.5102	**0.5619**
	Recall	0.3100	0.3662	0.3017	0.3576	0.3280	0.3136	0.3170	0.3497	0.3676	0.3842	0.4220	**0.4887**	0.4332	0.4876
	F-measure	0.3406	0.3619	0.3193	0.3548	0.3136	0.2996	0.3214	0.3803	0.3936	0.4109	0.4454	0.4433	0.5169	**0.5376**
	Purity	0.3741	0.4134	0.3027	0.4417	0.3110	0.3421	0.3355	0.4097	0.4081	0.4344	0.4725	0.4837	**0.5104**	0.4591
	Entropy	0.8028	0.6611	0.7473	0.5953	0.6125	0.6767	0.6481	0.7723	0.7547	0.7121	0.6631	0.6833	0.5811	**0.5407**
	p value	–	–	–	–	–	–	–	–	–	–	–	–	< 0.000001	1.0000
	Rank	13	7	14	6	10	12	11	9	8	5	4	3	2	1
DS8	ER	0.4384	0.4899	0.4271	0.4422	0.4682	0.4136	0.4484	0.4272	0.4767	0.4669	0.4599	0.4549	**0.4019**	0.4279
	Accuracy	0.2971	0.3795	0.3045	0.4481	0.3373	0.3357	0.3776	0.4075	0.3677	0.4485	0.4796	0.5205	0.5465	**0.5617**
	Precision	0.3522	0.4215	0.3218	0.4923	0.4508	0.3748	0.4385	0.4298	0.4128	0.5075	0.5355	0.4798	0.6025	**0.6398**
	Recall	0.2944	0.3778	0.3234	0.4341	0.3055	0.3099	0.3453	0.4264	0.3549	0.4398	0.3893	0.4974	**0.5138**	0.5109
	F-measure	0.3222	0.4176	0.4048	0.4090	0.3781	0.2916	0.4470	0.4278	0.3826	0.4706	0.5860	0.5428	0.5776	**0.5956**
	Purity	0.3908	0.4808	0.4728	0.5553	0.4132	0.3803	0.4986	0.4878	0.4513	0.5448	0.5876	0.6038	0.5948	**0.6063**
	Entropy	0.7138	0.5094	0.5010	0.3978	0.4932	0.5183	0.4517	0.5720	0.6233	0.5224	0.5757	0.6380	0.4224	**0.3753**
	p value	–	–	–	–	–	–	–	–	–	–	–	–	< 0.001	1
	Rank	14	9	11	5	10	13	7	8	12	6	4	3	2	1
DS9	ER	0.4459	0.4816	0.4552	0.4405	0.4556	0.4212	**0.3885**	0.4185	0.5048	0.4749	0.4389	0.4449	0.4029	0.3939
	Accuracy	0.4126	0.4260	0.3704	0.4610	0.3787	0.4054	0.3590	0.4870	0.4320	0.4630	0.4740	0.5940	0.5680	**0.6150**
	Precision	0.3945	0.3769	0.4242	0.3445	0.3630	0.4191	0.3710	0.4505	0.4140	0.4569	0.5010	0.5245	0.5489	**0.5690**
	Recall	0.3813	0.3559	0.4291	0.3487	0.3480	0.4579	0.3888	0.4497	0.4008	0.4419	0.4768	0.5127	0.5079	**0.5178**
	F-measure	0.3876	0.4299	0.4152	0.4187	0.3691	0.4137	0.3701	0.4497	0.4071	0.4569	0.4701	**0.5647**	0.5119	0.5201
	Purity	0.4108	0.5471	0.6025	0.4760	0.4866	0.6063	0.4863	0.5790	0.5603	0.6081	0.5273	0.6420	0.6421	**0.6673**
	Entropy	0.5874	0.4745	0.4776	0.4821	0.5386	0.4041	0.4539	0.5391	0.5469	0.5355	0.5029	0.5371	0.3965	**0.3669**

