applied sciences

MDPI

*Article*

# Enhanced Firefly-K-Means Clustering with Adaptive Mutation and Central Limit Theorem for Automatic Clustering of High-Dimensional Datasets

**Abiodun M. Ikotun [1] and Absalom E. Ezugwu [1,2,*]**

[1] School of Mathematics, Statistics, and Computer Science, Pietermaritzburg Campus, University of KwaZulu-Natal, King Edward Avenue, Pietermaritzburg 3201, South Africa
[2] Unit for Data Science and Computing, North-West University, Potchefstroom, 1 Hoffman Street, Potchefstroom 2520, South Africa
* Correspondence: ezugwua@ukzn.ac.za

**Abstract:** Metaheuristic algorithms have been hybridized with the standard K-means to address the latter's challenges in finding a solution to automatic clustering problems. However, the distance calculations required in the standard K-means phase of the hybrid clustering algorithms increase as the number of clusters increases, and the associated computational cost rises in proportion to the dataset dimensionality. The use of the standard K-means algorithm in the metaheuristic-based K-means hybrid algorithm for the automatic clustering of high-dimensional real-world datasets poses a great challenge to the clustering performance of the resultant hybrid algorithms in terms of computational cost. Reducing the computation time required in the K-means phase of the hybrid algorithm for the automatic clustering of high-dimensional datasets will inevitably reduce the algorithm's complexity. In this paper, a preprocessing phase is introduced into the K-means phase of an improved firefly-based K-means hybrid algorithm using the concept of the central limit theorem to partition the high-dimensional dataset into subgroups of randomly formed subsets on which the K-means algorithm is applied to obtain representative cluster centers for the final clustering procedure. The enhanced firefly algorithm (FA) is hybridized with the CLT-based K-means algorithm to automatically determine the optimum number of cluster centroids and generate corresponding optimum initial cluster centroids for the K-means algorithm to achieve optimal global convergence. Twenty high-dimensional datasets from the UCI machine learning repository are used to investigate the performance of the proposed algorithm. The empirical results indicate that the hybrid FA-K-means clustering method demonstrates statistically significant superiority in the employed performance measures and reducing computation time cost for clustering high-dimensional dataset problems, compared to other advanced hybrid search variants.

**Keywords:** clustering algorithms; metaheuristic algorithms; hybrid clustering; K-means; firefly algorithms; central limit theorem; high-dimensional datasets



**Citation:** Ikotun, A.M.; Ezugwu, A.E. Enhanced Firefly-K-Means Clustering with Adaptive Mutation and Central Limit Theorem for Automatic Clustering of High-Dimensional Datasets. *Appl. Sci.* **2022**, *12*, 12275. https://doi.org/10.3390/app122312275

Academic Editors: Agostino Forestiero and Xianpeng Wang

Received: 9 October 2022
Accepted: 28 November 2022
Published: 30 November 2022



## 1. Introduction

Explosive growth in data generation, acquisition, and storage has been observed recently, with significant and valuable knowledge hidden within this large amount of stored data. There is a need to extract the information and knowledge trapped within this massive data to improve organizations' decision-making processes. However, the explosive yet increasing data size makes the extraction process difficult and complex, surpassing the usual ability required to process, analyze, and understand the data [1]. Data mining as a part of knowledge discovery in databases is one method to address this challenge.

Data clustering is an essential unsupervised data classification technique in data mining. It involves grouping unlabeled data objects into clusters based on their similarities, such that objects within a cluster are more similar to each other than to data objects in

other clusters. It has found wide application in different areas such as pattern recognition, image analysis, artificial intelligence, machine learning, computer vision, recommendation systems, spatial databases, medicine, information retrieval, marketing, web mining, and statistics [2,3].

High-dimensional datasets involve those whose dimensions vary from a few scores to several thousands of dimensions [1]. According to [4], a dataset can be judged as large in three ways: when the number of elements in the dataset is large, when each element in the dataset has many dimensions, and when the dataset has many clusters. Large datasets are common in domains such as social media data, recommendation systems, microarray data, medicine, bioinformatics, text document clustering, and biology [1,5]. It is computationally expensive to use a traditional clustering algorithm for a high-dimensional dataset [5]. This problem associated with clustering high-dimensional data is generally referred to as the "curse of dimensionality".

The K-means algorithm is a traditional clustering algorithm that has gained wide popularity and acceptability with wide usage based on its efficiency, ease of implementation, and simplicity. It is categorized as a partitional clustering algorithm where data objects of datasets are divided into separate groups such that each data object can only belong to a single group. It employs a distance-based optimization criterion to minimize the intra-cluster distance and maximize the inter-cluster distance. Several limitations, however, have been identified in using the K-means clustering algorithm, including sensitivity to initialization parameters, undesirable sample distribution vulnerability, and susceptibility to noise [6,7]. The optimum performance of K-means is premised on the specification of the correct number of clusters (which is difficult to determine in a real-life dataset) and the selection of the optimum number of the initial cluster centroid. The K-means algorithm uses the hill-climbing approach in its search for the optimum cluster centroid resulting in local search around the initial cluster centroid with the algorithm having a high probability of getting trapped into local optima. Moreover, the clustering process of the classical K-means uses square distance limits to discover spherical-shaped clusters, which is often unsuitable considering the actual nature of the complex data distribution of real-life datasets. The clustering of real-life datasets characterized by high-dimensionality, presence of noise and outliers, imbalance, sparse and irregular sample distribution, and narrow or overlapping cluster margins poses many challenges to the K-means algorithm. According to Xie et al. [8], the K-means finds it difficult to obtain the desired clustering results when dealing with a high-dimensional dataset due to the dimensional disaster impact, data size, noise, and distribution with low computational efficiency. The wide acceptability and usage of K-means makes its performance enhancements inevitable; thus, this paper aims at addressing the high computation time challenge of K-means due to the inherent problem of the curse of dimensionality in big data. to improve its performance in finding solution to automatic clustering problems of big data.

Several approaches have been proposed in the literature to address the high-computation-time problems of the classical K-means algorithm in clustering high-dimensional data. According to Xie et al. [8], a common approach is the dimensionality reduction approach which involves seeking and clustering within low-dimensional features of high-dimensional data. However, some of the difficulties identified in this approach are that there is a need first to determine whether low-dimensional data are the dominating feature needed for clustering practical problems, and if the distance mapping between low-dimensional data points is conductive clustering [8]. In Alguliyev et al. [9], a K-means-based parallel batch clustering algorithm was proposed, where the dataset is divided into equal partitions to reduce the exponential computation growth. Each partition's cluster centers are calculated using the K-means algorithm, and the generated cluster centroids are merged and clustered. Although the dataset's characteristics are preserved with increased clustering speed, determining the optimum number of clusters within each partition is still a problem. An improved K-means algorithm based on density canopy was proposed in [10] which uses density canopy as a preprocessing method for the K-means algorithm to determine the number

of clusters and initial cluster centroids. According to the authors, the main goal is to resolve the initialization problems, while the issue with clustering big data in terms of high computation time is not addressed. In Alguliyev et al. [11], a batch clustering algorithm for big data sets is proposed where large data sets are clustered in batches. Each batch is clustered using the K-means algorithm, with cluster centroids of the previous batch added to the subsequent batches until all batches are clustered. The cluster centroid obtained from the final batch is used as the final cluster centroid for the final data points' assignments. Although the exponential growth of computational time is avoided, there is no guarantee that the final cluster centroid is optimum for the entire dataset. In Olukanmi et al. [12], a simple algorithm that addresses the poor scalability of K-means in relation to clustering large datasets based on a statistical inference approach was proposed. Their algorithm uses samples from the dataset to compute the centroids based on an intuitive extension of the classical central limit theorem (CLT).

In recent times, hybridizing K-means with metaheuristics has also assisted in solving most of the problems associated with real-life datasets based on their global search capability for the optimized number of clusters and cluster centroids [6,13–21]. Primarily, metaheuristics algorithms have been used to help the K-means algorithm to escape from local optima convergence by searching for global optimum initial cluster centroids. Their effectiveness has been extensively validated in the literature [22]. However, some of the metaheuristic algorithms has been hybridized with K-means algorithm to improve its performance in solving automatic clustering problems [23]. In this paper, a further enhancement is proposed for the firefly-based K-means hybrid algorithm for automatic clustering of high-dimensional data by using a CLT-based K-means in the hybrid algorithm to reduce the number of distance calculations required in the K-means phase thereby reducing the algorithm's computational time.

The firefly algorithm (FA) is a population-based metaheuristic algorithm proposed by [24]. It has been applied to solve numerous problems in different fields because of its efficiency, robustness, versatility, and ability to find an approximate solution to NP-hard problems, among other benefits [25]. FA has been extensively used for automatic clustering [25,26] with superior clustering performance reported in literatures in comparison with other metaheuristic algorithms. The FA is easy to understand, simple to implement, and has been used successfully to find the solution to problems in different areas. It has outperformed several other metaheuristic algorithms and has demonstrated superior performance when hybridized with other metaheuristic algorithms [25,27]. According to Kumar and Kumar [28], the efficient performance of FA is based on the fact that it combines the advantages of some of the existing metaheuristic algorithms, namely, simulated annealing (SA), particle swarm optimization (PSO) and differential evolution (DE). Thus, FA is regarded as a generalized form of DE, SA and PSO [28]. The FA has been combined with K-means in several studies. In Hassanzadeh and Meybodi [29], the FA was combined with the classical K-means; similarly, the investigation presented in [30] combined the FA with parallel K-means, while [31] adopted optimized K-means and canopies in their hybridization with FA. Behera et al. [16] used fuzzy C-means with FA. The study presented in [32,33] aimed to improve the general clustering performance in their modification of FA hybridized with the K-means algorithm. The popularity of the FA among global optimization researchers is due to the algorithm's application versatility in terms of robustness and performance superiority, and because it is backed with mathematical proofs in dealing with diverse, complex optimization problems encountered in the real-world, making it the first choice for enhancing the inferior performance stability of the classical K-means clustering algorithm [16]. However, the primary motivation for selecting the FA as a global optimization enhancer for the classical K-means is because of its multimodality features, adaptability to the problem landscape, and automatic subdivision of its population into subgroups, which best fits the context of the problem at hand [16].

In this paper, an enhanced adaptive mutation-based FA is combined with the classical K-means for the automatic clustering of high-dimensional data with a focus on addressing

the curse of dimensionality problem of big data; this paper adopted the idea from the CLT proposed in [12] to break the high-dimensional dataset into several overlapping subsets on which the K-means algorithm is applied to obtain representative cluster centers for the final clustering procedure. This preprocessing task reduces the number of distance calculations needed in the K-means phase of the hybrid algorithm, thereby reducing the computation time required for the clustering process to ensure scalability when handling high-dimensional datasets. An initial investigation was conducted involving the proposed algorithm and seven other metaheuristic-based K-means modelled using the same framework as the proposed algorithm to justify the selection of the proposed algorithm. The proposed hybrid algorithms' performance was investigated using 20 high-dimensional datasets, with the Davies Bouldin cluster validity index (DBI) [34] serving as the measuring index to determine the validity of the clustering solutions obtained.

The rest of the paper is outlined as follows: Section 2 presents the related works on FA-based K-means hybrid algorithms. Section 3 describes the classical firefly algorithm, K-means algorithms, the central limit theorem, and its application in the design of a scalable K-means algorithm. It also describes the proposed improved FA-based K-means hybrid algorithm. Section 4 presents the evaluation of the hybrid algorithm on high-dimensional datasets and comparison with other metaheuristic algorithms from the literature. The conclusion and future research direction are presented in Section 5.

## 2. Related Research

The FA is a metaheuristic algorithm that has been applied to solve numerous problems in different fields as earlier stated. It has the unique capability of automatic subdivision, which gives it the advantage of handling multimodal optimization problems compared with other metaheuristic algorithms [28]. It records superior performance in clustering analysis due to the clustering process's high non-linearity and sub-optimal distraction [6]. A systematic review of the FA is presented in [28] describing the various characteristics and variants of the algorithm. A comprehensive review of the FA regarding the various areas of its successful application in a wide spectrum of real-world applications is discussed in [35], and a performance study using classification error percentage criteria on the FA for cluster analysis is presented in [36]. The algorithm used in this study was able to generate the optimal cluster centroids, and their study acknowledged the algorithm's strength. The authors' report shows that the classification efficiency of the FA is superior compared with that of PSO, artificial bee colony (ABC) and other clustering methods.

For enhancement of the FA for clustering problems, its hybridization with other metaheuristic algorithms has been reported in the literature. An automatic data clustering using a hybrid firefly particle swarm optimization algorithm was reported by [26], where an improved FA was integrated with PSO (FAPSO). The study result showed superior performance over each of the individual clustering algorithms FA, PSO, and classical DE with a high level of stability. Comparisons with other hybrid algorithms from the literature (ACDE, DCPSO, and GCUK) were also performed, and FAPSO outperformed the competing hybrid algorithms. Rajah and Ezugwu [37] reported a hybrid of SOS with FA (SOSFA) alongside other SOS-based hybrid algorithms for handling automatic clustering. SOSFA was reported as having superior performance in some of the datasets used during the algorithms' performance investigation.

Moreover, a comparative study of hybrid FAs for automatic data clustering was performed and discussed in [38]. Their work investigated automatic clustering tasks on unlabeled, large-scaled and high-density datasets using four firefly-based hybrid algorithms. The FA was combined with PSO, ABC, invasive weed optimization (IWO), and teaching–learning-based optimization (TLBO). These hybrid algorithms require no prior knowledge of the data object to be classified, and the optimal number of clusters is automatically determined during the execution of the hybrid algorithms. The cluster separation index (CSI) and DBI cluster validity indices were used to evaluate the hybrid algorithms' performances on 12 University of California Irvine (UCI) machine learning repository

datasets. The clustering results were compared with other hybrid algorithms from the literature with a noticeable performance improvement. However, it was observed that there was an exponential increment in the computational cost of the hybrid algorithm relative to the population size.

Based on the superior performances of the FA in handling both general and automatic clustering, a few hybridizations of FA with K-means have been reported in the literature. One of the earliest hybridizations with K-means is presented by [29] as a new approach to clustering algorithm using FA tagged KFA. The FA finds optimum k-numbered cluster centroids for the K-means algorithm to refine the cluster centroids. Their proposed algorithm used a modified FA that uses global optima in the firefly's movement in such a way that the firefly with the maximum or minimum value influences the movement of other fireflies. The cartesian distance was used to calculate the distance of fireflies for global optima convergence. The performance of their algorithm was evaluated using five UCI datasets and compared with other clustering algorithms. Their result recorded a relatively stable algorithm with better clustering efficiency. Performance on the high-dimensional datasets was not investigated, and the classical K-means algorithm was used.

