Indonesian Journal of Electrical Engineering and Computer Science
Vol. 24, No. 3, December 2021, pp. 1744~1758
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v24.i3.pp1744-1758

Grey wolf optimization algorithm for hierarchical document clustering

Ayad Mohammed Jabbar[1], Ku Ruhana Ku-Mahamud[2]
[1]Department of Computer Science, Shatt Al-Arab University College, Basra, Iraq
[2] Data Science Research Lab, School of Computing, Universiti Utara Malaysia, Kedah, Malaysia
[2]Shibaura Institute of Technology, Tokyo, Japan

Article Info

Article history:

Received May 12, 2021
Revised Oct 8, 2021
Accepted Oct 13, 2021

Keywords:

Centroid-based clustering
Data clustering
Medoid-based clustering
Optimization
Swarm intelligence
Unsupervised classification

ABSTRACT

In data mining, the application of grey wolf optimization (GWO) algorithm has been used in several learning approaches because of its simplicity in adapting to different application domains. Most recent works that concern unsupervised learning have focused on text clustering, where the GWO algorithm shows promising results. Although GWO has great potential in performing text clustering, it has limitations in dealing with outlier documents and noise data. This research introduces medoid GWO (M-GWO) algorithm, which incorporates a medoid recalculation process to share the information of medoids among the three best wolves and the rest of the population. This improvement aims to find the best set of medoids during the algorithm run and increases the exploitation search to find more local regions in the search space. Experimental results obtained from using well-known algorithms, such as genetic, firefly, GWO, and k-means algorithms, in four benchmarks. The results of external evaluation metrics, such as rand, purity, F-measure, and entropy, indicates that the proposed M-GWO algorithm achieves better document clustering than all other algorithms (i.e., 75% better when using Rand metric, 50% better than all algorithm based on purity metric, 75% better than all algorithms using F-measure metric, and 100% based on entropy metric).

Corresponding Author:

Ayad Mohammed Jabbar
Department of Computer Science
Shatt Al-Arab University College
Basra, Iraq
Email: ayadmohammed@sa-uc.edu.iq

1. INTRODUCTION

Data clustering refers to a method of classifying a set of items into several groups wherein each single group contains items with high similarity in terms of the distance between the features of each item [1]-[4]. It has been used in different applications dominions such as wireless sensor networks, text classification, and analysis of Twitter data [5]-[7]. Two types of clustering approach, namely, partitional and hierarchical, can be constructed [8]. Items in partitional clustering groups are based on the similarity among the items. This clustering group is spherical, wherein each item is placed in a single cluster. The data are divided into a set of clusters where the similarity among the items within a cluster is high and differ from the other items that belong to another cluster [9]. A distance-based similarity measure is used to determine the degree of similarity among the items to produce the clustering assignment. Partitional clustering approach includes center-based, grid-based, model-based, and density-based clustering [10]. Center-based clustering constructs spherical groups represented in the center of the cluster [11]. The center

of cluster is a metaphor that is calculated as the average distance among all items within the cluster (i.e., centroid) or when clustering is performed according to medoid-based clustering (i.e., medoid) [12]. Medoid-based clustering considers the center of each cluster as the medoid that represents one instance that belongs to the dataset. Grid-based clustering divides the dataset into different numbers of cells for various resolutions, wherein a cell level corresponds to a level of resolution [13]. Each cell represents a statistical value that is calculated and used to answer queries. The advantage of this approach includes its ability to deal with high-dimensional space [14]. However, the performance of the algorithm is based on the grid size, which is lower than the dataset size. For highly irregular data distributions, the use of single uniform grid is insufficient to obtain the required clustering quality or fulfil the time requirement. Model-based clustering method attempts to optimize the fit among mathematical models and data [15]. This method characterizes the description of data groups where each group appears with a class. The main challenges in model-based clustering include the curse of dimensionality problem, the initial selection, and the problem of estimating the number of parameters [15]. In density-based clustering, the algorithm groups the data item based on the dense regions by identifying the high- and low-density areas [16]. The algorithm can handle noise, detect outliers, and identify clustering shapes. However, it has difficulty in dealing with high-dimensional data and different level of densities within clusters. These difficulties can limit the clustering performance of the algorithm such as low accuracy due to wrong clustering assignment [16], [17]. Hierarchical clustering is performed using either agglomerative or divisive hierarchical approaches [18]. The former performs oppositely, that is, clustering is performed by computing the similarity between clusters and then joining the two most similar clusters until one single cluster is observed. The latter works in reverse manner, that is, a tree is constructed by dividing the dataset into sub-clusters recursively until each data item represents a single cluster. The advantage of the former over the latter is its suitability for text clustering because it reviews the data as a hierarchy of nested quality clusters and does not require the number of clusters, which is a drawback of partitional algorithm [19]. Hierarchical clustering performs clustering based on the proximity matrix, which is calculated using the distance between each point. The distance between each point can be measured using three methods, namely, single, complete, and average linkages. The single linkage in hierarchical clustering can be measured based on the shortest distance between two elements that reside in two different clusters. The complete linkage in hierarchical clustering is the longest distance between two elements that reside in two different clusters. The average linkage in hierarchical clustering is the average of the sum of the distance between each element in a single cluster to every element in the other cluster.