Table 4 continued

Dataset	Measure	Non-metaheuristic methods					Metaheuristic methods					Hybrid metaheuristic methods			
		K-mean	K-mean++	DBSCAN	Agglomerative	Spectral	KHA	HS	PSO	GA	MVO	H-PSO	H-GA	H-MVO1	H-MVO2
	p value	–	–	–	–	–	–	–	–	–	–	–	–	< 0.001	1
	Rank	13	9	8	12	14	5	11	6	10	7	4	3	2	1
DS10	ER	0.4885	0.5145	0.5372	0.425	0.481	0.4505	0.4725	0.4598	0.5083	0.4392	0.4132	0.4272	0.3922	**0.3762**
	Accuracy	0.5012	0.4937	0.4846	0.4933	0.4905	0.5191	0.5032	0.5623	0.5316	0.5291	0.5532	**0.6523**	0.5931	0.6176
	Precision	0.4626	0.4913	0.5004	0.4479	0.4454	0.4773	0.4630	0.5249	0.5314	0.5213	0.5450	0.5749	0.6063	**0.6344**
	Recall	0.4011	0.3508	0.4176	0.4471	0.3423	0.4106	0.4447	0.4811	0.4706	0.4496	0.4907	0.5671	**0.5906**	0.5855
	F-measure	0.4315	0.4462	0.4878	0.4487	0.4141	0.4131	0.4611	0.5017	0.4998	0.4831	0.5521	**0.6187**	0.5911	0.6105
	Purity	0.4759	0.5797	0.4007	0.5445	0.5142	0.5939	0.5589	0.6125	0.4917	0.6069	0.6939	**0.7315**	0.6849	0.7107
	Entropy	0.7044	0.5961	0.5706	0.4875	0.5468	0.5595	0.5470	0.5765	0.6216	0.6625	0.6310	0.6585	0.4365	**0.4196**
	p value	–	–	–	–	–	–	–	–	–	–	–	–	< 0.001	1
	Rank	14	11	12	7	13	9	8	5	10	6	4	3	2	1
DS11	ER	0.3461	0.3735	0.3951	0.3486	0.3348	0.3031	0.3166	0.315	0.3694	0.2951	0.2861	0.2791	0.2461	**0.2391**
	Accuracy	0.5858	0.6799	0.6206	0.6683	0.6326	0.5761	0.5963	0.6363	0.6621	0.7043	0.6853	**0.7273**	0.7163	0.7271
	Precision	0.5699	0.6880	0.5708	0.6449	0.6620	0.6246	0.6064	0.6604	0.6726	0.6919	0.7354	**0.7774**	0.7129	0.7226
	Recall	0.5259	0.7028	0.5796	0.6224	0.6528	0.5550	0.5674	0.6164	0.6320	0.6844	0.6754	0.6944	0.6844	**0.7050**
	F-measure	0.5472	0.6920	0.6173	0.6281	0.6764	0.5529	0.5787	0.6377	0.6519	0.6881	0.6297	0.6887	0.6951	**0.7079**
	Purity	0.5938	0.6227	0.6089	0.5812	0.6352	0.5550	0.5983	0.6243	0.6320	0.6742	0.6413	0.6733	0.7042	**0.7320**
	Entropy	0.5601	0.3540	0.4854	0.4033	0.4002	0.5321	0.4946	0.5306	0.5781	0.5113	0.5656	0.6166	0.3553	**0.3341**
	p value	–	–	–	–	–	–	–	–	–	–	–	–	< 0.0001	1
	Rank	14	4	12	8	6	13	11	9	10	5	7	3	2	1
Average		13.7	8.5	11.5	8.0	10.3	9.8	9.3	7.2	10.5	5.7	4.2	3.2	2.2	1.0
Final rank		14	8	13	7	11	10	9	6	12	5	4	3	2	1

Table 5 Results of accuracy, entropy, purity, ER, precision, recall, and F-measure for two SAD

Dataset	Measure	Non-metaheuristic methods					Metaheuristic methods					Hybrid metaheuristic methods			
		K-mean	K-mean++	DBSCAN	Agglomerative	Spectral	KHA	HS	PSO	GA	MVO	H-PSO	H-GA	H-MVO1	H-MVO2
DS12	ER	0.477	0.3413	0.3483	0.3672	0.4045	0.3707	0.3507	0.3685	0.3754	0.2824	0.2874	0.2804	0.2644	**0.2514**
	Accuracy	0.5433	0.5545	0.5281	0.5472	0.5593	0.6306	0.6211	0.5802	0.5503	0.6081	0.6556	0.6272	0.6477	**0.6656**
	Precision	0.4489	0.4994	0.5443	0.5340	0.4929	0.5914	0.5413	0.6440	0.6209	0.6113	0.6384	0.6246	0.6593	**0.6724**
	Recall	0.5211	0.3913	0.4706	0.4797	0.3911	0.4986	0.5156	0.5717	0.5571	0.5756	0.6075	0.5901	0.6248	**0.6786**
	F-measure	0.5197	0.4701	0.4251	0.4661	0.4547	0.5298	0.4991	0.6461	0.5797	0.5941	0.6195	0.5965	0.6302	**0.6668**
	Purity	0.5735	0.5262	0.6459	0.5839	0.5685	0.5127	0.6919	0.7089	0.5015	0.6899	**0.7877**	0.7589	0.6457	0.7457
	Entropy	0.5415	0.5218	0.5565	0.5030	0.5555	0.5556	0.5915	0.5190	0.5085	0.5005	0.4536	0.4544	0.4371	**0.4146**
	p value	–	–	–	–	–	–	–	–	–	–	< 0.0001	–	–	1
	Rank	12	13	11	10	14	9	7	6	8	5	2	4	3	1
DS13	ER	0.475	0.501	0.4535	0.5715	0.4978	0.5243	0.5382	0.4692	0.5062	0.4302	0.4242	0.4225	0.4085	**0.3892**
	Accuracy	0.4887	0.4896	0.4892	0.5635	0.4188	0.5811	0.5945	0.6242	0.5395	0.6238	0.6811	0.6685	0.6886	**0.6937**
	Precision	0.6323	0.5459	0.6175	0.5219	0.4865	0.6139	0.6125	0.5635	0.5429	0.6625	**0.7289**	0.6439	0.6782	0.6930
	Recall	0.6304	0.5734	0.5324	0.4415	0.3977	0.5860	0.6447	0.6414	0.5255	0.6087	0.6100	0.6083	0.6384	**0.6632**
	F-measure	0.6295	0.5043	0.5615	0.4983	0.4805	0.6084	0.6172	0.5875	0.5103	0.6395	**0.6554**	0.6342	0.6435	0.6548
	Purity	0.6464	0.5512	0.6127	0.5299	0.4280	0.6502	0.6758	0.6187	0.5629	0.6380	0.6572	0.6456	0.6787	**0.6891**
	Entropy	0.6054	0.5402	0.6267	0.5579	0.6190	0.5632	0.5268	0.5917	0.5859	0.5780	0.5532	0.5016	0.4957	**0.4931**
	p value	–	–	–	–	–	–	–	–	–	–	–	–	< 0.0001	1
	Rank	9	11	10	13	14	8	6	7	12	5	3	4	2	1
Average		10.5	12	10.5	11.5	14	8.5	6.5	6.5	10	5	2.5	4	2.5	1
Final rank		10	13	10	12	14	8	6	6	9	5	2	4	2	1