The hybridization of the K-means algorithm with firefly using parallel architecture to handle a large number of clusters was proposed by [30]. The K-means algorithm was used in refining the initial optimal cluster centroids generated by the FA for improved clustering accuracy. The FA was executed with the K-means algorithm for data clustering in a sequential manner to avoid local optimal convergence entrapment. Parallel architecture was adopted to reduce the execution time for large dimensional datasets, thus making this class of problems more computationally feasible which otherwise would have been computationally prohibitive. The proposed algorithm's performance was evaluated using four UCI repository datasets using six validity metrics (DBI, sum of squares within (SSW), sum of squares between (SSB), Dunn-Dunn index, accuracy, and silhouette coefficient) to establish the validity of the clustering results. Their results were compared with the standard parallel K-means algorithm showing a better clustering result with higher accuracy.

The FA was hybridized with fuzzy C-means in [16]. The fuzzy C-means (FCM) algorithm is a variant of the K-means algorithm that uses a generalized least square objective function to generate fuzzy partitions and prototypes for numerical data [39]. The FA was integrated with the FCM to generate initial optimal cluster centroids for the FCM and avoid the latter from converging into local optimal. The performance of the hybridized FCM-FA was investigated using various real-world datasets and compared with other algorithms. The experimental results showed a better and more effective clustering algorithm. In the same vein, Nayak et al. [32] leveraged the global search capacity of the FA to assist the classical K-means in escaping convergence into local minimal in their proposed firefly-based K-means (FA-K-means). Their simulation results showed that the hybrid algorithm can efficiently handle the clustering problems without the K-means getting trapped into local minimal and faster convergence.

The FA was integrated with the canopy pre-clustering algorithm in [31] and hybridized with K-means. The canopy pre-clustering algorithm uses an approximate and inexpensive distance measure to partition initial data into overlapping subsets tagged as canopies. The main clustering is then performed using distances between points that occur in the common canopy [40]. The Haberman's survival dataset from the UCI repository was used to investigate their proposed algorithm's performance. The result showed that the proposed data clustering algorithm had a better classification accuracy when compared with the traditional K-means algorithm.

Two variants of the FA were hybridized with K-means by [6] to resolve the K-means algorithm's local optimal trap and initialization sensitivity problems. Their hybrid model was based on the FA's unique property of automatic sub-division and ability to tackle multimodal optimization problems of the firefly algorithm. To improve the performance of the FA, matrix-based search parameters and dispersing mechanisms were embedded into the two FA variants. These enhance the exploration and exploitation capability of

the variants. In the first variant, a randomly generated control matrix replaces the FA's attractiveness coefficient to elevate the one-dimensional search mechanism to a multi-dimensional one. In the second variant, fireflies with high similarities are moved to a new position using the dispersing mechanism for global exploration. The dispersing mechanism introduces enough variance between fireflies to increase search efficiency. Their proposed algorithms were evaluated on 15 UCI datasets and a skin lesion dataset. The algorithms significantly outperformed the classical K-means algorithm in performance and distance measure.

The FA-based-K-means hybrid algorithms have also been applied in solving some real-life problems. The FA-K-means algorithm was applied for color image quantization, computer graphics, and processing problems to reduce colors in an image with minimum distortion [41]. The hybrid algorithm was tested on three commonly used images and the results were compared with the outputs from classical K-means and conventional FA. Their algorithm produced a better result than the baseline techniques. A hybrid algorithm that integrated FA-based K-means with wavelet transform and FA-based support vector regression was proposed by [42] as a three-stage forecasting model to develop a prediction system for export trade value. The FA-based K-means was adopted to perform cluster analysis on the export trade value dataset. The prediction model was then built for each cluster, resulting in better performance of the prediction system.

In Kaur et al. [18], the FA-based K-means algorithm was applied for data-clustering purposes in the clustering-based intrusion detection system (IDS), which replaced the regular signature-based IDS. The clustering phase was used to build the training model for the IDS, which then evaluated the test set using classification. Compared with the results from other clustering algorithms, an impressive result was observed when the system was tested on the NSL-KDD (network security laboratory–knowledge discovery dataset). K-member fuzzy clustering was integrated with the FA by [43] for a combined anonymizing algorithm for privacy preservation in social networks. In the hybrid algorithm, the modified version of the fuzzy C-means—the K-member fuzzy algorithm—is employed for creating balance clusters, with each cluster having at least K-members. The FA is then used to optimize the clusters to anonymize the network graph and data. The system was tested using four social network databases and observed a minimal loss of information on the published graph and data.

Using a firefly-based K-means hybrid algorithm, a meteorological system was built to accurately and quickly estimate reference evapotranspiration [37]. Reference evapotranspiration is used in designing irrigation schedules, determining crop water requirements, and planning and managing agricultural water resources when the available meteorological data are limited. They coupled K-means clustering and FA with a novel kernel extreme learning machine model alongside limited meteorological data ranging from 5 to 40 with pooled temperature data from 26 weather stations in parallel computation to estimate monthly evapotranspiration in the Poyang Lake basin of South China. Their algorithm outperformed the existing methods in terms of the count of absolute errors. There was a significant reduction in the computation time recorded by the proposed parallel algorithm compared with its sequential counterpart and other competing existing methods.

Most of the existing methods focused on addressing the fundamental problems of local optimal convergence and initialization sensitivity of the K-means algorithm except [30,31]. Hence, all except [31] used the convectional K-means algorithm or its fuzzy c-means variants in their hybridization with the FA. A parallel implementation approach was adopted by [30] to reduce the computation time for high-dimensional datasets. In Nayak et al. [31], the problem of the curse of dimensionality is also considered; the authors incorporated the canopy pre-clustering method along with their proposed firefly-K-means hybridization algorithm. The canopy pre-clustering method requires two parameter specifications of T1 and T2 representing the two distance thresholds required to determine the canopy center that a data point will be assigned to, in order to specify the accurate number of clusters. Determining the accurate distance thresholds for canopy formation for optimum clustering

is difficult. In this work, we aimed to achieve better clustering robustness, clustering optimality and efficient clustering results with reduced computational time when dealing with the issue of high-dimensional datasets.

## 3. Methodology

This section describes the classical FA model, the conventional K-means algorithm model, the central limit theorem, and its application in the design of a scalable K-means algorithm. It also presents the hybridization models of the CLT-based hybrid FA-K-means algorithm for the automatic clustering problem. The DBI is also described.

### *3.1. Firefly Algorithm*

The FA is a nature-inspired metaheuristics algorithm developed by Yang et al. [44]. It is characterized as a swarm-based metaheuristic optimization algorithm with a high convergence rate and short execution time compared to other metaheuristic techniques.

Fireflies emit bioluminescence flashes as a signaling system to communicate among themselves. The signaling system uses the flashing rate, the flashes' intensity, and the amount of time between the flashes to pass a message across to other fireflies. The brightness of a firefly flash and the timing accuracy of the flashes determines its attractiveness [44]. The characteristics of the signaling system inherent in the firefly formed the basis for developing the FA. A nonlinear term based on the exponential decay of light absorption and the inverse square of light variation with distance is applied in simulating the variation in the intensity of the light flashes of the firefly to determine its attractiveness [44].

In finding the optimization problem's solution vector, the FA algorithmic equation is given in Equation (1), illustrating the firefly's movement from point $i$ to point $j$

$$X_i^{t+1} = X_i^t + \beta_0 e^{-\gamma r_{ij}^2}\left(X_j^t - X_i^t\right) + \alpha\epsilon_i^t \quad (1)$$

where:

$\alpha$—a scaling factor that controls the random walk step sizes.
$\beta_0$—the attractiveness constant when the distance between two fireflies equals zero, that is, $rij = 0$.
$\gamma$—the scale-dependent parameter that controls the visibility of the fireflies.
$\beta_0 e^{-\gamma r_{ij}^2}\left(X_j^t - X_i^t\right)$—gives the nonlinear attractiveness of a firefly, which varies with distance.
$\alpha\epsilon_i^t$—the randomization term (where $\epsilon_i^t$ refers to the use of Gaussian distribution for generating random values at each iteration).

The distance between two fireflies can be defined as the Cartesian distance between them. Sometimes, it could be the delay time in a routing problem or even the Hamming distance between them for combinatorial problems. The firefly's brightness is associated with its objective landscape with its position as the indicator; thus, its attractiveness is determined by its relative positions and relative brightness in relation to another firefly. This necessitates a paired comparison of fireflies.

Three assumptions are made in the basic design of the FA: one, all fireflies are considered as unisex; two, the attractiveness of a firefly is directly proportional to its brightness; and three, the fitness function landscape determines the brightness of the flashlight produced by any firefly. The distance and intensity of light emitted into the atmosphere affects the brightness of any firefly. This is illustrated in Equation (2).

$$I(r) = I_0 e^{-\gamma r^2} \quad (2)$$

where:

$I_0$—the intensity of light when $r = 0$.
$\gamma$—the coefficient of light absorption.
$r$—the distance.

The distance between two fireflies is illustrated by Equation (3)

$$r_{ij} = \|x_i - x_j\| = \sqrt{\sum_{k=1}^{d}\left(x_{i,k} - x_{j,k}\right)^2} \tag{3}$$

where:

$d$ is the dimension of the problem.

For an automatic clustering problem, a given dataset $DS = \{ds_1, ds_2, \ldots, ds_n\}$ with dimensions $d_i(i = 1, 2, \ldots, n)$ is required to be partitioned into non-overlapping groups called clusters $C = \{c_1, c_2, \ldots, c_k\}$ with cluster centers $cc_i(i = 1.2. \ldots, k)$ such that:

$$C_i \cap C_j = \varnothing, \qquad i, j = 1, 2, \ldots, K, \quad i \neq j \tag{4}$$

$$C_1 \cup C_2 \cup \ldots \cup C_k = DS \tag{5}$$

where:

$C_i\ DS\ and\ C_i \neq 0,\ i = 1, 2, \ldots, K$

At the initialization stage, a population of size $n$ is defined as $X = (x_1, x_2, \ldots, x_n)$ as a representative data point with reference to a clustering problem where $x$ is a $k$ $x$ $d$-dimensional vector $DS_{n \times b}$ defined as $X = x_1, x_2, \ldots, x_k = (x_{11}, x_{12}, \ldots, x_{1d}), (x_{21}, x_{22}, \ldots, x_{2d}), \ldots, (x_{k1}, x_{k2}, \ldots, x_{kd})$. The FA uses Equation (5) as the optimization function to minimize the sum of the distance between the datasets $ds$ and the cluster center $cc$ with the lower and upper bound set as $\min\{ds_1, ds_2, \ldots, ds_d\}$ and $\max\{ds_1, ds_2, \ldots, ds_d\}$ represented as $lb = (lb_1, lb_2, \ldots, lb_d)$ and $ub = (ub_1, ub_2, \ldots\ldots ub_d)$, respectively, for the number of groups in the population.

To find the solution to the clustering problem, the $i$th particle $X_i$ is evaluated using Equation (6)

$$X_i = rand\ (1, k \times d)\ .^*(ub - lb) + lb \tag{6}$$

where $rand(1, k \times d)$ represents a vector of uniformly distributed random integer numbers with values between 0 and 1. The FA algorithm uses the squared error function shown in Equation (7) as its objective function.

$$f(DS, C) = \sum_{i=1}^{n} min\left\{\|dc_i - c_j\|^2 \;\middle|\; j = 1, 2, \ldots, k\right\} \tag{7}$$

For cluster analysis, the cluster centers are the decision variables, and the cluster validity index is used as the fitness function to evaluate the quality of each firefly with respect to its light intensity.

For a more efficient clustering task, the concept of mutation strategy suggested by [38] is incorporated to improve the exploitation and exploration of the FA. The mutation operator probability represented by Equation (8) is used for additional diversity among the swarm of fireflies.

$$M_p = f((x_{new}) - f(x_{old}) \tag{8}$$

where $f(x_{new})$ is the fitness value of the new firefly and $f(x_{old})$ is the fitness value of an old firefly. For optimum performance of FA, Kumar and Kumar [28] suggested the range of values for the algorithm parameters $\beta$, $\gamma$, and $\alpha$. The mutation strategy leverages the desirable features of the attractive fireflies added to the less bright fireflies to enhance their attractiveness. The mutation probability is used to calculate the extent of the feature enhancement modification of any identified less bright firefly, such that fireflies with low light intensity have a higher mutation probability. In comparison, those whose light intensity is high have a lower mutation probability [38]. The introduction of the mutation probability ensures that good-quality solutions are not reduced and low-quality solutions

are improved. The basic algorithm listing for the standard FA can be found in [24], while the improved FA used in this paper is given in Algorithm 1 [38].

The choice of using the FA over other metaheuristic algorithms for improving the clustering capability of the classical K-means algorithm is justified by the track record of the application of FA in handling similar clustering problems as presented in the literature. Highlights of some of the outstanding performances of the FA in achieving better clustering results are discussed in the extensive experimentation presented in [38].

**Algorithm 1:** Pseudocode for the improved FA

**Input** Data points $X = \{x_1, x_2, \ldots, x_n\}$

**Output** Obtained best location : $x_{1min}$

**Begin**
  Define the initial parameters for the firefly algorithms : $MaxGen$, $\beta$, $\gamma$, $\alpha$, *and* $n$
  Specify the fitness function $fitfunc(a)$, $\ni$ $a = (a_1, a_2, \ldots, a_D)^{iter}$
  Generate $n$ initial positions of fireflies $(i = 1, 2, \ldots, k)$
  Calculate the $fitfunc(a)$ to generate the light intensity $Lint_i$ of firefly $a_{firefly}(i)$
**Do**
    **for** $i$ = 1:$n$
      **for** $j = 1 : n$
        **if** $Lint_i < Lint_j$
          Move $a_{firefly}(i)$ towards $a_{firefly}(j)$ using Equations (2) and (3);
**end**
        Compute $MP = ff\left(a_{fireflynew}\right) - ff\left(a_{fireflyold}\right)$
Execute mutation ()
**End**
      Compute the attractiveness variance with distance $r$ using $\exp(-\gamma r)$

Compute new fitness value for all fireflies
Accept new solutions with the best fitness
**End**
    Perform firefly light intensity $Lint_i$ update
    Increment algorithm counter $iter = iter + 1$
    Perform $\alpha$ reduction by a factor;
  **While** $iter < MaxGen$
**End**

### 3.2. K-Means Algorithm

The K-means algorithm uses the Euclidean distance as the objective function to group $N$ data points of $D$-dimensions into a predefined number of clusters $k$. The algorithm selects a user-specified $k$ number of data points at the initialization level to represent the initial cluster centroids. Two significant steps follow this: the assignment step and the centroid update step, which are iteratively repeated until the convergence condition is achieved. In the assignment step, the algorithm calculates the distance of each data point from the cluster centroids and assigns it to the closest cluster. The centroid update is then performed by calculating the cluster's mean to find a new cluster center. The algorithm stops when a maximum number of iterations has been reached or when there are no longer changes in the centroid vectors over many iterations. According to [45–48], the basic steps of the K-means algorithm are:

i. Perform an initial partition into $k$ number of clusters based on user-specified $k$.
ii. Repeat steps 3 and 4 until cluster membership is stable.
iii. Assign each data object to the closest cluster to it to generate a new partition.
iv. Perform cluster center update.