In any clustering approaches, the assessment of clustering tendency focuses on the feasibility of the clustering analysis to determine whether the data can be meaningfully clustered or not [20]. Hierarchical clustering is more suitable in inducing the clustering tendency of data than the partitional one [21]. The hierarchical clustering algorithm visualizes data as a hierarchical tree that illustrates the fusion or division in each stage because the tree contains nested clusters. As mentioned, the hierarchical clustering approach is classified into agglomerative and divisive methods [22]. The agglomerative method begins by merging distinct clusters (items) based on similarity until a single cluster that contains all members is obtained. The divisive method performs a series of partitions that contain single clusters and separates them into multiple sub-clusters successively. The important issue that arises in agglomerative clustering is how to obtain the tree that is used to explore the individual items from the partition. To the best of the author's knowledge, the hierarchical tree may contain different resolutions at each level of the tree that corresponds to the number of clusters [23]. Thus, estimating the number of optimal partition and deciding the optimal cut level of a dendrogram can be challenging [24]. Other problems in the hierarchical clustering involve the outliers that directly affect the accuracy of the results. Outlier refers to a data item that does not fit into any cluster. Outliers are objects with low connectivity in opposition to those with high connectivity in the intra-cluster region. The hierarchical clustering is sensitive to outliers because many terms are not represented by all documents after pre-processing [13], [25].

Different related works on text clustering for several optimization algorithms have been proposed in the literature. Literature shows different approaches, such as density- and centroid-based clustering, that are used to produce the documents clustering tree. In 2015, related work has proposed a text clustering algorithm to determine the density of peaks [26]. The algorithm was implemented by calculating text distance and density, which were based on the calculation of the text vector similarity. Vector space model (VSM) was used to express text to obtain the vector mapping for the similarity calculation. The local density and distance from the points of higher density of each text are determined, the noise points are removed, and the cluster centers are selected. However, cluster centers obtained based on centroid-based approach are sensitive to outliers and noise points. Other related works used density-based algorithms to perform hierarchical clustering [27]. The number of parameters needed in the density-based method is reduced by using the k-means algorithm. Average and single linkage algorithms are improved and

combined with density-based methods, which reduce the time complexity required in the end. Evaluated results showed improvement when k-means and single linkage are compared. However, the algorithm performs density-based clustering without attending to outlier problems as part of its solution.

In 2010, other related work proposed the maximum capturing algorithm [28], which consisted of two procedures, namely, constructing document clusters and assigning cluster topics. The first procedure, which is employed for groups, documents pairs with the largest similarity in the dataset as one single cluster. The second procedure utilizes groups with the most frequent item sets of each cluster presented as topics of the cluster. Similarly, it adopts frequent item sets for documents when measuring instead of VSM to use the power of frequent item sets in handling document size. The similarity measure for documents is calculated in three ways: the number of frequent item sets, the total weights of frequent item sets, and the asymmetrical binary similarity between their frequent item sets. The evaluated results conclude a better performance when compared with other methods. Other research proposed a hybrid algorithm that combines density and partitional clustering [29]. Hybridization aims to overcome the effect of high time cost on the density-based spatial clustering of applications with noise (DBSCAN) algorithm. It utilizes the k-means algorithm for partitions of the dataset as the first stage and then employs the Max–Min method to select points for the DBSCAN algorithm as the second stage. The results of the proposed algorithm outperformed the original DBSCAN. The quality of result is sensitive because it is based on the pre-defined number of clusters that should be assumed by the user. Moreover, the algorithm is sensitive to outliers.