to increase good attributes sharing between the best solution and others. Please note that the average of 30 runs are recorded in this paper.

As for DS1, the H-MVO2 obtained the best ER, accuracy, precision, purity, and entropy compared to other existing algorithms. Regarding DS2, H-MVO2 achieved the best accuracy, precision, recall, purity, and entropy. However, H-PSO and H-GA obtained the best ER and F-measure, respectively. The DS3, H-MVO2 achieved the best values, while agglomerative achieved the best value over statistic measure Recall. The DS4, H-MVO2 achieved the best values. However, the best value over F-measure by agglomerative clustering was achieved. Finally, the DS5, H-MVO2 obtained the best values over statistic measures. Therefore, the proposed method demonstrated the ability to solve the clustering problem effectively compared to the existing clustering techniques, as well as the existing optimization algorithms. The obtained results are reported in Table 3.

The H-MVO2 performance based on the clusters quality utilizing six benchmark standard text datasets is shown in Table 4. H-MVO2 has efficiently performed and, therefore, exceeded other popular algorithms. Moreover, it has scored better performance according to the external measurements in all the datasets (i.e., DS6, DS7, DS8, DS9, DS10, and DS11) and the H-MVO1 in five datasets (i.e., DS7, DS8, DS9, DS10, and DS11). Then, the introduced H-MVO2 achieved a significant improvement compared to the basic MVO. The performance measures was achieved by utilizing the introduced H-MVO2 on all datasets. Also, a comparison was conducted against the performance measures achieved by the comparative algorithms in all datasets.

Two scientific articles datasets (i.e., DS12 and DS13) are reported in Table 5. The H-MVO2 algorithm, which is proposed in this study, performed better compared with the existing algorithms. This proposed H-MVO2 algorithm is the highest algorithm among the datasets, followed by H-MVO1, H-PSO, H-GA, MVO, PSO, HS, KHA, GA, DBSCAN, K-mean, K-mean++, agglomerative, and spectral.

The results of experiments show that the H-MVO2 algorithm in solving the TDC problem has been effective. The evaluation was performed according to the performance of algorithms; it was compared against the state-of-the-art algorithms. The results over six TCD, Five DCD, and two SAD, are provided depending on accuracy, entropy, purity, ER, precision, recall, and F-measure as illustrated in Tables 3, 4 and 5, respectively. The best results are emphasized using a bold font. Based on text clustering evaluation measures, improving the start point of the initial solutions and the best solution at each iteration compared to the basic MVO algorithm and comparative algorithms, it was effective. It is also possible to observe that the proposed method and other algorithms were substantially different.

The MVO algorithm complexity is based on the number of iterations number, the number of universes, and the universe arrangement mechanism of the roulette wheel method. The universe sorting is carried out in each iteration. *Quicksort* is used in the algorithm that possesses the complexity of $O(n\,logn)$ and $O(n^2)$ in the best and worst case scenario, respectively. The selection of the roulette wheel can be run for every single variable in every single universe over the iterations, and it is of $O(n)$ or $O(logn)$ according to the implementation. Therefore, the total complexity is as follows:

$$O(MVO) = O(l(O(Quicksort) + n \times d \times (O(roulettewheel)))) \tag{23}$$

$$O(MVO) = O(l(n^2 + n \times d \times logn)) \tag{24}$$

The complexity of the hybrid strategy of the MVO algorithm (H-MVO2) as defined below:

The K-means algorithm takes $O(t \times k \times m \times d)$ time [86]. Here, t is the number of iterations, k is the number of clusters, $m * d$-dimensional points. If there are n solutions, then for each solution, the *obj* objective functions should be calculated. The total complexity for initializing the population (including objective function calculation) is $O(n(t \times k \times n \times d) + M))$ to enhance the initial candidate solutions and $O(L(t \times k \times n \times d) + M))$ to enhance the best solution over the course of iterations L (i.e., maximum number of iterations). Therefore, the total run time complexity is as follows:

$$\begin{aligned} O(HMVO2) = O((n(t \times k \times n \times d) + M) \\ + (l(n^2 + n \times d \times logn)) \\ + (L(t \times k \times n \times d) + M)) \end{aligned} \tag{25}$$

To investigate the statistical significance of the H-MVO2, its average clustering accuracy, precision, recall, F-measure, purity, and entropy have been compared against the average clustering accuracy, precision, recall, F-measure, purity, and entropy of the competitive algorithms. At 5 % (0.05) significance level, the statistical test nonparametric Mann-Whitney-Wilcoxon test [87] was utilized to carry out the analysis. A hypothesis can be assumed, in this case, which stipulates that the entire tested algorithms share equal accuracy, precision, recall, F-measure, purity, and entropy. An alternative hypothesis stipulates that the entire algorithms cannot share equal accuracy, entropy, purity, ER, precision, recall, and F-measure. Tables 3, 4 and 5 illustrate the obtained p-values for DTC, TCD, and SAD.

6.5 Convergence analysis

The behavior of convergence illustrates how effective the various H-MVO versions (i.e., H-MVO1 and H-MVO2) compared to the existing methods of the state of the art. The clustering algorithms convergence rate signifies a criterion of an evaluation with the aim of finding an optimal solution. Figure 7 shows the average convergence behaviors of HS, GA, PSO, KHA, MVO, H-PSO, H-GA, H-MVO1, and H-MVO2 on DTC, TCD, and SAD. Thirty runs were carried out for each of the datasets. The average value was calculated depending on the behavior of convergence of each of the algorithms. The average distance of the documents to the cluster centroid (ADDC) values was plotted compared to 1000 iterations on the thirteen datasets. The state-of-the-art methods of convergence were faster compared with H-MVO1 and H-MVO2. However, The H-MVO2 result was more effective than the original MVO algorithm. A better quality of clustering was generated compared with the other algorithms.

Although the basic MVO approached the local optima slowly in comparison with the hybrid versions, that is, the H-MVO1 and the H-MVO2, it converged to the optimal solution more efficiently. The H-MVO2, on the other hand, achieved the best solution in comparison with comparative algorithms. It also achieved high-quality clusters compared to various comparative algorithms. Based on examining the proposed versions' behavior of convergence, the introduced H-MVO2 achieved the best performance in addition to achieving the optimal results quickly. Its convergence, however, is quicker than that of the MVO algorithm's versions, as well as the familiar clustering algorithms. The proposed method provides the minimum average distance (i.e., intra-cluster) for all datasets.