The pseudocode for the basic steps of the classical K-means algorithm is given in Algorithm 2 [23].

**Algorithm 2:** Standard K-means Pseudocode

**Input** :Array D $\{d_1, d_2, \ldots, d_n\}$ //Input Dataset
$k$ //Specified k number of clusters
**Output** :$FCC = \{fcc_1, fcc_2, \ldots, fcc_k\}$//Final cluster centroids
**1** **Begin**
**2** //Parameter Initialization
**3** $A = (a_1, a_2, \ldots, a_n\})$ //dataset
**4** $RCC = (rcc_1, rcc_2, \ldots, rcc_k\})$//Select initial cluster centroids randomly
**5** **Do**
**6** //Calculations of Distance
**7** **for** $i = 1$ *to* $n$ do
**8** **for** $j = 1$ *to* $k$ do
**9** Compute data objects' Euclidean distance to all clusters
**10** **end** $j$
**11** //Data object Assignment to clusters
**12** Assign data objects to the closest cluster
**13** **end** $i$
**14** //Updating Cluster centroid
**15** Evaluate the new cluster centroid
**16** **While** cluster centroids' difference of two consecutive iterations are not the same
**End**

The number of distance calculations in the convectional K-means algorithm increases as the data points in a dataset increase. This increases the computation time of the algorithm. As a result, the classical K-means scalability in terms of a high-dimensional datasets is poor. There is a significant performance increase and efficiency in the K-means algorithm's clustering process if the distance calculations required are significantly reduced.

Some distance-calculation reduction strategies have been proposed in the literature. The use of Kd tree-based techniques was suggested by [48,49] and the single pass method is another strategy proposed by [50,51]. For these two techniques, the uniformity of the cluster centroid is not guaranteed. In view of this, Jin et al. [52] proposed the exact method, which corrects the approximate cluster centroid as an improvement on the earlier techniques. In Domingos and Hulten. [53], a general framework for scaling machine learning algorithms was developed and applied to K-means. The objective was to develop a system that guarantees results close to what is obtainable for infinite data points while minimizing the running time. Their method is based on computation of bounds for the loss as a function of the number of data points used at each step.

A distance-calculation reduction strategy inferred from the classical central limit theorem was proposed by [54]. It involves making inferences from a few small samples of data points to avoid exhaustive comparisons between the data points and centroids. This method is incorporated in this hybridization of K-means with the FA for automatic clustering of high-dimensional datasets.

### *3.3. The Central Limit Theorem*

The classical central limit theorem (CLT) is fundamental and popular in probability theory and inferential statistics. It is mostly applicable in many practical scenarios where the exhaustive study of the entire population of interest is not feasible. In CLT, samples from large populations are used to draw a meaningful conclusion about the population. The central limit theorem states that "Given a population with a mean $\mu$ and standard deviation $\sigma$, regardless of the distribution of the population; the distributions of means $\mu_1 \ldots \mu_n$, of $n$ samples of size $s$ collected from the population approaches a normal distribution with mean $\mu$ and standard deviation $\frac{\sigma}{\sqrt{s}}$ as $n, s \to \infty$".

The statistical tradition for sample size given as $n, s \geq 30$ is considered sufficiently large enough for the inference to be true [54]. According to Olukanmi et al. [54], the K-means algorithm is a particular case of the classical CLT with k = 1 (which can be

generally inferred for other values of k) where the entire dataset is regarded as a single cluster k with the $\mu$ representing the centroid. In K-means clustering, the central $\mu$ for the entire population can be obtained by clustering several data samples from the entire population to find the means $\mu_1 \ldots \mu_n$ of $n$ samples of size $s$ which represents the centroids of the $n$ data samples. Clustering the collection of the $n$ data sample means $\mu_1 \ldots \mu_n$ (the centroids for the sample means) using $k = 1$ produces a central mean $\mu$ of the entire population as $n, s$ grows in value.

Based on this basic understanding of the central limit theorem and its inference on the design of the K-means algorithm, the CLT-based version follows these simple steps [54]:

i. Select $n$ number of samples such that $n, s \geq 30$.
ii. Use the K-means algorithm on the selected data samples and store the cluster centroids of each sample.
iii. Combine the $n$ numbered cluster centroids obtained from step 2 to produce a new population of cluster centroids of size nk.
iv. Perform the data assignment step using the centroids obtained from step 3.

The pseudocode for the basic steps of the CLT-based K-means algorithm is given in Algorithm 3.

### 3.4. The Proposed CLT-Based FA-K-Means Model

The proposed improved hybridized algorithm focuses on improving the scalability of the FA-K-means algorithm on high-dimensional datasets. The advantages of FA and K-means have been combined and enhanced by reducing the computation time required for clustering datasets with massive data points and those with higher dimensions. An improved FA suggested by [38] introduced a mutation priority into the classical FA to maintain a good exploitation and exploration balance during the search process. The improved FA increases convergence speed, population diversity, and classical FA's solution accuracy.

For the implementation strategy of our algorithm, the search process is initiated by the FA for a global search of the search space. This is preferred because of its ability for strong exploration of the search space. The K-means is then used as a local search algorithm for intense exploitation around the promising region, thus enhancing the intensification process of the proposed hybrid algorithm. The proposed hybrid algorithm's efficiency and effectiveness are measured using the DBI discussed in the next section. The DBI assists in finding the best partition and also assists in determining the optimal number of clusters.

The search process commences by generating an initial population of fireflies, each representing a candidate solution. The fitness of each candidate solution is evaluated using the DBI cluster validity index. Subsequently, in an iterative manner, new solutions with the best fitness values are updated using the operators of the firefly algorithm. The results obtained from the first phase are improved using the K-means algorithm. The best results generated by the FA algorithm are passed on to the K-means algorithm as its initial cluster centroids. The execution of the two phases to convergence forms a complete run cycle of the proposed algorithm. The best fitness with the smallest DBI value of the two cycles is compared, and the best of the two is selected as the optimal value for the run. The steps illustrated above are depicted in Algorithm 4, and the implementation flowchart of the proposed algorithm is presented in Figure 1.

**Algorithm 3:** Pseudocode for CLT-based K-means

**Input:** Array D $\{d_1, d_2, \ldots, d_n\}$ //input Dataset
$k$ //specified number of clusters
ns //number of samples
ss //sample size

**Output:** $FCC = \{fcc_1, fcc_2, \ldots, fcc_k\}$//final cluster centroids

*For I = 1 to ns*
//Generate random samples from dataset
$SA(i) = Datasample\ (DS, ss)$ //select Sample Dataset randomly
//Generate optimal cluster centroids for the data sample
//Run K-means algorithm on the data sample
**Begin**
//Parameter Initialization
$SA(i) = (\ sa_1, sa_2, \ldots, sa_{ss}\})$ //dataset
$RSACC(i) = (\ rsacc_1, rsacc_2, \ldots, rsacc_k\ \})$//Select initial cluster centroids randomly
**Do**
//Calculations of Distance
**for** $i$ = 1 *to ss* do
**for** $j = 1$ *to* $k$ do
Compute data objects' Euclidean distance to all cluster centroids
**end** $j$
//Data object assignment to clusters
Assign data objects to the closest cluster
**end** $i$
//Updating cluster centroids

2Evaluate the new cluster centroid
$IFCC(i) = \{ifcc_1, ifcc_2, \ldots, ifcc_k\}$
**While** cluster centroids' difference of two consecutive iterations are not the same
//Keep the generated sample cluster centroids as a representative dataset
Add the generated sample cluster centroid to the existing collation of cluster centroids
$IFCC = \{\ ifcc_1, ifcc_2, \ldots, ifcc_{k \times ns}\}End$
//
//Generate final cluster centroids from the combined cluster centroids obtained from each data sample
$IFCC = \{ifcc_1, ifcc_2, \ldots, ifcc_{k \times ns}\}$
//Run K-means on collated cluster centroids as the new datasets
**Begin**
//Parameter Initialization
$IFCC = \{\ ifcc_1, ifcc_2, \ldots, ifcc_{k \times ns}\}$//collated cluster centroids form the new dataset
$RIFCC = (\ rifcc_1, rifcc_2, \ldots, rifcc_k\ \})$//Select initial cluster centroids randomly
**Do**
//Calculations of distance
**for** $i = 1$ *to* $(ns \times k)$ do
**for** $j$ = 1 *to* $k$ do
Compute data objects' Euclidean distance to all clusters
**end** $j$
//Data object Assignment to clusters
Assign data objects to the closest cluster
**end** $i$
//Updating Cluster centroid
Evaluate the new cluster centroid
**While** cluster centroids' difference of two consecutive iterations are not the same
*End*

**Algorithm 4:** Pseudocode for the HFA-K-means Algorithm

**Input:** Data point $D = \{d_1, d_2, \ldots, d_n\}$

**Output:** Optimal Cluster centers $OCC = \{occ_1, occ_2 \ldots, occ_n\}$

**Begin**
Randomly create a $k$ number of cluster centres as the initial population
Evaluate the objective function using the DB cluster validity index
***for*** $i = 1$ *to* $n$
    Compute the cost function to determine the best individual using the Euclidean distance
***if*** current $Pop(i).Cost <= BestSol.Cost$
        Update Current $Pop(i)$ as the best solution;
    ***end if***
***end for***
***while*** iteration is not more than the maximum do
    ***for*** $i = 1$ *to* $n$
***for*** $j = 1$ *to* $n$
            ***if*** $Pop(j) < Pop(i).Cost$
                Use the firefly operators to move $Pop(i)$ towards $Pop(j)$
                ***if*** $NewSol.Cost <= NewPop(i).Cost$
*NewPop*(*i*) is made the new solution;
***if*** *NewPop*(*i*).*Cost* <= *BestSol.Cost*
                        Update *NewPop*(*i*) as the new solution
                    ***end if***
                ***end if***
            ***end if***
        ***end For***
    ***end for***
***end while***
Apply CLT-based K-means using output from FA
Compute cost function using optimal clusters from CLT-based K-means
***if*** $CLTKmeans.Cost <= BestSol.Cost$
    $BestSol.Cost = CLTKmeans.Cost$
***end if***
***end***

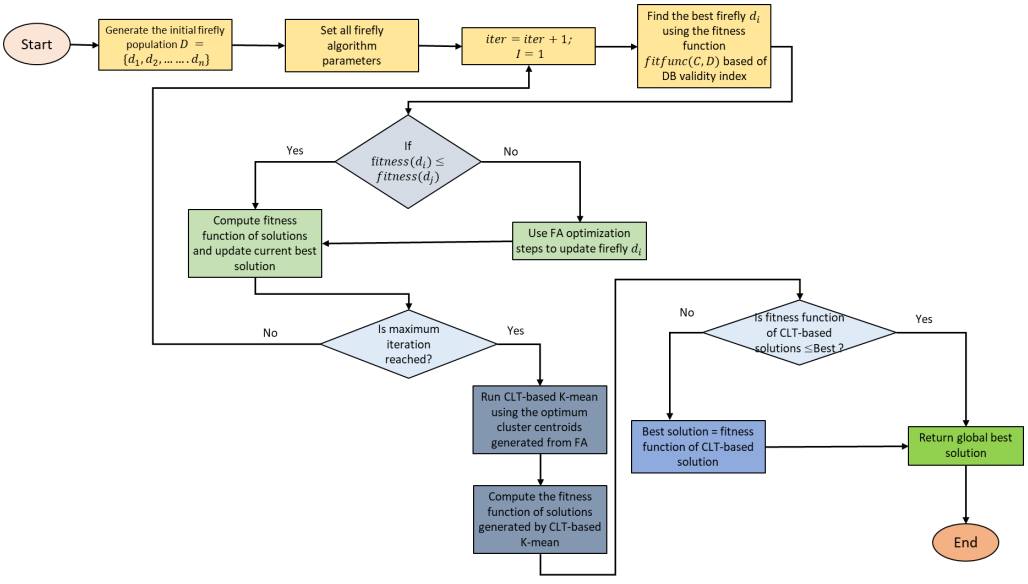


**Figure 1.** A flowchart for the proposed hybrid firefly algorithm-K-means clustering.

### 3.5. Davies Bouldin Index

The Davies Bouldin index (DBI) is a cluster validity index introduced by [34]. In the DBI, the quality of clustering is evaluated using the mean distances of data points from the centroid (intra-cluster distance) and the mean distances between the cluster centroids (inter-cluster distance) based on Equation (9).

$$Y_i = \left[\frac{1}{A_i}\sum_{C\in A_i} Z(P, c_i)^t\right]^{\frac{1}{t}} \tag{9}$$

where $Z(P,\ c_i)$ represents the distance between the data points in $A_i$ and its centroid $c_i$ and $t\ \geq 1$ represents an independently selected integer. The value $Y_i$ is the average Euclidean distance of the objects from the centroid with the cluster when the value of $t = 1$. However, when the value of $t\ =\ 2$, $Y_i$ represents the standard deviation of the objects' distances from the centroid within a cluster. If $L_{ij}$ represents the inter-cluster distance between two centroids $c_i$ and $c_j$, then

$$L_{ij} = Z(c_i, c_j), \qquad i\ \neq j \tag{10}$$

let $Z_i$ be defined as:

$$Z_i = max\left\{\frac{Y_i + Y_j}{L_{ij}}\middle| 1 \leq i,\ j \leq K,\ i \neq j\right\} \tag{11}$$

Then, the DBI is given as:

$$DB_{val}(A, Z) = \frac{1}{K}\sum_K^1 Z_i \tag{12}$$

where $K$ is the number of clusters.

### 3.6. Computational Complexity of the HFA-K-Means Algorithm

The proposed HFA-K-means clustering method is simple in terms of complexity, making it easy to implement for the problem at hand. The HFA-K-means is designed to retain the two inner loop structures of the classical FA when going through the population $n$ and one outer loop for evaluation in terms of the maximum number of function evaluations (MaxFE). So, the complexity in the extreme case is as of the standard FA, which is $O(n^2 MaxFE)$. Moreover, as n is small (typically the value of $n$ = 25 in our case), and MaxFE is large (MaxFE = 50,000), the computation cost is relatively inexpensive because the algorithm complexity is linear in terms of MaxFE. Therefore, considering the clustering objective, in this case, the algorithm's overall computational complexity is $O(n.K.MaxFE)$.