In 2016, other related work proposed an agglomerative hierarchical clustering that employed a new clustering validity index performed on the basis of two concepts, including dispersion and synthesis degrees [30]. The first calculation differentiates the intra-cluster compactness and inter-cluster separation, whereas the synthesis degree sums intra-cluster compactness with inter-cluster separation. The ratio of the clustering dispersion degree and clustering synthesis determines the quality of cluster according to the level of partition in agglomerative hierarchical clustering. However, the clustering ratio can be calculated in the final stage of the agglomerative hierarchical clustering results. The hybrid algorithm contains divisive and agglomerative-performed text clustering [31]. It contains the advantage of both algorithms. This hybridization solves the problem of local minimum produced in the hierarchical clustering due to the difficulty in re-joining items in different stages. In the first stage, a greater total number of clusters than the actual number is produced. Obtained centroids are merged to form actual clusters that are considered the second stage of hybrid algorithm. Results are sensitive to the initial number of clusters, outliers, and noise. Research has been conducted to determine the optimal number of clusters in hierarchical clustering by proposing a new cluster validity index to find the best partition [32]. The proposed method extends the search space to find the extended partition. Thus, it is a new clustering validity index called context-independent optimality that can be used to find the best partition in hierarchical clustering. However, this study uses a validity index that calculated the similarity on the basis of centroid-based approach, which is sensitive to outlier and noise data. The partitional and hierarchical approaches have been hybridized to overcome disadvantages of both algorithms [33]. A modified k_mode algorithm performed partitional clustering based on a pre-defined number of clusters as the initial stage. Meanwhile, the final stage performed agglomerative clustering to construct a hierarchy tree. However, the proposed hybrid algorithm produced unstable results. Similarly, a hybrid method between partitioning around medoids (PAM) and divisive hierarchical clustering, in which the number of main clusters is determined using Davies Bouldin index (DBI) value with an optimal number of clusters if the results minimize the DBI value, was proposed [18]. Based on the obtained number of clusters, PAM performs partitional clustering followed by divisive hierarchical clustering to conduct the final results. Other studies proposed a hybrid hierarchical clustering algorithm to improve the time complexity of hierarchical clustering algorithm. The proposed algorithms perform clustering as the first step using k-means algorithm [34]. Subsequently, objects within each cluster are clustered hierarchically by using the exact agglomerative hierarchical clustering algorithm. The time complexity of agglomerative hierarchical clustering was improved in terms of time. However, the result is critical and dependent on the first stage, which may produce or contain local minimum results.

In 2012, other related work proposed a divisive approach to solve high-dimensional space problems [35]. The proposed algorithm consists of two phases. First, the original dataset is divided into two sub-clusters to measure the homogeneity of new clusters to determine whether the partitioning should be continued or whether the original cluster need to be retained. Second, stability is determined by substituting the location of one member of a cluster to another. If the quality improves, then partitioning is performed. Otherwise, the cluster remains in the same cluster. The main challenge in hierarchical clustering involves handling the huge collection of data, which causes high computational cost. Other research improved the efficiency of hierarchical clustering by building a hierarchy algorithm based on adjacent points in clusters with a centroid [35]. To reduce the overall computational cost, the proposed

algorithm combines partitional and agglomerative clustering when k-means is used for partitional clustering. The process begins by assigning data objects to the appropriate centroid and generates a number of clusters called middle-level clusters. Agglomerative clustering is applied on the obtained centroids to generate clustering hierarchy. The evaluated results clearly demonstrate improvements in terms of computational cost when compared with the standard agglomerative hierarchical clustering method. Other related works use ant colony optimization (ACO) algorithm to perform numerical clustering similar to what PAM algorithm do [36], [37]. However, the proposed method, such as in [34], [35], have high limitation, that is, no information about the best medoids can be identified, and the algorithm does not swap the medoids during the run. Such phenomenon is known as the medoid recalculation process that exists in PAM algorithm. The medoids are randomly generated, and clustering assignment is performed. The limitation of swaping does not allow more regions to be found in the search space with more optimial clustering assignments. As an optimization algorithm grey wolf optimization (GWO) has been used for different applications domains due to its simplicity to adapt in any optimization algorithm [38]-[43]. It successfully showed promising results such as in wireless sensor [44], resource allocation in cloud environment [45], classification [45], feature selection [46], and other optimization problems [47], [48]. Following the same procedure, GWO has been employed for document clustering, wherein the algorithm performs text clustering based on centroid-based clustering [49]. However, although the algorithm shows remarkable results for text clustering, it has a limitation in dealing with datasets that contain outlier text document, specially if documents have few terms when compared with other documents contain several terms that represent different key information about the meaning of the documents.