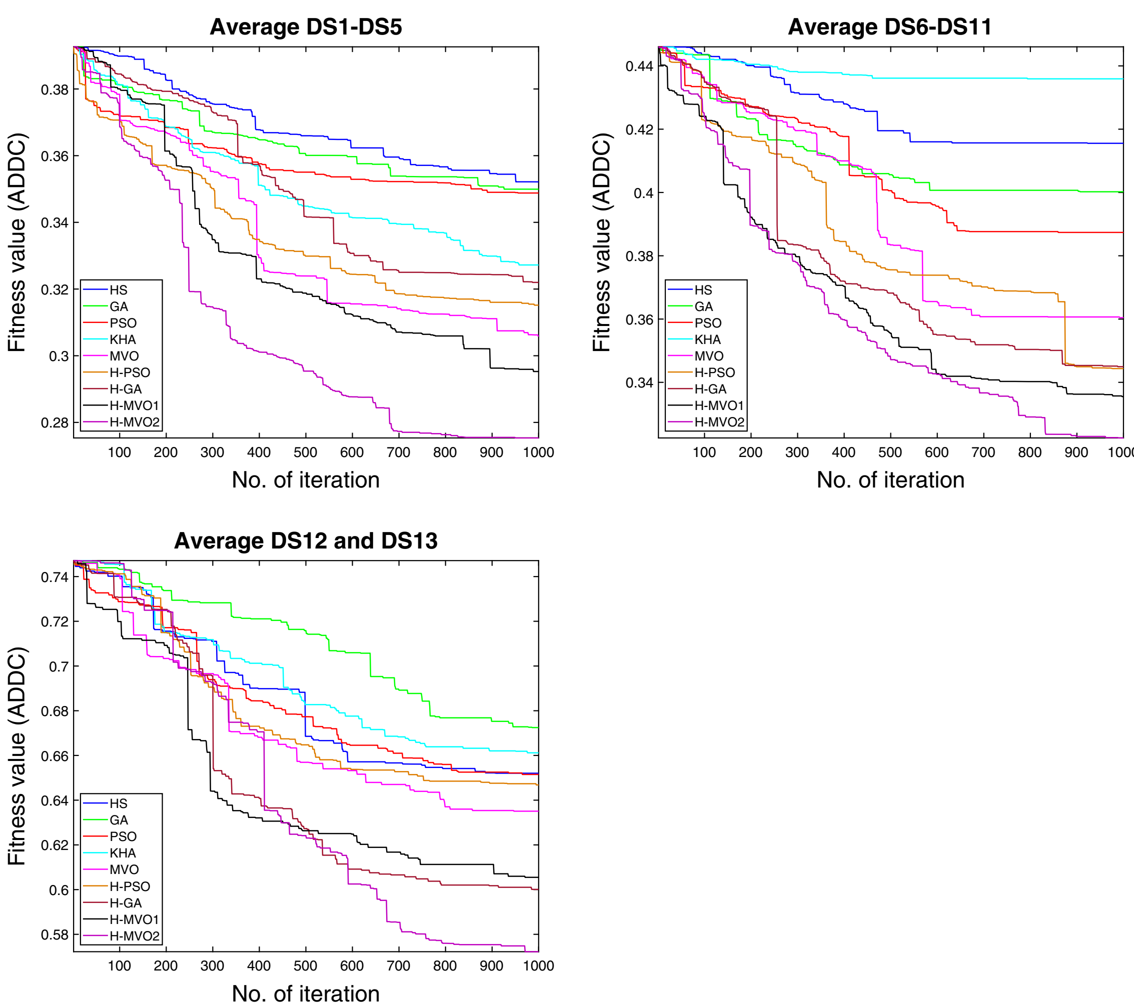


Fig. 7 Average convergence characteristics of optimization algorithms for DTC, TCD, and SAD

7 Conclusion and future works

The problem of Text clustering is a significantly serious problem and, therefore, it has attracted many researchers' attention. The population-based MVO algorithm represents a new optimization algorithm, which aims to solve many serious problems of global optimization. The MVO algorithm aims at simultaneously providing a good exploration of various search space regions of for locating the optimal solution at the exploitation cost. This work was conducted with the aim of tackling two critically fundamental issues that are related to MVO, including the objective function's initial value for candidate solutions, as well as the best solution that can be produced by MVO at each iteration.

A novel hybrid algorithm was developed in this paper to solve the problem of Text clustering according to a combination of k-means clustering with the MVO algorithms with the aim of solving critical issues. An improved version of MVO was invented with the aim of enhancing the initial candidate solutions, in addition to enhancing the best solution. Using this hybridization, the H-MVO searched more efficiently and effectively, and it quickly converged to the optimal solutions.

For the evaluation of the new H-MVO, seven measures of the evaluation were implemented, including accuracy, entropy, purity, ER, precision, recall, and F-measure in addition to the convergence behavior, as well as the statistical analysis. The implemented measures represent the most popularly used evaluation criteria in the domain of data and text mining with the aim of evaluating the new clustering method. The best-recorded results can be produced by the new H-MVO for all the used benchmark datasets compared to the existing versions with several successfully implemented clustering methods, as well as techniques based on the literature. The MVO algorithm with k-means is, therefore, an effective approach for the clustering techniques. Accordingly, many upcoming stories of success are potentially expected in the data, as well as text clustering domain. The results of the study revealed that the introduced hybridization is active; it is an efficient method to tackle the clustering problems. Also, the experimental results were compared against the existing comparative algorithms. Consequently, it was found that the new MVO (H-MVO) hybridization appropriately solved the problems of clustering in relation to the data, as well as the text. Therefore, H-MVO appropriately contributes to the clustering domain. In addition, various clustering problems can be investigated in further studies to confirm the proposed algorithm's capability in the domain. Also, an additional powerful local search can be, accordingly, hybridized with the aim of achieving further enhancements of the MVO's capability of exploitation. Finally, the proposed algorithm in this paper can be investigated in future works on the benchmark function datasets.

Acknowledgements This work was supported by Universiti Sains Malaysia (USM) under Grant (1001/PKOMP/ 8014016).

Compliance with ethical standards

Conflict of interest The authors state that there are no conflicts of interest.