## 4. Experimentation, Performance Evaluation, and Discussion

This section discusses the experimental configuration of the proposed enhanced HFA-K-means with the parameter settings for its performance evaluation. The benchmark datasets used in validating the HFA-K-means are also described. The simulation results and discussion on the proposed algorithm are presented alongside its comparison with other results found in the literature.

### 4.1. System Configuration

Experiments were carried out using a 2.80 GHz Intel® Core™ i7-1165 processor with 8 GB RAM on the Windows 11 OS Version 21H2. The proposed algorithm was programmed in MATLAB R2020b and used the IBM SPSS statistics Version 25 to validate the experimental results' statistically significant differences.

### 4.2. Parameter Settings

The proposed HFA-K-means algorithm was set up using the following parameter settings: the population size is set using the established statistical tradition of a population size $\geq$ 30. Therefore, a population size of $n$ = 40 + 2k is adopted for the FA algorithm

and for the size of the dataset subsets used in the K-means phase of the HFA-K-means algorithm. The *MaxFE* (maximum function evaluation) is 50,000; mutation coefficient $m$ is 2; attraction coefficient $\beta$ is 2, the coefficient for light absorption $\gamma$ is 1; and the damping ratio $\alpha$ for the mutation coefficient is 1. The parameter configuration for the different algorithms is detailed in Table 1.

**Table 1.** Parameter Setting of Algorithms.

| GA [55,56] | | ABC [25] | | DE [25] | | PSO [25] | | FA [24] | | IWO [25] | | TLBO [55] | | SOS [57] | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Parameter** | **Value** | **Parameter** | **Value** | **Parameter** | **Value** | **Parameter** | **Value** | **Parameter** | **Value** | **Parameter** | **Value** | **Parameter** | **Value** | **Parameter** | **Value** |
| $n$ | 25 | $n$ | 25 | $n$ | 25 | $n$ | 25 | $n$ | 25 | $n$ | 25 | $n$ | 25 | $n$ | 25 |
| $\mathbf{P_c}$, $\mathbf{P_m}$ | 0.8, 0.3 | a | 0.009 | $CR_{min}$, $CR_{max}$ | 0.2, 1.0 | $W_1$, $W_2$ | 1.0, 0.99 | $\beta_0$ | 2 | s | 5 | m | 2 | $P_c$ | 0.2 |
| **MP** | 0.02 | #nOnlooker bees | 25 | F | 0.8 | $\tau_1$, $\tau_2$ | 1.5, 2.0 | $\gamma$ | 1 | $\Sigma_1$, $\Sigma_2$ | 0.5, 0.001 | | | Fl, Fu | 0.2, 0.8 |
| $\mathbf{K_{min}}$ | 2 | m | 2 | $K_{min}$ | 2 | $K_{min}$ | 2 | $K_{min}$ | 2 | $e$ | 2 | | | mr | 0.02 |
| $\mathbf{K_{max}}$ | 16 | MaxFE | 50,000 | $K_{max}$ | 256 | $K_{max}$ | 256 | $K_{max}$ | 256 | $K_{max}$ | 256 | | | | |
| **MaxFE** | 50,000 | | | MaxFE | 50,000 | MaxFE | 50,000 | MaxFE | 50,000 | MaxFE | 50,000 | MaxFE | 50,000 | MaxFE | 40 + 2k |

**Remarks**: $n$: population size; $P_c$: crossover probability; $P_m$: mutation probability; $W_1$: inertia weight; $W_2$: inertia weight damping ratio; $\tau_1$: personal learning coefficient; $\tau_2$: global learning coefficient; $K_{min}$ and $K_{max}$ denote, respectively, the maximum and minimum number of clusters that can be encoded in a single trial solution vector for GA, DE, PSO, FA and IWO; CR: crossover rate; F: scaling factor; Fl: lower bound of scaling factor; Fu: upper bound of scaling factor; mr: mutation rate; MP: mutation probability; $\beta_0$: attractiveness; $\gamma$: light absorption coefficient; $e$ = variance reduction exponent; $\Sigma_1$ = initial value of standard deviation; $\Sigma_2$ = final value of standard deviation; s = maximum number of seeds. Note that FADE [19], PSODE [47], IWODE [47], FAPSO [19], and FATLBO [19] use the same parameter representation scheme as their respective standard algorithms as provided in the given references.

### *4.3. Data Sets*

The characteristics of data samples, such as dimensionality, data distribution, and noise, significantly influence the performance of clustering algorithms. Given this, 20 datasets from different domains with various characteristics were used to investigate the proposed algorithm's efficiency. Specifically, among the datasets are 12 synthetically generated high-dimensional datasets, including three A-datasets (A1, A2, and A3), three Birch datasets (Birch1, Birch 2, and Birch 3), two DIM datasets (DIM002 and DIM1024) and four S-datasets (S1, S2, S3, and S4). The A-datasets and the Birch datasets are two-dimensional, with the A-datasets having varying clusters: 20, 35, and 50, respectively, while the Birch datasets have a uniform number of clusters which is 100, each with varying structures. The DIM 002 dataset has 15 dimensions with clusters, while the DIM1024 has 1024 dimensions with 16 clusters characterized by well-separated clusters. The S-datasets are spatial-distribution-based two-dimensional datasets with 15 clusters of varying complexity. Additionally, included among the datasets are two high-dimensional UCI datasets: Letter with 16 dimensions and 26 clusters, and Yeast with 8 dimensions and 10 clusters; two high-dimensional RGB images, Housec5 and Housec8, have 3 dimensions and 256 clusters, each having 34,112 data points. A two-dimensional shape set, D31, with 31 clusters was also included. Two-dimensional Mopsil-location datasets (Finland) with four clusters and one two-dimensional miscellaneous dataset with three clusters completed the number of datasets used. The summary of the 20 datasets is presented in Table 2.

**Table 2.** The characteristics of twenty high-dimensional datasets.

| Datasets | Dataset Types | Dimension of Data | Number of Data Objects | Number of Clusters | References |
|---|---|---|---|---|---|
| A1 | Synthetically generated | 2 | 3000 | 20 | [58–60] |
| A2 | Synthetically generated | 2 | 5250 | 35 | [58–60] |
| A3 | Synthetically generated | 2 | 7500 | 50 | [58–60] |
| Birch1 | Synthetically generated | 2 | 100,000 | 100 | [38,40,61] |
| Birch2 | Synthetically generated | 2 | 100,000 | 100 | [38,40,61] |
| Birch3 | Synthetically generated | 2 | 100,000 | 100 | [58,60,61] |
| Bridge | Grey-scale image blocks | 16 | 4096 | 256 | [60,62] |
| D31 | Shape sets | 2 | 3100 | 31 | [60,63] |
| Dim002 | Synthetically generated | 2–15 | 1351–10,126 | 9 | [58,60,64] |
| Dim1024 | High-dimensional | 1024 | 1024 | 16 | [58,60,65] |
| Housec5 | RGB image | 3 | 34,112 | 256 | [60,62] |
| Housec8 | RGB image | 3 | 34,112 | 256 | [60,66] |
| Letter | UCI dataset | 16 | 20,000 | 26 | [60,62] |
| Finland | Mopsi locations | 2 | 13,467 | 4 | [60,67] |
| S1 | Synthetically generated | 2 | 5000 | 15 | [58,60,68] |
| S2 | Synthetically generated | 2 | 5000 | 15 | [58,60,68] |
| S3 | Synthetically generated | 2 | 5000 | 15 | [58,60,68] |
| S4 | Synthetically generated | 2 | 5000 | 15 | [7,58,60,68] |
| T4.8k | Miscellaneous | 2 | 8000 | 3 | [60,69] |
| Yeast | UCI dataset | 8 | 1484 | 10 | [60,62] |

### *4.4. Metrics for Performance Evaluation*

At the first instance, experiments were conducted coupling seven other metaheuristic algorithms (ABC, IWO, SOS, TLBO, DE, PSO and genetic algorithm (GA)) with K-means using the same framework as the proposed HFA-K-means algorithm. Their performances were evaluated and compared with that of the proposed HFA-K-means algorithm over the 20 high-dimensional datasets using the DBI as the cluster validity index. This was to investigate the performance of the proposed HFA-K-means algorithm in comparison with these metaheuristic-based K-means hybrid algorithms in order to further justify its selection as the best option before its comparison with results from the literature. The experimental results of the performances of these seven metaheuristic-based K-means hybrid algorithms compared with the proposed HFA-K-means algorithm are presented in Table 3. The non-parametric Friedman mean rank test was also conducted to further validate the clustering solution of the various metaheuristic-based K-means hybrid algorithms. The results from the Friedman mean rank test are presented in Table 4.

The proposed HFA-K-means algorithm was compared with conventional metaheuristic search methods and other FA-based hybrid algorithms for a comprehensive and objective investigation of the clustering performance. The proposed algorithm was evaluated against classical FA, K-means, GA, DE, PSO, and IWO with four hybridized metaheuristics algorithms (FADE, PSODE, IWODE, Improved SOSK-means) and other FA variants (FAPSO, FATLBO, and FADE). The DBI was used to evaluate the proposed algorithm's performance over 20 high-dimensional datasets with dimensions between 9 and 256 and dataset sizes between 1000 and 100,000, as shown in Table 2.

**Table 3.** Computation results for Improved FA-K-means and other seven metaheuristic-based K-means hybrid algorithms on twenty high-dimensional datasets.