Data that contain outliers increase dissimilarity among each cluster and may result in an algorithm that produces inaccurate results. Text clustering suffers in identifying local outliers because many terms do not appear in all documents after the pre-processing step. Thus, the algorithm that produced hierarchical clustering should avoid outliers. Alternatively, the algorithm may use a pre-processing step to overcome this problem such as using other methods to remove the outliers. From this point, the proposed algorithm uses medoid-based clustering, which is an initial phase for data clustering, followed by agglomerative approach, which represents the data of each cluster as an agglomerative hierarchical tree. Selecting medoid-based approaches is better than opting for centroid-based ones because of the presence of noise and outliers in the data. The search space of medoid-based approaches is lesser compared with that of the search space of the centroid-based one where it is limited to the number of instances in the dataset. The limitation in [49] leads to the use of GWO algorithm to group a set of documents into different number of clusters based on medoids-based clustering, which is better than what was used by [49] in dealing with outlier documents and noise data. Medoid calculation process, which is a limitation in [36], [37] wherein the proposed algorithm will swap the medoids between the wolves to keep the best medoids during the algorithm run and find more regions with better document clustering assignments, has been added in the research.

This paper has been organized into several sections. Section 2 shows the methodology of the study. section 3 represents the proposed algorithm. Sections 4 shows the steps of pre-processing and section 5 discuss the analysis of the experiential results. Section 6 is the conclusion and future work, respectively.

2. RESEARCH METHOD

The process of hierarchical clustering text based on the proposed algorithm technique begins by performing text clustering based on medoid-based clustering known as medoid grey wolf optimization algorithm (M-GWO). After finishing the pre-processing step in document representation, each document is represented by a set of features that shows the weight of the document compared with other documents. Weight is used in clustering to classify each document to appropriate cluster according to the distance between the documents and the center (medoid) of each cluster. This clustering assignment ensures that the output of each cluster is better than when the GWO algorithm uses centroid-based clustering. The output of M-GWO is a spherical clustering assignment that can be used to identify the document groups. However, it is not useful as the hierarchical tree, which shows each document in a cluster with each close document, thereby needing an adaptational stage that converts each single cluster to a single tree. The next stage is to apply agglomerative clustering for each of the single cluster produced by the M-GWO algorithm, as shown in Figure 1.