References

1. Figueiredo E, Macedo M, Siqueira HV, Santana CJ Jr, Gokhale A, Bastos-Filho CJA (2019) Swarm intelligence for clustering a systematic review with new perspectives on data mining. Eng Appl Artif Intell 82:313–329
2. Bharti KK, Singh PK (2016) Chaotic gradient artificial bee colony for text clustering. Soft Comput 20(3):1113–1126
3. Jensi R, Jiji DGW (2014) A survey on optimization approaches to text document clustering. arXiv preprint arXiv:1401.2229
4. Kalogeratos A, Likas A (2012) Text document clustering using global term context vectors. Knowl Inf Syst 31(3):455–474
5. Abasi AK, Khader AT, Al-Betar MA, Naim S, Makhadmeh SN, Alyasseri ZAA (2020) Link-based multi-verse optimizer for text documents clustering. Appl Soft Comput 87:106002
6. Rani MS, Babu GC (2019) Efficient query clustering technique and context well-informed document clustering. In: Wang J, Reddy G, Prasad V, Reddy V (eds) Soft computing and signal processing. Springer, Singapore, pp 261–271
7. Kumar Y, Sahoo G (2015) A hybrid data clustering approach based on improved cat swarm optimization and k-harmonic mean algorithm. AI Commun 28(4):751–764
8. Büyüksaatçı S, Baray A (2016) A brief review of metaheuristics for document or text clustering. In: Intelligent techniques for data analysis in diverse settings. IGI Global, pp 252–264
9. Niknam T, Amiri B (2010) An efficient hybrid approach based on pso, aco and k-means for cluster analysis. Appl Soft Comput 10(1):183–197
10. Abualigah LM, Khader AT, Hanandeh ES, Gandomi AH (2017) A novel hybridization strategy for krill herd algorithm applied to clustering techniques. Appl Soft Comput 60:423–435
11. Han J, Pei J, Kamber M (2011) Data mining: concepts and techniques. Elsevier, Amsterdam
12. Abasi AK, Khader AT, Al-Betar MA, Naim S, Makhadmeh SN, Alyasseri ZAA (2019) A text feature selection technique based on binary multi-verse optimizer for text clustering. In: 2019 IEEE Jordan international joint conference on electrical engineering and information technology (JEEIT). IEEE, pp 1–6
13. Sahoo G et al (2017) A two-step artificial bee colony algorithm for clustering. Neural Comput Appl 28(3):537–551
14. Alyasseri ZAA, Khadeer AT, Al-Betar MA, Abasi A, Makhadmeh S, Ali NS (2019) The effects of EEG feature extraction using multi-wavelet decomposition for mental tasks classification. In: Proceedings of the international conference on information and communication technology. ACM, pp 139–146
15. Kaveh A, Khayatazad M (2012) A new meta-heuristic method: ray optimization. Comput Struct 112:283–294
16. Forsati R, Mahdavi M, Shamsfard M, Meybodi MR (2013) Efficient stochastic algorithms for document clustering. Inf Sci 220:269–291