| Dataset | HFAK-Means | | | | HABCK-Means | | | | HIWOK-Means | | | | HSOSK-Means | | | | HTLBOK-Means | | | | HDEK-Means | | | | HPSOK-Means | | | | HGAK-Means | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Min | Max | Mean | Std. Dev | Min | Max | Mean | Std. Dev | Min | Max | Mean | Std. Dev | Min | Max | Mean | Std. Dev | Min | Max | Mean | Std. Dev | Min | Max | Mean | Std. Dev | Min | Max | Mean | Std. Dev | Min | Max | Mean | Std. Dev |
| A1 | 0.2441 | 0.3503 | 0.2942 | 0.0304 | 0.2892 | 0.6123 | 0.5794 | 0.0691 | 0.2387 | 0.6550 | 0.2964 | 0.0866 | 0.2519 | 0.3126 | **0.2768** | 0.0194 | 0.2424 | 0.5902 | 0.2983 | 0.0717 | 0.2638 | 0.3323 | 0.2853 | 0.0148 | 0.2512 | 0.5901 | 0.3017 | 0.0720 | 0.5902 | 0.6116 | 0.5944 | 0.0087 |
| A2 | 0.2186 | 0.4406 | 0.3155 | 0.0746 | 0.2865 | 0.6840 | 0.6594 | 0.0879 | 0.2445 | 0.4275 | 0.3027 | 0.0465 | 0.2530 | 0.4622 | 0.3344 | 0.0573 | 0.2198 | 0.5199 | 0.3407 | 0.0916 | 0.2349 | 0.6776 | 0.3595 | 0.1025 | 0.2226 | 0.4016 | **0.2954** | 0.0493 | 0.6776 | 0.6780 | 0.6778 | 0.0002 |
| A3 | 0.7908 | 0.7922 | 0.7915 | 0.0007 | 0.7796 | 0.7929 | **0.7881** | 0.0036 | 0.7730 | 0.9172 | 0.8402 | 0.0322 | 0.8124 | 0.8743 | 0.8406 | 0.0181 | 0.7908 | 0.7970 | 0.7932 | 0.0026 | 0.7908 | 0.7922 | 0.7912 | 0.0005 | 0.7922 | 0.8019 | 0.7994 | 0.0037 | 0.7909 | 0.8043 | 0.7967 | 0.0054 |
| Birch 1 | 0.7774 | 0.8018 | 0.8001 | 0.0053 | 0.7753 | 0.7965 | **0.7881** | 0.0059 | 0.8081 | 0.8996 | 0.8335 | 0.0204 | 0.8165 | 0.8577 | 0.8345 | 0.0143 | 0.8010 | 0.8018 | 0.8014 | 0.0003 | 0.8010 | 0.8018 | 0.8010 | 0.0002 | 0.8010 | 0.8025 | 0.8020 | 0.0005 | 0.8010 | 0.8034 | 0.8018 | 0.0006 |
| Birch 2 | 0.1585 | 0.2154 | 0.1880 | 0.0198 | 0.5070 | 0.5082 | 0.5074 | 0.0004 | 0.1617 | 0.2071 | **0.1852** | 0.0117 | 0.1511 | 0.2170 | 0.1886 | 0.0192 | 0.1630 | 0.2490 | 0.1972 | 0.0231 | 0.1740 | 0.2318 | 0.1977 | 0.0155 | 0.1471 | 0.2226 | 0.1909 | 0.0167 | 0.5070 | 0.5071 | 0.5070 | 0.0000 |
| Birch 3 | 0.6130 | 0.7160 | 0.6539 | 0.0421 | 0.7161 | 0.7438 | 0.7244 | 0.0082 | 0.5188 | 0.7973 | 0.6264 | 0.0747 | 0.4679 | 0.7413 | 0.6631 | 0.0690 | 0.5187 | 0.7139 | 0.6155 | 0.0578 | 0.4217 | 0.7160 | 0.6160 | 0.0882 | 0.4928 | 0.7179 | **0.6150** | 0.0722 | 0.7160 | 0.7660 | 0.7356 | 0.0223 |
| Bridge | 0.3289 | 0.3714 | **0.3489** | 0.0132 | 0.5202 | 0.9266 | 0.6549 | 0.1471 | 0.3110 | 1.0576 | 0.5210 | 0.1509 | 0.3168 | 0.6670 | 0.5014 | 0.1096 | 0.3064 | 0.5795 | 0.3667 | 0.0751 | 0.3044 | 0.5810 | 0.3589 | 0.0567 | 0.3216 | 0.6117 | 0.4277 | 0.0930 | 0.4650 | 0.6305 | 0.5930 | 0.0551 |
| D31 | 0.5773 | 0.7929 | **0.6923** | 0.0658 | 0.8017 | 0.8556 | 0.8271 | 0.0114 | 0.5385 | 0.8354 | 0.7251 | 0.0858 | 0.5906 | 0.8701 | 0.7374 | 0.0821 | 0.4945 | 0.8363 | 0.7056 | 0.0845 | 0.5503 | 0.7930 | 0.7134 | 0.0669 | 0.6315 | 0.8412 | 0.7290 | 0.0679 | 0.7930 | 0.8591 | 0.8236 | 0.0237 |
| Dim002 | 0.7309 | 0.7309 | 0.7309 | 0.0000 | 0.7160 | 0.7841 | 0.7563 | 0.0160 | 0.7309 | 0.8429 | 0.7862 | 0.0304 | 0.7046 | 0.9054 | 0.8395 | 0.0599 | 0.7309 | 0.7614 | 0.7552 | 0.0110 | 0.6872 | 0.7377 | **0.7293** | 0.0101 | 0.7309 | 0.7913 | 0.7547 | 0.0217 | 0.7311 | 0.7928 | 0.7586 | 0.0183 |
| Dim1024 | 0.3717 | 1.1511 | **1.0698** | 0.1783 | 1.8425 | 1.9023 | 1.8814 | 0.0166 | 0.8700 | 1.9619 | 1.8897 | 0.2403 | 0.3701 | 2.0337 | 1.8350 | 0.3987 | 1.4323 | 1.5769 | 1.5023 | 0.0306 | 1.4495 | 1.6282 | 1.5808 | 0.0541 | 0.7368 | 2.0528 | 1.9086 | 0.3991 | 1.7855 | 1.8308 | 1.8122 | 0.0137 |
| Housec5 | 0.2465 | 0.4808 | **0.3255** | 0.0572 | 0.5159 | 0.7018 | 0.6150 | 0.0622 | 0.2730 | 0.6019 | 0.3405 | 0.0671 | 0.2349 | 0.4649 | 0.3321 | 0.0669 | 0.2576 | 0.5400 | 0.3558 | 0.0927 | 0.2815 | 0.5557 | 0.4950 | 0.0528 | 0.2819 | 0.4987 | 0.3333 | 0.0464 | 0.4987 | 0.6445 | 0.6073 | 0.0643 |
| Housec8 | 0.3131 | 0.4720 | 0.4164 | 0.0460 | 0.3941 | 0.5715 | 0.5264 | 0.0376 | 0.3487 | 0.5184 | 0.4113 | 0.0435 | 0.3246 | 0.7094 | 0.4276 | 0.0874 | 0.2689 | 0.5209 | 0.3926 | 0.0587 | 0.3519 | 0.4874 | 0.4401 | 0.0438 | 0.3009 | 0.4781 | **0.4110** | 0.0422 | 0.4644 | 0.5212 | 0.5097 | 0.0232 |
| Letter | 0.7548 | 0.8157 | **0.7921** | 0.0231 | 0.8457 | 0.9814 | 0.9368 | 0.0359 | 0.6294 | 1.3885 | 1.1558 | 0.2253 | 0.2180 | 1.3460 | 0.9572 | 0.3256 | 0.3461 | 0.9189 | 0.8158 | 0.1167 | 0.7773 | 0.8305 | 0.8064 | 0.0146 | 0.7701 | 0.8880 | 0.8233 | 0.0346 | 0.8033 | 0.9284 | 0.8616 | 0.0261 |
| Finland | 0.2042 | 0.4789 | 0.3061 | 0.0910 | 0.2193 | 0.4994 | 0.4424 | 0.0553 | 0.1833 | 0.7142 | 0.3327 | 0.1364 | 0.2000 | 0.6692 | 0.3628 | 0.1282 | 0.2212 | 0.4397 | 0.3105 | 0.0724 | 0.1797 | 0.4393 | **0.2799** | 0.0784 | 0.2153 | 0.5731 | 0.3632 | 0.1305 | 0.4393 | 0.5766 | 0.4473 | 0.0305 |
| S1 | 0.6801 | 0.7767 | 0.7714 | 0.0215 | 0.7660 | 0.7903 | 0.7730 | 0.0057 | 0.6693 | 0.8565 | 0.8074 | 0.0403 | 0.6395 | 0.8380 | 0.8068 | 0.0428 | 0.7500 | 0.7767 | 0.7751 | 0.0060 | 0.7741 | 0.7767 | 0.7755 | 0.0011 | 0.5897 | 0.7969 | **0.7693** | 0.0425 | 0.7746 | 0.7973 | 0.7800 | 0.0053 |
| S2 | 0.5412 | 0.9898 | 0.7332 | 0.1215 | 0.7382 | 0.7700 | 0.7453 | 0.0067 | 0.5991 | 0.7928 | 0.7082 | 0.0699 | 0.6173 | 0.8288 | 0.7291 | 0.0629 | 0.4908 | 0.7470 | **0.7029** | 0.0729 | 0.6072 | 0.7391 | 0.7103 | 0.0510 | 0.5469 | 0.7470 | 0.7053 | 0.0574 | 0.7392 | 0.7470 | 0.7447 | 0.0030 |
| S3 | 0.3782 | 0.6038 | 0.5073 | 0.0641 | 0.7126 | 0.7305 | 0.7174 | 0.0047 | 0.3878 | 0.6352 | **0.4823** | 0.0599 | 0.3824 | 0.7527 | 0.4891 | 0.0773 | 0.3853 | 0.5973 | 0.4883 | 0.0529 | 0.3868 | 0.6838 | 0.4993 | 0.0727 | 0.3863 | 0.5955 | 0.4846 | 0.0600 | 0.7112 | 0.7305 | 0.7162 | 0.0063 |
| S4 | 0.6802 | 0.7690 | 0.7645 | 0.0198 | 0.7684 | 0.7810 | 0.7761 | 0.0032 | 0.6714 | 0.8334 | 0.7962 | 0.0329 | 0.6784 | 0.8170 | 0.7928 | 0.0330 | 0.6951 | 0.7699 | **0.7641** | 0.0167 | 0.7690 | 0.7696 | 0.7691 | 0.0002 | 0.7229 | 0.7898 | 0.7726 | 0.0152 | 0.7693 | 0.7829 | 0.7773 | 0.0033 |
| T4.8k | 0.0178 | 0.0227 | 0.0218 | 0.0020 | 0.0213 | 0.0427 | 0.0300 | 0.0067 | 0.0178 | 0.3229 | 0.1248 | 0.0990 | 0.0179 | 0.3341 | 0.1582 | 0.0783 | 0.0176 | 0.0227 | 0.0220 | 0.0017 | 0.0182 | 0.0227 | 0.0220 | 0.0016 | 0.0179 | 0.0227 | **0.0215** | 0.0020 | 0.0218 | 0.0227 | 0.0226 | 0.0002 |
| Yeast | 0.4379 | 0.5559 | 0.5205 | 0.0570 | 0.5882 | 0.9583 | 0.8232 | 0.1141 | 0.2335 | 0.5569 | 0.5286 | 0.0784 | 0.7694 | 0.9057 | 0.8665 | 0.0300 | 0.2248 | 0.7931 | 0.6228 | 0.1514 | 0.2323 | 0.5660 | **0.4892** | 0.0745 | 0.2021 | 0.7869 | 0.5539 | 0.1992 | 0.2986 | 0.8131 | 0.6637 | 0.1409 |
| Overall average | 0.4533 | 0.6164 | **0.5522** | 0.0467 | 0.6402 | 0.7717 | 0.7276 | 0.0349 | 0.4604 | 0.7911 | 0.6347 | 0.0816 | 0.4409 | 0.7803 | 0.6487 | 0.0890 | 0.4678 | 0.6776 | 0.5813 | 0.0545 | 0.5028 | 0.6581 | 0.5860 | 0.0400 | 0.4581 | 0.7005 | 0.6031 | 0.0713 | 0.6689 | 0.7424 | 0.7116 | 0.0225 |

**Table 4.** Friedman mean rank for the eight metaheuristic-based K-mean hybrid algorithms.

| Metaheuristic-Based K-Means Hybrid Algorithm | Mean Rank |
|---|---|
| **HFAK-means** | 2.48 |
| **HTLBOK-means** | 3.21 |
| **HDEK-means** | 3.36 |
| **HPSOK-means** | 3.67 |
| **HIWOK-means** | 4.90 |
| **HSOSK-means** | 5.62 |
| **HABCK-means** | 6.29 |
| **HGAK-means** | 6.48 |

The summary of the results obtained by the proposed HFA-K-means for the clustering problem over 20 high-dimensional datasets is presented in Table 5. The results are presented to show the best, worst, average, and standard deviation values over 20 independent runs of each of the algorithms on 20 datasets. This signifies the optimal, the worst, the mean, and the standard deviation of the clustering solutions. The average computation time taken to achieve the optimal solution for the HFA-K-means is shown in Figure 2. To investigate the performance gain in terms of the computation time, the HFA-K-means computation time is compared with the computation time of a standard hybrid of FA and K-means. This is presented in Figure 3. The computation results of the HFA-K-means are compared with the classical FA and K-means algorithms. The summary of the results obtained by the proposed HFA-K-means compared with the classical FA and K-means algorithm for the problem at hand over 20 high-dimensional datasets is presented in Table 6.

**Table 5.** Computation result for Improved FA-K-means on twenty high-dimensional datasets.

| High-Dimensional Datasets | DB Index | | | |
|---|---|---|---|---|
| | **Best** | **Worst** | **Mean** | **Std. Dev** |
| A1 | 0.2441 | 0.3503 | 0.2942 | 0.0304 |
| A2 | 0.2186 | 0.4406 | 0.3155 | 0.0746 |
| A3 | 0.7908 | 0.7922 | 0.7915 | 0.0007 |
| Birch1 | 0.7774 | 0.8018 | 0.8001 | 0.0053 |
| Birch2 | 0.1585 | 0.2154 | 0.1880 | 0.0198 |
| Birch3 | 0.6130 | 0.7160 | 0.6539 | 0.0421 |
| Bridge | 0.3289 | 0.3714 | 0.3489 | 0.0132 |
| D31 | 0.5773 | 0.7929 | 0.6923 | 0.0658 |
| Dim002 | 0.7309 | 0.7309 | 0.7309 | 0.0000 |
| Dim1024 | 0.3717 | 1.1511 | 1.0698 | 0.1783 |
| Housec5 | 0.2465 | 0.4808 | 0.3255 | 0.0572 |
| Housec8 | 0.3131 | 0.4720 | 0.4164 | 0.0460 |
| Letter | 0.7548 | 0.8157 | 0.7921 | 0.0231 |
| Finland | 0.2042 | 0.4789 | 0.3061 | 0.0910 |
| S1 | 0.6801 | 0.7767 | 0.7714 | 0.0215 |
| S2 | 0.5412 | 0.9898 | 0.7332 | 0.1215 |
| S3 | 0.3782 | 0.6038 | 0.5073 | 0.0641 |
| S4 | 0.6802 | 0.7690 | 0.7645 | 0.0198 |
| T4.8k | 0.0178 | 0.0227 | 0.0218 | 0.0020 |
| Yeast | 0.4379 | 0.5559 | 0.5205 | 0.0570 |
| Average | 0.4544 | 0.6164 | 0.5522 | 0.0464 |

The performance of the HFA-K-means is also compared with results from the literature. The mean and standard deviation metrics from experiments in this study and that available in the literature are used for the comparisons. There are three categories of comparisons with results from the literature. The first category compares with classical metaheuristics algorithms GA, DE, PSO, and IWO. This is presented in Table 7. The second category compares HFA-K-means with other hybrid metaheuristics algorithms: ISOSK-means, PSODE, FADE, and IWODE, as shown in Table 8. A comparison with other FA-based hybrid meta-

heuristics algorithms, FADE, FAPSO, and FATLBO, is considered in the third category. This can be seen in Table 9.

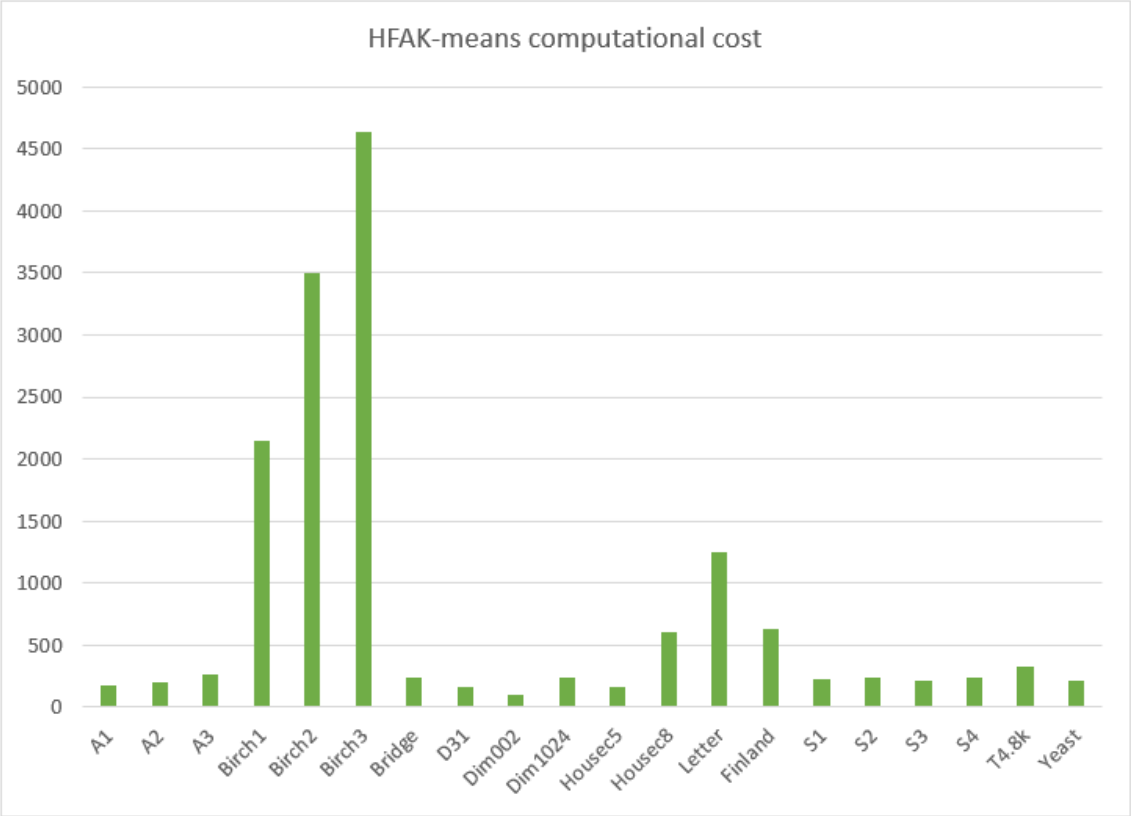


**Figure 2.** Mean execution run time of HFA-K-means on DBI for 20 high-dimensional datasets over 20 runs.

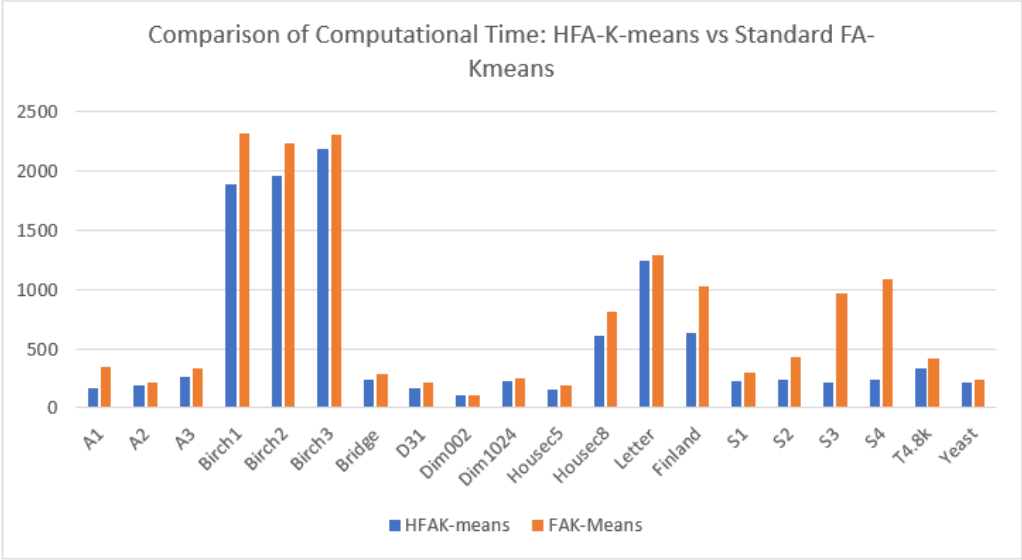


**Figure 3.** Mean execution run time of HFA-K-means and standard FA-K-means on DBI for 20 high-dimensional datasets over 20 runs.