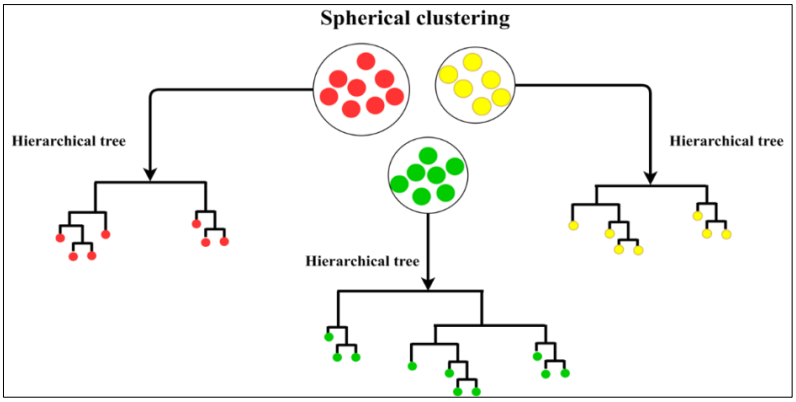

Figure 1. Research method of M-GWO algorithm

3. PROPOSED M-GWO ALGORITHM FOR TEXT CLUSTERING

The major differences between the original GWO algorithm and the M-GWO algorithm is that the former produces the clustering results based on centroids, which require wide search space, whereas the latter is based on medoids that requires search space that is equals to the number of objects (documents). However, few modifications are required to perform the GWO algorithm in a medoid algorithm. These modifications will be described and supported by examples and explanations. The M-GWO steps are described in detail in M-GWO algorithm. Here, the wolf is designated as a search agent. The proposed M-GWO algorithm has one modification placed in the update position procedure, where medoids of omega wolf are updated according to the three best wolves, including alpha (α), beta (β), delta (δ). The algorithm begins its text clustering by setting up all the required parameters, such as the number of wolves N, the number of iterations, and the number of clusters k, which are set by the user, as shown in Step 1. Step 2 involves the initialization of the population where each agent assigned a document clustering solution based on the medoids of the same agent, which are randomly generated. This step is explained in Section 4.1, which includes agent representation and population initialization. In Step 3, each agent calculates its fitness function to find the three best wolves, which includes α, β, and δ in ascending order, as described in Section 4.2. In Steps 4 and 5, the algorithm begins its iteration followed by Step 6 and 7, wherein the position of each agent is updated according the three best wolves, namely, α, β, and δ. The operation aims to move the position of the agent into a better position near the three best wolves. Step 7 is described in detail in Section 4.3, which contains the contribution of this research. In Steps 9, 10, and 11, the algorithm computes the fitness of all wolves after updating the position and identifying the three best wolves X_α , X_β , and X_δ based on the fitness of each agent, as shown in Step 10. This step is followed by increasing the loop to start improving the fitness function of each agent again. The final solution in Step 13 applies agglomerative clustering to produce the hierarchical clustering tree of each single cluster.

M-GWO algorithm

1: Setup the parameters: number of wolves, N, and maximum number of iterations $Max\_iteration$ and number of clusters k
2: Generate initial wolf population, N, where each wolf, X_i, $i = \{1,2,3,\ldots,N\}$ and each X_i has own clustering assignment and own medoids randomly generated.
3: Calculate fitness, $f(X_i)$, for each wolf, X_i and identify three best wolves in the population based on fitness of each one and sort them in ascending order as X_α, X_β, and X_δ
4: Set $iteration = 0$
5: **while** ($iteration < Max\_iteration$) **do**
6: **for** each X_i = 1 **to** N **do**
7: Update position of agents
8: **end-for**
9: Compute fitness of all wolves
10: Update value of X_α , X_β , and X_δ based on fitness
11: $iteration = iteration + 1$
12: **end-while**
13 Print X_α tour solution with its fitness and apply the agglomerative clustering on X_α tour solution
end-algorithm

3.1. Agent representation and population initialization

The M-GWO algorithm begins with the initial clustering solution, where each document represented by unique medoid number. As shown in Figure 2, a clustering solution consists of nine documents where each document is assigned to a closet medoid, such as first one to medoid 1, second document to medoid 3, and so on. The medoids of each clustering solution can be represented as three medoids for this single solution, as shown as example in Figure 2. In Figure 3, an example of agent has three medoids, the three medoids (three documents) are presented to perform clustering solution as shown in Figure 2. The first, second, and third medoids have two attributes each (0.99, 0.87), (0.34, 0.54), and (0.68, 0.98), respectively. Medoids are unique, which means no medoids have duplicated attributes.

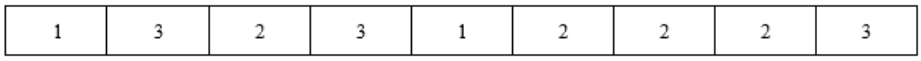

Figure 2. Solution representation

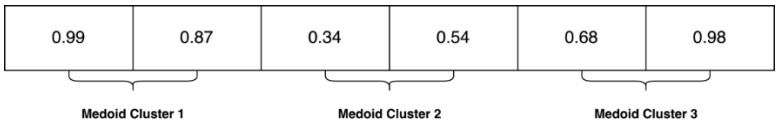

Figure 3. Medoid representation

Each agent in the initial solution phase is represented by single solution (Figure 2), and the medoids are set on the basis of the maximum of number of clusters known as k set, as shown in Figure 3. In the initial phase, the medoids are selected randomly. Subsequently, during the algorithm run, the medoids are stochastically improved to find the best set of medoids that produce minimum error among the clusters.

3.2. Fitness function computation

The fitness function, as shown in (1), is the cosine similarity measurement between the medoids and the documents $\overrightarrow{d_i}, \overrightarrow{d_j}$ where $\overrightarrow{d_i}$, is the first medoid and $\overrightarrow{d_j}$ is a document. Notably, this fitness function is used as the main objective function in the clustering assignment that identifies the distance between the medoids and the documents and later used as a fitness function to calculate the total distance between each cluster medoid and the remaining documents in the same cluster.

$$Cos\left(\overrightarrow{medoid_i}, \overrightarrow{d_j}\right) = \frac{\overrightarrow{medoid_i} * \overrightarrow{d_j}}{\sqrt{\sum_{i=1}^{t} \overrightarrow{medoid_i}^2} * \sqrt{\sum_{j=1}^{t} \overrightarrow{d_j}^2}} \quad (1)$$