17. Mirjalili S, Mirjalili SM, Lewis A (2014) Grey wolf optimizer. Adv Eng Softw 69:46–61
18. Zaw MM, Mon EE (2015) Web document clustering by using PSO-based cuckoo search clustering algorithm. In: Yang XS (ed) Recent advances in swarm intelligence and evolutionary computation. Springer, Cham, pp 263–281
19. Mirjalili S, Gandomi AH, Mirjalili SZ, Saremi S, Faris H, Mirjalili SM (2017) Salp swarm algorithm: a bio-inspired optimizer for engineering design problems. Adv Eng Softw 114:163–191
20. Pan W-T (2012) A new fruit fly optimization algorithm: taking the financial distress model as an example. Knowl Based Syst 26:69–74
21. Mirjalili S (2016) Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems. Neural Comput Appl 27(4):1053–1073
22. Gandomi AH, Alavi AH (2012) Krill herd: a new bio-inspired optimization algorithm. Commun Nonlinear Sci Numer Simul 17(12):4831–4845
23. Degertekin SO, Hayalioglu MS (2013) Sizing truss structures using teaching–learning-based optimization. Comput Struct 119:177–188
24. Mirjalili S (2015) The ant lion optimizer. Adv Eng Softw 83:80–98
25. Kaveh A, Farhoudi N (2013) A new optimization method: dolphin echolocation. Adv Eng Softw 59:53–70
26. Cura T (2012) A particle swarm optimization approach to clustering. Expert Syst Appl 39(1):1582–1588
27. Shelokar PS, Jayaraman VK, Kulkarni BD (2004) An ant colony approach for clustering. Anal Chim Acta 509(2):187–195
28. Barrow JD, Davies PCW, Harper CL Jr (2004) Science and ultimate reality: quantum theory, cosmology, and complexity. Cambridge University Press, Cambridge
29. Mirjalili S, Mirjalili SM, Hatamlou A (2016) Multi-verse optimizer: a nature-inspired algorithm for global optimization. Neural Comput Appl 27(2):495–513
30. Shukri S, Faris H, Aljarah I, Mirjalili S, Abraham A (2018) Evolutionary static and dynamic clustering algorithms based on multi-verse optimizer. Eng Appl Artif Intell 72:54–66
31. Kumar P, Garg S, Singh A, Batra S, You Ilsun N (2018) Mvo-based two-dimensional path planning scheme for providing quality of service in uav environment. IEEE Internet Things J 5:1698–1707
32. Benmessahel I, Xie K, Chellal M (2017) A new evolutionary neural networks based on intrusion detection systems using multiverse optimization. Appl Intell 48:1–13
33. Janiga D, Czarnota R, Stopa J, Wojnarowski P, Kosowski P (2017) Performance of nature inspired optimization algorithms for polymer enhanced oil recovery process. J Petrol Sci Eng 154:354–366
34. Ewees AA, El Aziz MA, Hassanien AE (2017) Chaotic multi-verse optimizer-based feature selection. Neural Comput Appl 31:1–16
35. Faris H, Hassonah MA, Ala'M A-Z, Mirjalili S, Aljarah I (2017) A multi-verse optimizer approach for feature selection and optimizing svm parameters based on a robust system architecture. Neural Comput Appl 30:1–15
36. Alyasseri ZAA, Khader AT, Al-Betar MA, Abasi AK, Makhadmeh SN (2019) EEG signals denoising using optimal wavelet transform hybridized with efficient metaheuristic methods. IEEE Access 8:10584–10605
37. Makhadmeh SN, Khader AT, Al-Betar MA, Naim S, Abasi AK, Alyasseri ZAA (2019) Optimization methods for power scheduling problems in smart home: survey. Renew Sustain Energy Rev 115:109362
38. Shehab M, Khader AT, Laouchedi M, Alomari OA (2019) Hybridizing cuckoo search algorithm with bat algorithm for global numerical optimization. J Supercomput 75(5):2395–2422
39. MacQueen J, et al (1967) Some methods for classification and analysis of multivariate observations. In: Proceedings of the fifth Berkeley symposium on mathematical statistics and probability, vol 1. Oakland, CA, USA, pp 281–297
40. Park H-S, Jun C-H (2009) A simple and fast algorithm for k-medoids clustering. Expert Syst Appl 36(2):3336–3341
41. Bezdek JC (2013) Pattern recognition with fuzzy objective function algorithms. Springer, Berlin
42. Chen C-H (2017) Improved tfidf in big news retrieval: an empirical study. Pattern Recogn Lett 93:113–122
43. Hussain SF, Haris M (2019) A k-means based co-clustering (KCC) algorithm for sparse, high dimensional data. Expert Syst Appl 118:20–34
44. Vishwakarma S, Nair PS, Rao DS (2017) A comparative study of K-means and K-medoid clustering for social media text mining. Int J Adv Sci Res Eng Trends 2(11):297–302
45. Balabantaray RC, Sarma C, Jha M (2015) Document clustering using k-means and k-medoids. arXiv preprint arXiv:1502.07938
46. Aggarwal CC, Zhai CX (2012) A survey of text clustering algorithms. In: Aggarwal C, Zhai C (eds) Mining text data. Springer, Boston, MA, pp 77–128
47. Al-Betar MA, Alomari OA, Abu-Romman SM (2019) A TRIZ-inspired bat algorithm for gene selection in cancer classification. Genomics 112:114–126
48. Zeugmann T et al (2011) Particle swarm optimization. In: Sammut C, Webb GI (eds) Encyclopedia of machine learning. Springer, Boston, pp 760–766
49. Alam S, Dobbie G, Koh YS, Riddle P, Ur Rehman S (2014) Research on particle swarm optimization based clustering: a systematic review of literature and techniques. Swarm Evolut Comput 17:1–13
50. Esmin AAA, Coelho RA (2013) Consensus clustering based on particle swarm optimization algorithm. In: 2013 IEEE international conference on systems, man, and cybernetics. IEEE, pp 2280–2285
51. Makhadmeh SN, Khader AT, Al-Betar MA, Naim S, Alyasseri ZAA, Abasi AK (2019) Particle swarm optimization algorithm for power scheduling problem using smart battery. In: 2019 IEEE Jordan international joint conference on electrical engineering and information technology (JEEIT). IEEE, pp 672–677
52. Song W, Li CH, Park SC (2009) Genetic algorithm for text clustering using ontology and evaluating the validity of various semantic similarity measures. Expert Syst Appl 36(5):9095–9104
53. Akter R, Chung Y (2013) An evolutionary approach for document clustering. IERI Procedia 4:370–375
54. Karaa WBA, Ashour AS, Sassi DB, Roy P, Kausar N, Dey N (2016) Medline text mining: an enhancement genetic algorithm based approach for document clustering. In: Applications of intelligent optimization in biology and medicine. Springer, pp 267–287
55. Karaboga D (2005) An idea based on honey bee swarm for numerical optimization. Technical report, Technical report-tr06, Erciyes University, Engineering Faculty, Computer Engineering Department
56. Karaboga D, Gorkemli B, Ozturk C, Karaboga N (2014) A comprehensive survey: artificial bee colony (ABC) algorithm and applications. Artif Intell Rev 42(1):21–57
57. Alia OM, Al-Betar MA, Mandava R, Khader AT (2011) Data clustering using harmony search algorithm. In: International conference on swarm, evolutionary, and memetic computing. Springer, pp 79–88