**Table 6.** Comparison of HFA-K-means with FA and K-means over 20 independent program executions.

| High-Dimensional Datasets | HFA-K-Means | | | | FA | | | | K-Means | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | **Best** | **Worst** | **Mean** | **Std. Dev** | **Best** | **Worst** | **Mean** | **Std. Dev** | **Best** | **Worst** | **Mean** | **Std. Dev** |
| A1 | 0.2441 | 0.3503 | **0.2942** | 0.0304 | 0.5901 | 0.5901 | 0.5901 | 0.0000 | 0.7788 | 0.7788 | 0.7788 | 0.0000 |
| A2 | 0.2186 | 0.4406 | **0.3155** | 0.0746 | 0.6776 | 0.6776 | 0.6776 | 0.0000 | 0.9229 | 0.9229 | 0.9229 | 0.0000 |
| A3 | 0.7908 | 0.7922 | 0.7915 | 0.0007 | 0.7908 | 0.7922 | 0.7915 | 0.0007 | 1.2433 | 1.2793 | 1.2631 | 0.0184 |
| Birch1 | 0.7774 | 0.8018 | **0.8001** | 0.0053 | 0.8010 | 0.8018 | 0.8013 | 0.0003 | 1.2875 | 1.2875 | 1.2875 | 0.0000 |
| Birch2 | 0.1585 | 0.2154 | **0.1880** | 0.0198 | 0.5070 | 0.5070 | 0.5070 | 0.0000 | 0.5882 | 0.5882 | 0.5882 | 0.0000 |
| Birch3 | 0.6130 | 0.7160 | **0.6539** | 0.0421 | 0.7160 | 0.7160 | 0.7160 | 0.0000 | 1.1592 | 1.1726 | 1.1650 | 0.0037 |
| Bridge | 0.3289 | 0.3714 | **0.3489** | 0.0132 | 0.6108 | 0.6116 | 0.6113 | 0.0003 | 0.9876 | 0.9876 | 0.9876 | 0.0000 |
| D31 | 0.5773 | 0.7929 | **0.6923** | 0.0658 | 0.7929 | 0.7929 | 0.7929 | 0.0000 | 1.3446 | 1.3446 | 1.3446 | 0.0000 |
| Dim002 | 0.7309 | 0.7309 | 0.7309 | 0.0000 | 0.7309 | 0.7309 | 0.7309 | 0.0000 | 1.1535 | 1.1535 | 1.1535 | 0.0000 |
| Dim1024 | 0.3717 | 1.1511 | **1.0698** | 0.1783 | 0.9489 | 1.1511 | 1.1106 | 0.0625 | 3.4291 | 3.4291 | 3.4291 | 0.0000 |
| Housec5 | 0.2465 | 0.4808 | **0.3255** | 0.0572 | 0.4987 | 0.6422 | 0.5561 | 0.0721 | 0.8678 | 0.8678 | 0.8678 | 0.0000 |
| Housec8 | 0.3131 | 0.4720 | **0.4164** | 0.0460 | 0.4644 | 0.5209 | 0.5068 | 0.0251 | 0.6395 | 0.6395 | 0.6395 | 0.0000 |
| Letter | 0.7548 | 0.8157 | 0.7921 | 0.0231 | 0.7548 | 0.8157 | 0.7921 | 0.0231 | 2.0355 | 2.0360 | 2.0359 | 0.0002 |
| Finland | 0.2042 | 0.4789 | **0.3061** | 0.0910 | 0.4393 | 0.4393 | 0.4393 | 0.0000 | 0.5461 | 0.5461 | 0.5461 | 0.0000 |
| S1 | 0.6801 | 0.7767 | **0.7714** | 0.0215 | 0.7743 | 0.7767 | 0.7762 | 0.0010 | 1.2048 | 1.3463 | 1.3044 | 0.0455 |
| S2 | 0.5412 | 0.9898 | **0.7332** | 0.1215 | 0.7391 | 0.7391 | 0.7391 | 0.0000 | 1.1230 | 1.1230 | 1.1230 | 0.0000 |
| S3 | 0.3782 | 0.6038 | **0.5073** | 0.0641 | 0.7111 | 0.7111 | 0.7111 | 0.0000 | 1.0750 | 1.0750 | 1.0750 | 0.0000 |
| S4 | 0.6802 | 0.7690 | **0.7645** | 0.0198 | 0.7690 | 0.7771 | 0.7694 | 0.0018 | 1.2354 | 1.2354 | 1.2354 | 0.0000 |
| T4.8k | 0.0178 | 0.0227 | **0.0218** | 0.0020 | 0.0227 | 0.0227 | 0.0227 | 0.0000 | 0.9649 | 0.9649 | 0.9649 | 0.0000 |
| Yeast | 0.4379 | 0.5559 | 0.5205 | 0.0570 | 0.4379 | 0.5559 | 0.5205 | 0.0570 | 0.7084 | 1.7844 | 1.7257 | 0.2395 |
| Average | 0.4544 | 0.6164 | **0.5522** | 0.0464 | 0.6389 | 0.6686 | 0.6581 | 0.0122 | 1.1647 | 1.2281 | 1.2219 | 0.0154 |

**Table 7.** Comparison of HFA-K-means with GA, DE, PSO, and IWO on twenty datasets.

| High-Dimensional Datasets. | HFA-K-Means | | GA | | DE | | PSO | | IWO | |
|---|---|---|---|---|---|---|---|---|---|---|
| | **Mean** | **Std. Dev** | **Mean** | **Std. Dev** | **Mean** | **Std. Dev** | **Mean** | **Std. Dev** | **Mean** | **Std. Dev** |
| A1 | **0.2942** | 0.0304 | 0.6907 | 0.0420 | 0.6016 | 0.0116 | 0.6662 | 0.0424 | 0.6308 | 0.0265 |
| A2 | **0.3155** | 0.0746 | 0.7549 | 0.0293 | 0.7081 | 0.0348 | 0.7134 | 0.0284 | 0.7483 | 0.0468 |
| A3 | 0.7915 | 0.0007 | 0.7758 | 0.0418 | 0.7349 | 0.0184 | **0.7279** | 0.0287 | 0.7545 | 0.0339 |
| Birch1 | 0.8001 | 0.0053 | 0.7976 | 0.0309 | **0.7480** | 0.0171 | 0.7528 | 0.0169 | 0.7644 | 0.0211 |
| Birch2 | **0.1904** | 0.0174 | 0.6197 | 0.0361 | 0.5086 | 0.0016 | 0.5876 | 0.0358 | 0.5168 | 0.0052 |
| Birch3 | **0.6577** | 0.0647 | 0.7718 | 0.0431 | 0.7488 | 0.0297 | 0.7213 | 0.0306 | 0.7568 | 0.0248 |
| Bridge | **0.3489** | 0.0132 | - | - | 0.8292 | 0.0516 | 0.9786 | 0.1331 | 1.1575 | 0.0967 |
| D31 | **0.6923** | 0.0658 | - | - | 0.8757 | 0.0302 | 0.7630 | 0.0514 | 0.7896 | 0.0312 |
| Dim002 | 0.7309 | 0.0000 | 0.6772 | 0.0664 | 0.6685 | 0.0263 | 0.5823 | 0.0772 | **0.6661** | 0.0403 |
| Dim1024 | **1.0698** | 0.1783 | - | - | 1.7678 | 0.0114 | 1.8224 | 0.0481 | 2.0105 | 0.0193 |
| Housec5 | **0.3255** | 0.0572 | - | - | 0.6190 | 0.0412 | 0.7456 | 0.0679 | 0.7034 | 0.0430 |
| Housec8 | **0.4164** | 0.0460 | - | - | 0.5245 | 0.0139 | 0.6418 | 0.0858 | 0.6206 | 0.0546 |
| Letter | **0.7921** | 0.0231 | - | - | 0.9852 | 0.0303 | 2.0354 | 0.1084 | 1.2245 | 0.0416 |
| Finland | **0.3061** | 0.0910 | 0.4407 | 0.0009 | 0.4575 | 0.0082 | 0.5761 | 0.0630 | 0.5036 | 0.0398 |
| S1 | 0.7714 | 0.0215 | **0.7006** | 0.0293 | 0.7363 | 0.0150 | 0.6472 | 0.0470 | 0.7572 | 0.0315 |
| S2 | 0.7332 | 0.1215 | 0.7417 | 0.0393 | 0.7256 | 0.0221 | **0.6878** | 0.0318 | 0.7625 | 0.0296 |
| S3 | **0.5073** | 0.0641 | 0.7682 | 0.0395 | 0.7265 | 0.0179 | 0.7261 | 0.0282 | 0.7567 | 0.0333 |
| S4 | 0.7645 | 0.0198 | 0.7663 | 0.0240 | 0.7635 | 0.0180 | **0.7455** | 0.0316 | 0.7809 | 0.0191 |
| T4.8k | **0.0218** | 0.0020 | 0.0023 | 0.0000 | 0.0228 | 0.0001 | 17.5004 | 22.6863 | 0.1119 | 0.0665 |
| Yeast | **0.5205** | 0.0570 | - | - | 0.8053 | 0.0401 | 0.7710 | 0.1219 | 0.5700 | 0.1055 |
| Average | 0.5526 | 0.0474 | 0.6544 | 0.0325 | 0.7279 | 0.0220 | 1.6696 | 1.1882 | 0.7793 | 0.0405 |

**Table 8.** Comparison of HFA-K-means with ISOS-K-means, PSODE, FADE, and IWODE on 20 high-dimensional datasets.

| High-Dimensional Datasets | HFA-K-Means | | ISOSK-Means | | PSODE | | FADE | | IWODE | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Mean | Std. Dev | Mean | Std. Dev | Mean | Std. Dev | Mean | Std. Dev | Mean | Std. Dev |
| A1 | **0.2942** | 0.0304 | 0.5911 | 0.0003 | 0.5949 | 0.0086 | 0.6171 | 0.0347 | 0.6525 | 0.0621 |
| A2 | **0.3155** | 0.0746 | 0.6781 | 0.0002 | 0.6912 | 0.0161 | 0.6976 | 0.0215 | 0.7296 | 0.0391 |
| A3 | 0.7915 | 0.0007 | 0.7945 | 0.0011 | 0.7106 | 0.0176 | **0.7085** | 0.0332 | 0.7527 | 0.0319 |
| Birch1 | 0.8001 | 0.0053 | 0.8030 | 0.0006 | 0.7256 | 0.0276 | **0.7232** | 0.0257 | 0.7692 | 0.0279 |
| Birch2 | **0.1904** | 0.0174 | 0.5071 | 0.0001 | 0.5070 | 0.0002 | 0.5155 | 0.0235 | 0.5176 | 0.0084 |
| Birch3 | **0.6577** | 0.0647 | 0.7168 | 0.0004 | 0.7074 | 0.0151 | 0.7012 | 0.0191 | 0.7570 | 0.0247 |
| Bridge | **0.3489** | 0.0132 | 0.6464 | 0.0009 | 0.7141 | 0.1007 | 0.6405 | 0.0709 | 1.1397 | 0.0666 |
| D31 | **0.6923** | 0.0658 | 0.8125 | 0.0070 | 0.8021 | 0.0407 | 0.7788 | 0.0376 | 0.7972 | 0.0514 |
| Dim002 | 0.7309 | 0.0000 | 0.6384 | 0.0222 | 0.5975 | 0.0445 | **0.6280** | 0.0607 | 0.6705 | 0.0432 |
| Dim1024 | **1.0698** | 0.1783 | 1.1332 | 0.3795 | 1.7644 | 0.0112 | 1.4759 | 0.1200 | 1.9654 | 0.0261 |
| Housec5 | **0.3255** | 0.0572 | 0.5377 | 0.0123 | 2.2408 | 4.0861 | 0.5467 | 0.0287 | 0.6865 | 0.0229 |
| Housec8 | **0.4164** | 0.0460 | 0.4919 | 0.5305 | 0.5022 | 0.0315 | 0.4707 | 0.0383 | 0.6344 | 0.0408 |
| Letter | **0.7921** | 0.0231 | 0.9683 | 1.0545 | 0.9121 | 0.0628 | 0.8665 | 0.0663 | 1.2057 | 0.0571 |
| Finland | **0.3061** | 0.0910 | 0.4427 | 0.0011 | 0.4465 | 0.0060 | 0.4686 | 0.0547 | 0.4864 | 0.0371 |
| S1 | 0.7714 | 0.0215 | 0.7770 | 0.0009 | **0.6739** | 0.0351 | 0.6756 | 0.0270 | 0.7501 | 0.0280 |
| S2 | 0.7332 | 0.1215 | 0.7412 | 0.0008 | **0.6844** | 0.0280 | 0.6939 | 0.0345 | 0.7556 | 0.0190 |
| S3 | **0.5073** | 0.0641 | 0.7126 | 0.0005 | 0.7106 | 0.0199 | 0.7072 | 0.0181 | 0.7559 | 0.0317 |
| S4 | 0.7645 | 0.0198 | 0.7723 | 0.0009 | **0.7299** | 0.0162 | 0.7356 | 0.0226 | 0.7896 | 0.0303 |
| T4.8k | **0.0218** | 0.0020 | 0.0227 | 0.0000 | 0.0227 | 0.0000 | 0.0423 | 0.0882 | 0.0928 | 0.0515 |
| Yeast | **0.5205** | 0.0570 | 0.7489 | 0.0682 | 0.7193 | 0.0677 | 0.6375 | 0.1344 | 0.5949 | 0.1144 |
| Average | **0.5526** | 0.0474 | 0.6768 | 0.1041 | 0.7729 | 0.2318 | 0.6665 | 0.0480 | 0.7752 | 0.0407 |

**Table 9.** Comparison of HFA-K-means with FA-based hybrid algorithms: FAPSO, FADE, and FATLBO on seven high-dimensional datasets.

| High-Dimensional Datasets | HFA-K-Means | FAPSO | FADE | FATLBO |
|---|---|---|---|---|
| | Mean Value | Mean Value | Mean Value | Mean Value |
| Birch1 | 0.8001 | **0.6572** | 0.7232 | 0.6952 |
| Birch2 | **0.1904** | 0.5040 | 0.5155 | 0.5163 |
| Birch3 | **0.6577** | 0.6812 | 0.7012 | 0.7596 |
| Bridge | **0.3489** | 0.5901 | 0.6405 | 0.6108 |
| Housec5 | **0.3255** | 0.4187 | 2.2408 | 0.4158 |
| Housec8 | **0.4164** | 0.4245 | 0.5022 | 0.4559 |
| Letter | 0.7921 | **0.7146** | 0.9121 | 0.7775 |
| Average | 0.5046 | 0.5700 | 0.8908 | 0.6044 |

In comparing the performance of the proposed HFA-K-means with other algorithms, the solution quality determined by the DBI, which was used as the clustering objective during the program execution, was used. The clustering solutions are recorded using four decimal place values, and optimal clustering solutions are marked in boldface to indicate which algorithm produced the optimal clustering solution for each dataset.