An agent has three medoids, as shown in Figure 3, and a document is required to be assigned to one of the medoids, such as medoid 1, medoid 2 or medoid 3. If the document d has two attributes (terms), such as (0.43, 0.98), then the assignment will be performed as follows:

$Cos\left(\overrightarrow{medoid_1}, \vec{d}\right) = 0.906$
$Cos\left(\overrightarrow{medoid_2}, \vec{d}\right) = 0.989$
$Cos\left(\overrightarrow{medoid_3}, \vec{d}\right) = 0.981$

The cosine similarity measurement indicates that the similarity between medoid 2 and the document is higher than those of the other medoids, thus the document will be assigned to the close medoid 3. Notably, the details of calculation are described in the pre-processing section.

3.3. Position updating

Medoid grey wolf optimization (M-GWO) followed the same procedure of GWO in its hierarchy leaderships, where four main wolves include α, β, δ, and the remaining members are omega (ω), are available. The ω wolves update their position on the search space according the best positions represented by α, β, and δ. In M-GWO, the contribution of this research is that the position of ω wolves (medoids) are only updated according to the three main wolves, which are α, β, and δ. The process of update position begins by selecting one medoid from the ω wolf in randomly manner. The target of this selection is to perform swap operation between ω wolf and α, β, and δ, as shown in Figure 4. As shown in Figure 4, the swap operation will be performed between ω and the three main wolves. Number m is randomly generated where $m < k$.

Subsequently, based on m value, the algorithm selects one of ω medoids as an active medoid to be swapped with one of α, β, and δ medoids. However, the swap operation here is conditional if and only if the fitness of a new set of medoids is better than the older one. The other condition is that the swap operation always looks for the first improvement, which means swapping is stopped after the first improvement is achieved. Such condition allows the algorithm to converge to the optimal solution slowly and avoid stack at the local optima because if we accepted high-quality solutions in the beginning of the run, the algorithm will ignore the exploration of the wide regions of the search space. The swap operation that represented in Figure 4 is the main operation used in PAM known as the medoid calculation process, which allows the discovery of other optimal clustering assignments in the search space. An advantage of the operation includes reduced calculations process when only the ω wolf and the three best wolves are calculated. This operation is also better than PAM algorithm, which requires more calculation between the medoid that need to be changed with all other candidate medoids in the search space. Note that M in Figure 4 corresponds to the current medoid, and T refers to the term.

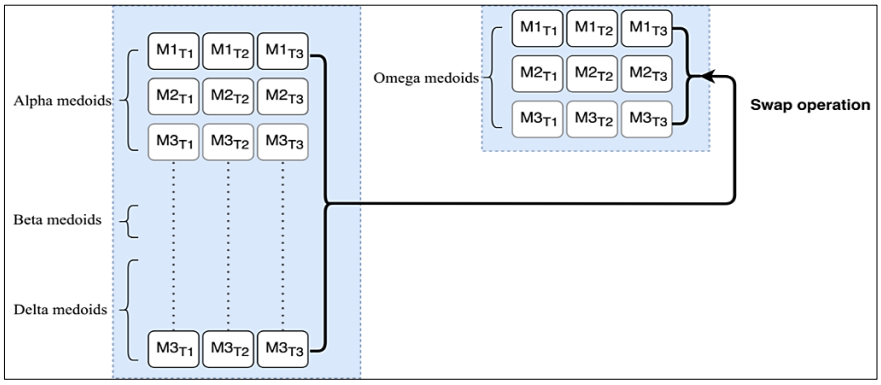

Figure 4. Medoids' position update representation

As shown in Figure 5, the swap operation is performed between the ω wolf and the three main wolves. The first condition is to ensure that the swap operation is active. Active means that the medoid that needs to be changed is not duplicated with one of the ω medoids. This condition is required because duplicated medoids produce wrong assignments by generating a smaller number of clusters than what was decided by the user. However, the swap operation works in sequence, that is, all medoids of three main wolves will be examined. Once improvement is observed, the operation stops finding the first improvement, otherwise, the operation continues in the current medoids. The full steps show in procedure medoids position update.