58. Yang F, Sun T, Zhang C (2009) An efficient hybrid data clustering method based on k-harmonic means and particle swarm optimization. Expert Syst Appl 36(6):9847–9852
59. Karaboga D, Ozturk C (2011) A novel clustering approach: artificial bee colony (ABC) algorithm. Appl Soft Comput 11(1):652–657
60. Wang G-G, Gandomi AH, Alavi AH, Hao G-S (2014) Hybrid krill herd algorithm with differential evolution for global numerical optimization. Neural Comput Appl 25(2):297–308
61. Wang G-G, Gandomi AH, Yang X-S, Alavi AH (2016) A new hybrid method based on krill herd and cuckoo search for global optimisation tasks. Int J Bio-Inspir Comput 8(5):286–299
62. Huang A (2008) Similarity measures for text document clustering. In: Proceedings of the sixth New Zealand computer science research student conference (NZCSRSC2008), Christchurch, New Zealand, pp 49–56
63. Bansal B, Srivastava S (2018) Hybrid attribute based sentiment classification of online reviews for consumer intelligence. Appl Intell 49:1–13
64. Cutting DR, Karger DR, Pedersen JO, Tukey JW (2017) Scatter/gather: a cluster-based approach to browsing large document collections. In: ACM SIGIR forum, vol 51. ACM, pp 148–159
65. Raghuvanshi M, Patel R (2017) An improved document clustering with multiview point similarity/dissimilarity measures. Int J Eng Comput Sci 6(2):20285–20288
66. Bouras C, Tsogkas V (2012) A clustering technique for news articles using wordnet. Knowl Based Syst 36:115–128
67. Collective Evolution (2018) New physics theory questions the big bang: how did our universe really begin? Accessed 9 Aug 2018
68. Smithsonian Institution (2016) Can physicists ever prove the multiverse is real? Accessed 19 Apr 2016
69. Nanda SJ, Panda G (2014) A survey on nature inspired metaheuristic algorithms for partitional clustering. Swarm Evolut Comput 16:1–18
70. Subhadra K, Shashi M, Das A (2015) Extended ACO based document clustering with hybrid distance metric. In: 2015 IEEE international conference on electrical, computer and communication technologies (ICECCT). IEEE, pp 1–6
71. Abualigah LM, Sawaie AM, Khader AT et al (2017) β-Hill climbing technique for the text document clustering. In: Proceedings of the new trends in information technology (NTIT-2017). The University of Jordan, Amman, Jordan, 25–27 April 2017
72. Sayed GI, Darwish A, Hassanien AE (2018) A new chaotic multiverse optimization algorithm for solving engineering optimization problems. J Exp Theor Artif Intell 30(2):293–317
73. Jangir P, Parmar SA, Trivedi IN, Bhesdadiya RH (2017) A novel hybrid particle swarm optimizer with multi verse optimizer for global numerical optimization and optimal reactive power dispatch problem. Eng Sci Technol Int J 20(2):570–586
74. Sayed GI, Darwish A, Hassanien AE (2019) Quantum multiverse optimization algorithm for optimization problems. Neural Comput Appl 31:2763–2780
75. OA Arqub (2017) Adaptation of reproducing kernel algorithm for solving fuzzy fredholm-volterra integrodifferential equations. Neural Comput Appl 28(7):1591–1610
76. Arqub OA, Mohammed AL-S, Momani S, Hayat T (2016) Numerical solutions of fuzzy differential equations using reproducing kernel hilbert space method. Soft Comput 20(8):3283–3302
77. Awadallah MA, Al-Betar MA, Bolaji AL, Doush IA, Hammouri AI, Mafarja M (2020) Island artificial bee colony for global optimization. Soft Comput. https://doi.org/10.1007/s00500-020-04760-8
78. Arqub OA (2019) Application of residual power series method for the solution of time-fractional schrodinger equations in one-dimensional space. Fundam Inform 166(2):87–110
79. Arqub OA, Al-Smadi M, Momani S, Hayat T (2017) Application of reproducing kernel algorithm for solving second-order, two-point fuzzy boundary value problems. Soft Comput 21(23):7191–7206
80. Deepa M, Revathy P, Student P (2012) Validation of document clustering based on purity and entropy measures. Int J Adv Res Comput Commun Eng 1(3):147–152
81. Del Buono N, Pio G (2015) Non-negative matrix tri-factorization for co-clustering: an analysis of the block matrix. Inf Sci 301:13–26
82. Lin Y-S, Jiang J-Y, Lee S-J (2014) A similarity measure for text classification and clustering. IEEE Trans Knowl Data Eng 26(7):1575–1590
83. Forsati R, Keikha A, Shamsfard M (2015) An improved bee colony optimization algorithm with an application to document clustering. Neurocomputing 159:9–26
84. Tan SC, Ting KM, Teng SW (2011) A general stochastic clustering method for automatic cluster discovery. Pattern Recogn 44(10–11):2786–2799
85. Wei T, Yonghe L, Chang H, Zhou Q, Bao X (2015) A semantic approach for text clustering using wordnet and lexical chains. Expert Syst Appl 42(4):2264–2275
86. Saini N, Saha S, Bhattacharyya P (2019) Automatic scientific document clustering using self-organized multi-objective differential evolution. Cogn Comput 11:271–293
87. Wilcoxon F (1945) Individual comparisons by ranking methods. Biom Bull 1:80–83