For validating the significant difference between the clustering results produced by the separate algorithms statistically, the Friedman statistical test with the Wilcoxon post hoc test was performed. The results of both tests are presented and recorded in Tables 10 and 11, respectively.

**Table 10.** Friedman rank-sum test for the HFA-K-means and the two classical algorithms tested across the twenty datasets for the twenty algorithm runs.

| Datasets | FA | K-Means | HFA-K-Means |
|---|---|---|---|
| A1 | 2.00 | 3.00 | 1.00 |
| A2 | 2.00 | 3.00 | 1.00 |
| A3 | 1.50 | 3.00 | 1.50 |
| Birch1 | 1.53 | 3.00 | 1.48 |
| Birch2 | 2.00 | 3.00 | 1.00 |
| Birch3 | 1.85 | 3.00 | 1.15 |
| Bridge | 2.00 | 3.00 | 1.00 |
| D31 | 1.95 | 3.00 | 1.05 |
| Dim002 | 1.50 | 3.00 | 1.50 |
| Dim1024 | 1.55 | 3.00 | 1.45 |
| Housec5 | 1.55 | 3.00 | 1.45 |
| Housec8 | 1.95 | 3.00 | 1.05 |
| Letter | 1.50 | 3.00 | 1.50 |
| Finland | 1.83 | 3.00 | 1.18 |
| S1 | 1.53 | 3.00 | 1.48 |
| S2 | 1.60 | 3.00 | 1.40 |
| S3 | 2.00 | 3.00 | 1.00 |
| S4 | 1.53 | 3.00 | 1.48 |
| T4.8k | 2.00 | 3.00 | 1.00 |
| Yeast | 1.50 | 3.00 | 1.50 |

**Table 11.** Wilcoxon signed-rank test $p$ values for equal medians on DB Index.

| Datasets | HFA-K-Means vs. FA | HFA-K-Means vs. K-Means | FA vs. K-Means |
|---|---|---|---|
| A1 | 0.001 | 0.001 | 0.001 |
| A2 | 0.001 | 0.001 | 0.001 |
| A3 | 1.000 | 0.001 | 0.001 |
| Birch1 | 0.317 | 0.001 | 0.001 |
| Birch2 | 0.001 | 0.001 | 0.001 |
| Birch3 | 0.002 | 0.001 | 0.001 |
| Bridge | 0.001 | 0.001 | 0.001 |
| D31 | 0.001 | 0.001 | 0.001 |
| Dim002 | 1.000 | 0.001 | 0.001 |
| Dim1024 | 0.180 | 0.001 | 0.001 |
| Housec5 | 0.180 | 0.001 | 0.001 |
| Housec8 | 0.001 | 0.001 | 0.001 |
| Letter | 1.000 | 0.001 | 0.001 |
| Finland | 0.001 | 0.001 | 0.001 |
| S1 | 0.317 | 0.001 | 0.001 |
| S2 | 0.638 | 0.001 | 0.001 |
| S3 | 0.001 | 0.001 | 0.001 |
| S4 | 0.317 | 0.001 | 0.001 |
| T4.8k | 0.001 | 0.001 | 0.001 |
| Yeast | 1.000 | 0.001 | 0.001 |

### *4.5. Results and Discussion*

The experimental results presented in Table 3 indicate that the proposed HFA-K-means algorithm had the lowest mean clustering value of 0.5522 for all the 20 high-dimensional datasets. This indicates that the proposed algorithm averagely performed better compared to the other seven metaheuristic-based K-means algorithms. The proposed HFA-K-means algorithm and PSO-based K-means recorded best results in five of the datasets each while the best performance records for the other ten datasets were shared among the other competing K-means-based hybrid algorithms. The Friedman mean rank test result of the eight competing algorithms presented in Table 4 also shows that the proposed HFAK-means

algorithm ranked better with the lowest mean rank of 2.48 to further justify its superior performance over the PSO-based K-means that recorded a mean rank of 3.67.

The computation results shown in Table 5 show that the proposed HFA-K-means can cluster high-dimensional datasets efficiently. The high convergence rate of the HFA-K-means is reflected in Figure 2, showing the average computation time required to achieve convergence. In comparison with the classical FA and K-means algorithms, as shown in Table 4, the proposed HFA-K-means recorded the best performance in 16 of the 20 datasets, namely, A1, A2, Birch1, Birch2, Birch3, Bridge, D31, DIM1024, Housec5, Housec8, Finland, S1, S2, S3, S4, and T4.8k. HFA-K-means recorded ties with FA in the remaining ones, namely A3, DIM002, Letter, and Yeast. In terms of average performance across all the 20 datasets, the HFA-K-means recorded the lowest value, followed by the FA. The lesser values of HFA-K-means compared with FA and K-means in most datasets show better performance. Based on this, we can infer that HFA-K-means can effectively and efficiently automatically cluster high-dimensional datasets.

Table 7 compares the proposed algorithm with other classical metaheuristic algorithms. The results show that HFA-K-means demonstrated the best performance in f14 datasets, namely, A1, A2, Birch2, Birch3, Bridge, D31, DIM1024, Housec5, Housec8, Letter, Finland, S3, T4.8k, and Yeast. PSO outperformed the HFA-K-means in three datasets, namely A3, S2, and S4. The remaining algorithms, GA, DE, and IWO, outperformed the HFA-K-means in one dataset, namely S1, Birch1, and DIM002, respectively. HFA-K-means had the minimum average clustering results compared with the classical metaheuristic algorithms. This shows that the proposed HFA-K-means demonstrates superior performance over all the other competing metaheuristic algorithms.

Table 8 compares the HFA-K-means with other hybrid algorithms, namely ISOS-K-means, PSODE, FADE, and IWODE. From the table, HFA-K-means also shows the best optimal solutions in 14 datasets, namely, A1, A2, Birch2, Birch3, Bridge, D31, DIM1024, Housec5, Housec8, Letter, Finland, S3, T4.8k, and Yeast. PSODE recorded a better performance in S1, S2, and S4, while FADE performed better on the A3, Birch1, and DIM002 datasets. The minimum average clustering result across the 20 datasets was recorded by HFA-K-means followed by FADE, ISOSK-means, PSODE, and IWODE. This indicates that HFA-K-means had performance superiority compared with other competing hybrid metaheuristic algorithms.

The HFA-K-means algorithm was also compared with three other FA-based hybrid metaheuristic algorithms, FAPSO, FADE, and FATLBO, on seven high-dimensional datasets, as shown in Table 8. From the results, the HFA-K-means recorded superior performance in five of the seven datasets, namely, Birch2, Birch3, Bridge, Housec5, and Housec8. FAPSO demonstrated better performance in the Birch 1 and Letter datasets. HFA-K-means again recorded the smallest average clustering solution across the seven datasets, followed by FAPSO. Based on this, it can be stated that on the basis of comparison of all the FA-based hybrid metaheuristic algorithms, it is evident that the HFA-K-means had superior performance.

Furthermore, in terms of comparison of HFA-K-means with the K-means-based algorithm ISOSK-means, it can be seen from Table 8 that HFA-K-means outperformed its counterpart in all except one of the datasets, namely, DIM002. This shows that the improvement in the K-means contributes substantially to the superior performances recorded by the HFA-K-means algorithm.

In Figure 3, the performance gain in the computation time using the CLT-based K-means in the proposed hybrid algorithm is clearly demonstrated against the regular hybrid of FA and K-means. The HFA-K-means recorded a lower computation time to achieve convergence than its standard FA-K-means counterpart in all 20 datasets.

### *4.6. Statistical Analysis Tests*

To further validate the results of the clustering solutions of the proposed HFA-K-means, a nonparametric test was performed using the Friedman rank-sum test. This test identifies significant differences and was used in this study to draw statistically meaningful con-

clusions about the performance claims of the proposed algorithm reported earlier in the previous section. The report of the Friedman rank-sum test for the HFA-K-means and the two classical algorithms tested across the 20 datasets for the 20 algorithm runs is presented in Table 10. The report shows that the HFA-K-means had the best performance having the lowest rank value of 1 and highest rank value of 1.5 compared to FA and K-Means. The FA recorded a minimum rank value of 1.53 and a maximum rank value of 2.0, while the K-means had its minimum and maximum rank values as 3.0.

An additional post hoc test was performed on the Friedman rank-sum test using the Wilcoxon signed-rank test to verify the significant differences between the algorithms. The Wilcoxon signed-rank test presents $p$ values for determining the significant differences between the competing algorithms. The $p$-values were tested against the significant value of 0.05, meaning that if the $p$-value is larger than 0.05, there is no significant difference between the two algorithms. The results of the Wilcoxon signed-rank test presented in Table 11 clearly show significant differences in the performance of the representative algorithms. From the report, it can be clearly stated that HFA-K-means achieved a significant performance improvement in clustering high-dimensional datasets over the classical K-means algorithm in all the datasets and over the FA in 11 of the 20 datasets.

## 5. Conclusions

In this study, a new HFA-K-means algorithm has been proposed and successfully implemented to cluster high-dimensional datasets automatically. The idea of the CLT was incorporated into the K-means phase of the hybrid algorithm to reduce the number of distance calculations required by the K-means algorithm. A mutation probability was also introduced into the FA phase to improve the exploitation and exploration of the FA. An initial investigative experiment conducted using another seven metaheuristic-based K-means hybrid algorithms modelled using the same framework as the proposed algorithm further confirmed the superior performance of the FA for automatic clustering, thus justifying its selection among other metaheuristic algorithms. The non-parametric mean rank test conducted on the experimental results of the eight metaheuristic-based K-means hybrid algorithms provided a further justification for the selection of FA among the other metaheuristic algorithms. The results obtained from the other computation experiments over the 20 high-dimensional datasets clearly show that the HFA-K-means algorithm demonstrates outstanding performance compared with the two classical algorithms, FA and K-means. Statistical tests were conducted on the simulation results to confirm the performance of the proposed hybrid algorithm, namely, the Friedman mean rank test and Wilcoxon rank-sum (a post hoc) test. A comparison of the results with other metaheuristics algorithms from the literature (GA, DE, PSO, and IWO) and other similar hybrid algorithms (SOSK-means, PSODE, FADE, and IWODE) was also performed. The proposed algorithm recorded superior results in 13 (65%) out of the 20 datasets compared with the other metaheuristics and hybrid metaheuristic algorithms. The numerical results were also compared with different FA-based hybrid algorithms from the literature, namely, FAPSO, FADE, and FATLBO, on seven high-dimensional datasets. The results showed that the proposed algorithm had superior performance in five of the seven datasets in terms of the quality of clustering solutions obtained. This indicates that the HFA-K-means performed well in 71% of the total datasets used for the comparison. The computational time for the proposed algorithm was also compared with regular FAK-means, and there was a substantial reduction in the computation time of the proposed HFAK-means over all the 20 high-dimensional datasets. These confirm that the adoption of the CLT in the K-means phase of the proposed algorithm assists in the reduction in the computation time, thereby enhancing the performance of the proposed algorithm in terms of reduced computational cost. The modified CLT-based K-means can be substituted for the standard K-means algorithm in many of the existing applications where it has been used to improve the clustering performance in various domains. For future study, the modified CLT-based K-means can be combined with other metaheuristic algorithms for respective hybridization models. Additionally, a further com-

putation time reduction in the FA phase can be explored by incorporating Levy flights into the algorithm update phase to increase the fireflies' movement, thereby reducing the foraging time of the fireflies. Moreover, further comparative analysis and performance studies involving FA-based hybrid algorithms and K-means-based metaheuristic algorithms can be carried out. Aside these, the proposed algorithm can also be applied to other real-life applications.

**Author Contributions:** All authors contributed to the conception and design of the research work. Material preparation, experiments, and analysis were performed by A.M.I. and A.E.E. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research received no external funding.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** Not applicable.