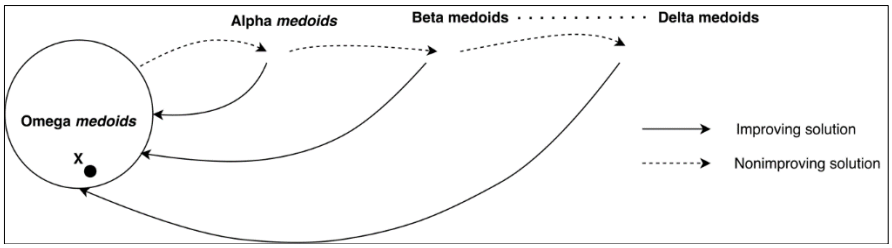

Figure 5. Medoids position swap operation

Procedure medoids position update

```
Initialize a new memory called Memory        % The Memory keeps only alpha, beta and
delta medoids, Initialize OM, NSMM, TMM where OM short memory keeps one medoid from
omega medoids, NSMM is amemory keeps not selected medoids from omega medoids, TMM is
temporary medoid memory keeps one medoids selected from Memory,
Memory← alpha, beta and delta medoids        % Copy the medoids into the memory
Generate M random number (1-k)                % generate random number, minimum
equals 1 and maximum equals k
Swapping operation starts
```

```
5:    Select one of omega medoids based on M value and calculate the fitness function f of
      the current assignment
6:    OM ← omega (M) % Copy the selected medoid into the omega memory medoid
7:    NSMM ← omega (Medoid! =M) % Copy the not selected medoid into the not selected
      medoid memory
8:    Status= false
9:    Count=0
10:    While (Count < Memory length) do
11:             TMM ←Select single medoid from Memory based on count value     % Copy the
          single   medoid into the temporary medoid memory
12:             if (NSMM != TMM)  then    % to ensure no duplicated medoids occurs
13:                  Status= true
14:            else
15:                 Status= false
                    Count=Count +1
16:           end-if
17:           if (Status= true) then
18:              Perform swap operation and calculate the new fitness function f* of the new
      soaution
19:          end-if
20:          if (f* < f) then
21:              Accept the swap operation and break the while loop
22:       else
23:            Count=Count +1
24:       end-if
25:   end-while
end-procedure
```

The producer begins by initializing all short memories, including $Memory$, OM, $NSMM$, and TMM. $Memory$ used to copy and keep all alpha, beta, and delta medoids, which are used later in the swap operation between the medoids in the memory and omega medoids. In Step 2, medoids from alpha, beta, and delta are copied into the memory followed by Step 3, which generates a random number between 1 and the maximin value k. Step 4 indicates the start of swapping operation between the omega medoids and the $Memory$, which keeps the alpha, beta, and delta medoids. In Steps 5 and 6, the omega medoid that is required to be changed based on the random number produced in previous step is selected and kept in OM memory, which is a short memory used to keep one medoid. Step 7 copies and keeps all remaining medoids that are not selected in a short memory called $NSMM$. The target of this memory is to ensure no medoids are duplicated in the selected medoids from the $Memory$. In Steps 8 and 9, the Status and count variables are initialized into false value and 0, respectively. The Status variable is used to check whether duplicated medoids are available or not. Meanwhile, the count variable is used in the loop to check all medoids in the $Memory$. In Step 10, the procedure starts its main loop by copying one medoid from the $Memory$ into the short memory TMM as shown in step 11. Check if there is no duplicated medoids between the omega medoids $NSMM$ and the medoid saved in TMM. The result is true if no duplication occurs, otherwise, the result is false, as shown in Steps 12–15. If the condition is true, then the swap operation is performed and accepted only if the new fitness f^* is better than the old fitness f, as shown in Steps 17–24. The output of this procedure is a set of medoids used in the next iteration to construct a new assignment.

4. PRE-PROCESSING TEXT FOR CLUSTERING

The text clustering begins with a data inspection phase, which is considered a pre-processing step wherein the data should be prepared. Data inspection considers the preparation of data, which includes data collection, data cleaning, and data representation. Data collection refers to the task of collecting the data that will be used and evaluated in this research. The benchmark datasets that will be used are those from UCI [50]. Before performing text clustering, a pre-processing step is required to convert the unstructured text into a structured format that is readable in clustering algorithms. The pre-processing includes tokenization, stemming of document words, and stop word removal [51], [52]. Tokenization corresponds to a general process that is used to break a text corpus into individual elements. Stemming is defined as the process of converting or transforming a word into a base word, that is, a word is converted into its original root form. The next step is to remove stopping words that are common and uninformative in a text corpus, such as so, and, or, and the. These stopping words, which are not important in information retrieval, are available in every document because they do not carry any information related to their documents. In text clustering, documents are not suitable in their original format unless represented in a suitable form. VSM is an effective representation model that treats documents as a bag-of-words and uses words as a measure to discover the similarity among documents [53]. Let D=$\{d_1, d_2, d_3 \dots \dots \dots d_n\}$ denote a collection of documents, and n is the