**Conflicts of Interest:** The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this article.

## References

1. Esmin, A.A.A.; Coelho, R.A.; Matwin, S. A review on particle swarm optimization algorithm and its variants to clustering high-dimensional data. *Artif. Intell. Rev.* **2015**, *44*, 23–45. [CrossRef]
2. Von Luxburg, U. A tutorial on spectral clustering. *Stat. Comput.* **2007**, *17*, 395–416. [CrossRef]
3. Kriegel, H.-P.; Kröger, P.; Zimek, A. Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering. *ACM Trans. Knowl. Discov. Data* **2009**, *3*, 1–58. [CrossRef]
4. McCallum, A.; Nigam, K.; Ungar, L.H. Efficient clustering of high-dimensional data sets with application to reference matching. In Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Boston, MA, USA, 20–23 August 2000; pp. 169–178. [CrossRef]
5. Agarwal, P.; Mehta, S.; Abraham, A. A meta-heuristic density-based subspace clustering algorithm for high-dimensional data. *Soft Comput.* **2021**, *25*, 10237–10256. [CrossRef]
6. Punhani, A.; Faujdar, N.; Mishra, K.K.; Subramanian, M. Binning-Based Silhouette Approach to Find the Optimal Cluster Using K-Means. *IEEE Access* **2022**, *10*, 115025–115032. [CrossRef]
7. Ikotun, A.M.; Ezugwu, A.E. A comprehensive survey of K-means clustering algorithm and analysis of variants. *Inf. Sci.* **2022**; *in press*.
8. Xie, T.; Liu, R.; Wei, Z. Improvement of the Fast Clustering Algorithm Improved by *K*-Means in the Big Data. *Appl. Math. Nonlinear Sci.* **2020**, *5*, 1–10. [CrossRef]
9. Alguliyev, R.M.; Aliguliyev, R.M.; Sukhostat, L.V. Parallel batch k-means for Big data clustering. *Comput. Ind. Eng.* **2021**, *152*, 107023. [CrossRef]
10. Zhang, G.; Zhang, C.; Zhang, H. Improved K-means algorithm based on density Canopy. *Knowl.-Based Syst.* **2018**, *145*, 289–297. [CrossRef]
11. Alguliyev, R.; Aliguliyev, R.; Bagirov, A.; Karimov, R. Batch clustering algorithm for big data sets. In Proceedings of the 2016 IEEE 10th International Conference on Application of Information and Communication Technologies (AICT), Baku, Azerbaijan, 12–14 October 2016; pp. 1–4. [CrossRef]
12. Olukanmi, P.O.; Nelwamondo, F.; Marwala, T. k-Means-Lite: Real Time Clustering for Large Datasets. In Proceedings of the 2018 5th International Conference on Soft Computing & Machine Intelligence (ISCMI), Nairobi, Kenya, 21–22 November 2018; pp. 54–59. [CrossRef]
13. Islam, Z.; Estivill-Castro, V.; Rahman, A.; Bossomaier, T. Combining K-Means and a genetic algorithm through a novel arrangement of genetic operators for high quality clustering. *Expert Syst. Appl.* **2018**, *91*, 402–417. [CrossRef]
14. Sinha, A.; Jana, P.K. A hybrid MapReduce-based k-means clustering using genetic algorithm for distributed datasets. *J. Supercomput.* **2018**, *74*, 1562–1579. [CrossRef]
15. Zhang, H.; Zhou, X. A novel clustering algorithm combining niche genetic algorithm with canopy and K-means. In Proceedings of the 2018 International Conference on Artificial Intelligence and Big Data (ICAIBD), Chengdu, China, 26–28 May 2018; pp. 26–32. [CrossRef]
16. Behera, H.S.; Nayak, J.; Nanda, M.; Nayak, K. A novel hybrid approach for real world data clustering algorithm based on fuzzy C-means and firefly algorithm. *Int. J. Fuzzy Comput. Model.* **2015**, *1*, 431. [CrossRef]
17. Paul, S.; De, S.; Dey, S. A Novel Approach of Data Clustering Using An Improved Particle Swarm Optimization Based K–Means Clustering Algorithm. In Proceedings of the 2020 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT), Bangalore, India, 2–4 July 2020; pp. 1–6. [CrossRef]

18. Kaur, A.; Pal, S.K.; Singh, A.P. Hybridization of K-Means and Firefly Algorithm for intrusion detection system. *Int. J. Syst. Assur. Eng. Manag.* **2017**, *9*, 901–910. [CrossRef]
19. Pavez, L.; Altimiras, F.; Villavicencio, G. A K-means Bat Algorithm Applied to the Knapsack Problem. In *Proceedings of the Computational Methods in Systems and Software*; Springer: Cham, Switzerland, 2020; pp. 612–621. [CrossRef]
20. Kumari, G.V.; Rao, G.S.; Rao, B.P. Flower pollination-based K-means algorithm for medical image compression. *Int. J. Adv. Intell. Paradig.* **2021**, *18*, 171. [CrossRef]
21. Wang, X.; Yu, H.; Lin, Y.; Zhang, Z.; Gong, X. Dynamic Equivalent Modeling for Wind Farms With DFIGs Using the Artificial Bee Colony With K-Means Algorithm. *IEEE Access* **2020**, *8*, 173723–173731. [CrossRef]
22. Ikotun, A.M.; Almutari, M.S.; Ezugwu, A.E. K-Means-Based Nature-Inspired Metaheuristic Algorithms for Automatic Data Clustering Problems: Recent Advances and Future Directions. *Appl. Sci.* **2021**, *11*, 11246. [CrossRef]
23. Ikotun, A.M.; Ezugwu, A.E. Boosting k-means clustering with symbiotic organisms search for automatic clustering problems. *PLoS ONE* **2022**, *17*, e0272861. [CrossRef]
24. Yang, X. Firefly Algorithms for Multimodal Optimization. *Stoch. Algorithms Found. Appl.* **2009**, *5792*, 169–178.
25. Fister, I.; Fister, I., Jr.; Yang, X.S.; Brest, J. A comprehensive review of firefly algorithms. *Swarm Evol. Comput.* **2013**, *13*, 34–46. [CrossRef]
26. Agbaje, M.B.; Ezugwu, A.E.; Els, R. Automatic Data Clustering Using Hybrid Firefly Particle Swarm Optimization Algorithm. *IEEE Access* **2019**, *7*, 184963–184984. [CrossRef]
27. Ariyaratne, M.K.A.; Fernando, T.G.I. A Comprehensive Review of the Firefly Algorithms for Data Clustering. *Adv. Swarm Intell.* **2022**, *1054*, 217–239.
28. Kumar, V.; Kumar, D. A Systematic Review on Firefly Algorithm: Past, Present, and Future. *Arch. Comput. Methods Eng.* **2020**, *28*, 3269–3291. [CrossRef]
29. Hassanzadeh, T.; Meybodi, M.R. A new hybrid approach for data clustering using firefly algorithm and K-means. In Proceedings of the 16th CSI International Symposium on Artificial Intelligence and Signal Processing (AISP 2012), Shiraz, Iran, 2–3 May 2012; pp. 007–011. [CrossRef]
30. Mathew, J.; Vijayakumar, R. Scalable parallel clustering approach for large data using parallel K means and firefly algorithms. In Proceedings of the 2014 International Conference on High Performance Computing and Applications (ICHPCA), Bhubaneswar, India, 22–24 December 2014; pp. 1–8. [CrossRef]
31. Nayak, S.; Panda, C.; Xalxo, Z.; Behera, H.S. An Integrated Clustering Framework Using Optimized K-means with Firefly and Canopies. In *Computational Intelligence in Data Mining-Volume 2*; Springer: New Delhi, India, 2015; pp. 333–343. [CrossRef]
32. Nayak, J.; Naik, B.; Behera, H.S. Cluster Analysis Using Firefly-Based K-means Algorithm: A Combined Approach. In *Computational Intelligence in Data Mining*; Springer: Singapore, 2017; pp. 55–64. [CrossRef]
33. Xie, H.; Zhang, L.; Lim, C.P.; Yu, Y.; Liu, C.; Liu, H.; Walters, J. Improving K-means clustering with enhanced Firefly Algorithms. *Appl. Soft Comput.* **2019**, *84*, 105763. [CrossRef]
34. Davies, D.L.; Bouldin, D.W. A Cluster Separation Measure. *IEEE Trans. Pattern Anal. Mach. Intell.* **1979**, *PAMI-1*, 224–227. [CrossRef]
35. Fister, I.; Yang, X.-S.; Fister, D. Firefly Algorithm: A Brief Review of the Expanding Literature. *Cuckoo Search Firefly Algorithm* **2014**, *516*, 347–360. [CrossRef]
36. Senthilnath, J.; Omkar, S.; Mani, V. Clustering using firefly algorithm: Performance study. *Swarm Evol. Comput.* **2011**, *1*, 164–171. [CrossRef]
37. Rajah, V.; Ezugwu, A.E. Hybrid Symbiotic Organism Search algorithms for Automatic Data Clustering. In Proceedings of the 2020 Conference on Information Communications Technology and Society (ICTAS), Durban, South Africa, 11–12 March 2020; pp. 1–9. [CrossRef]
38. Ezugwu, A.E.-S.; Agbaje, M.B.; Aljojo, N.; Els, R.; Chiroma, H.; Elaziz, M.A. A Comparative Performance Study of Hybrid Firefly Algorithms for Automatic Data Clustering. *IEEE Access* **2020**, *8*, 121089–121118. [CrossRef]
39. Bezdek, J.C.; Ehrlich, R.; Full, W. FCM: The fuzzy c-means clustering algorithm. *Comput. Geosci.* **1984**, *10*, 191–203. [CrossRef]
40. Kumar, A.; Ingle, Y.S.; Pande, A.; Dhule, P. Canopy Clustering: A Review on Pre-Clustering Approach to K-Means Clustering. *Int. J. Innov. Adv. Comput. Sci. (IJIACS)* **2014**, *3*, 22–29.
41. Jitpakdee, P.; Aimmanee, P.; Uyyanonvara, B. A Hybrid Approach for Color Image Quantization Using K-means and Firefly Algorithms. *World Acad. Sci. Eng. Technol.* **2013**, *7*, 142–149.
42. Kuo, R.; Li, P. Taiwanese export trade forecasting using firefly algorithm-based K-means algorithm and SVR with wavelet transform. *Comput. Ind. Eng.* **2016**, *99*, 153–161. [CrossRef]
43. Langari, R.K.; Sardar, S.; Mousavi, S.A.A.; Radfar, R. Combined fuzzy clustering and firefly algorithm for privacy preserving in social networks. *Expert Syst. Appl.* **2020**, *141*, 112968. [CrossRef]
44. Yang, X.-S.; Deb, S.; Zhao, Y.-X.; Fong, S.; He, X. Swarm intelligence: Past, present and future. *Soft Comput.* **2018**, *22*, 5923–5933. [CrossRef]
45. Jain, A.K.; Murty, M.N.; Flynn, P.J. Data clustering: A review. *ACM Comput. Surv. (CSUR)* **1999**, *31*, 264–323. [CrossRef]
46. Jain, A.K.; Dubes, R.C. *Algorithms for Clustering Data*; Prentice-Hall, Inc.: Hoboken, NJ, USA, 1988.
47. Jain, A.K. Data clustering: 50 years beyond K-means. *Pattern Recognit. Lett.* **2010**, *31*, 651–666. [CrossRef]

48. Pelleg, D.; Moore, A. Accelerating exact k-means algorithms with geometric reasoning. In Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Diego, CA, USA, 15–18 August 1999; pp. 277–281.
49. Alsabti, K.; Ranka, S.; Singh, V. An Efficient K-Means Clustering Algorithm. 1997. Available online: https://surface.syr.edu/eecshttps://surface.syr.edu/eecs/43 (accessed on 7 October 2020).
50. Nittel, S.; Leung, K.; Braverman, A. Scaling clustering algorithms for massive data sets using data streams. In Proceedings of the International Conference on Data Engineering, Boston, MA, USA, 30 March–2 April 2004; Volume 20, p. 830. [CrossRef]
51. Bradley, P.S.; Fayyad, U.; Reina, C. Scaling EM Clustering to Large Databases Scaling EM (Expectation-Maximization) Clustering to Large Databases. *Electr. Eng. Comput. Sci.—All Scholarsh.* **1998**, *43*, 0–25. Available online: https://surface.syr.edu/eecs/43 (accessed on 8 October 2022).
52. Jin, R.; Goswami, A.; Agrawal, G. Fast and exact out-of-core and distributed k-means clustering. *Knowl. Inf. Syst.* **2006**, *10*, 17–40. [CrossRef]
53. Domingos, P.; Hulten, G. A General Method for Scaling up machine learning algorithm and its application to clustering. In Proceedings of the Eighteenth International Conference on Machine Learning, Williamstown, MA, USA, 28 June–1 July 2001.
54. Olukanmi, P.O.; Nelwamondo, F.; Marwala, T. Rethining k-means clustering in the age of massive datasets: A constant-time approach. *Neural Comput. Applic.* **2020**, *32*, 15445–15467. [CrossRef]
55. José-García, A.; Gómez-Flores, W. Automatic clustering using nature-inspired metaheuristics: A survey. *Appl. Soft Comput.* **2016**, *41*, 192–213. [CrossRef]
56. Ezugwu, A.E. Nature-inspired metaheuristic techniques for automatic clustering: A survey and performance study. *SN Appl. Sci.* **2020**, *2*, 273. [CrossRef]
57. Cheng, M.-Y.; Prayogo, D. Symbiotic Organisms Search: A new metaheuristic optimization algorithm. *Comput. Struct.* **2014**, *139*, 98–112. [CrossRef]
58. Das, S.; Konar, A. Automatic image pixel clustering with an improved differential evolution. *Appl. Soft Comput.* **2009**, *9*, 226–236. [CrossRef]
59. Bandyopadhyay, S.; Maulik, U. Nonparametric genetic clustering: Comparison of validity indices. *IEEE Trans. Syst. Man Cybern. Part C (Appl. Rev.)* **2001**, *31*, 120–125. [CrossRef]
60. Zhou, X.; Gu, J.; Shen, S.; Ma, H.; Miao, F.; Zhang, H.; Gong, H. An Automatic K-Means Clustering Algorithm of GPS Data Combining a Novel Niche Genetic Algorithm with Noise and Density. *ISPRS Int. J. Geo-Inf.* **2017**, *6*, 392. [CrossRef]
61. Bandyopadhyay, S.; Maulik, U. Genetic clustering for automatic evolution of clusters and application to image classification. *Pattern Recognit.* **2002**, *35*, 1197–1208. [CrossRef]
62. Kumar, V.; Chhabra, J.K.; Kumar, D. Automatic Data Clustering Using Parameter Adaptive Harmony Search Algorithm and Its Application to Image Segmentation. *J. Intell. Syst.* **2016**, *25*, 595–610. [CrossRef]
63. Liu, Y.; Wu, X.; Shen, Y. Automatic clustering using genetic algorithms. *Appl. Math. Comput.* **2011**, *218*, 1267–1279. [CrossRef]
64. Lai, C.-C. A Novel Clustering Approach using Hierarchical Genetic Algorithms. *Intell. Autom. Soft Comput.* **2005**, *11*, 143–153. [CrossRef]
65. Lin, H.-J.; Yang, F.-W.; Kao, Y.-T. An Efficient GA-based Clustering Technique. *J. Appl. Sci. Eng.* **2005**, *8*, 113–122.
66. Anari, B.; Torkestani, J.A.; Rahmani, A. Automatic data clustering using continuous action-set learning automata and its application in segmentation of images. *Appl. Soft Comput.* **2017**, *51*, 253–265. [CrossRef]
67. Kumar, V.; Chhabra, J.K.; Kumar, D. Automatic cluster evolution using gravitational search algorithm and its application on image segmentation. *Eng. Appl. Artif. Intell.* **2014**, *29*, 93–103. [CrossRef]
68. Liu, R.; Zhu, B.; Bian, R.; Ma, Y.; Jiao, L. Dynamic local search based immune automatic clustering algorithm and its applications. *Appl. Soft Comput.* **2015**, *27*, 250–268. [CrossRef]
69. Chowdhury, A.; Bose, S.; Das, S. Automatic Clustering Based on Invasive Weed Optimization Algorithm. In *International Conference on Swarm, Evolutionary, and Memetic Computing*; Springer: Berlin/Heidelberg, Germany, 2011; pp. 105–112. [CrossRef]