number of documents in that collection. Each document d represents a vector of terms, which is denoted as T=$\{t_1, t_2, t_3 \dots \dots \dots t_m\}$, where term t represents a word, and m is the number of terms. The document d has a relation with term t, which is denoted as term frequency (TF); TF can be represented by a fixed value that reflects the degree of term frequency in that document d [54]. However, based on the definition of VSM, it represents all the collection of document forms as a matrix where the rows are documents and columns are words, and each cell contains the TF value of the word within the document. Figure 6 shows the term frequency matrix.

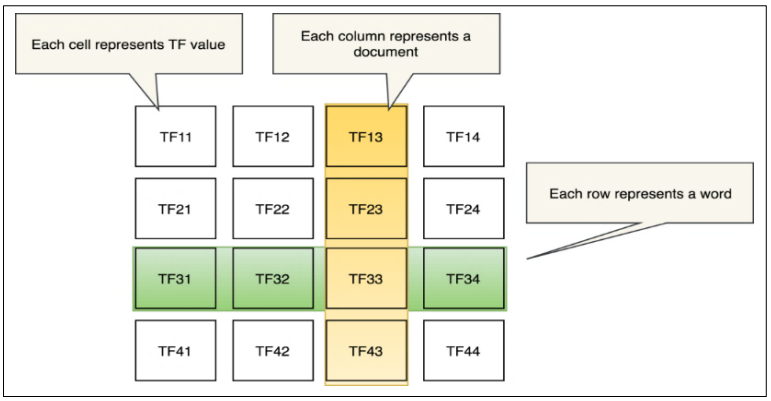

Figure 6. Term frequency matrix

To clarify the idea, let us suppose that three documents contain four terms, including class, university, book, and lecture, as described and shown in Table 1. Each document is related to its term count, which is the simplest choice of TF. However, other possible TF variants in the scheme, such as logarithmically scaled frequency, which is used in this research, can be used, as shown in (2) and applied in Table 2.

$$tf(t,d) = 1 + \log(tf_{t,d}) \quad (2)$$

Table 1. Term frequencies (counts)

Terms	Doc1	Doc 2	Doc 3
class	115	58	0
university	10	7	0
book	2	0	6
lecture	0	0	38

Table 2. Log frequency weighting

Terms	Doc 1	Doc2	Doc 3
Class	3.06	2.76	0
University	2.00	1.85	0
Book	1.30	0	1.78
Lecture	0	0	2.58

The next step involves inverse document frequency (IDF), which refers to the weight of each term with regard to the total number of documents (N) and the number of documents that contain that term in the collection (dft) as a score. IDF can be calculated using (3) and applied in Table 3.

$$idf = \log\left(\frac{N}{dft}\right) \quad (3)$$

Table 3. Log IDF weighting

Terms	IDF
Class	$\log\left(\frac{3}{2}\right) = 0.176$
University	$\log\left(\frac{3}{2}\right) = 0.176$
Book	$\log\left(\frac{3}{2}\right) = 0.176$
Lecture	$\log\left(\frac{3}{1}\right) = 0.477$

The next step is to obtain the weight of terms, which is calculated using (4) and applied in Table 4.

$$w_{t,d} = tf_{t,d} * idf_t \quad (4)$$

The next step is to perform normalization for the length of each term between (0,1). In (5) shows the normalization for the length and applied in Table 5.

$$N \frac{w_{t,d}}{\sqrt{\sum_1^m {w_{t,d}}^2}} \quad (5)$$

Finally, we calculate the similarity among documents using a cosine similarity score, as shown in (1) previously where $\overrightarrow{d_i}, \overrightarrow{d_j}$ are the document vectors, thereby producing a cosine similarity table among all documents, as shown in Table 6, and an example of the calculation of cosine similarly between documents 1 and 2 is provided, as shown before Table 6.

$$Cos(doc1, doc2) = \frac{0.9420415599999998}{0.9998889044322603} = 0.9421462282701233$$

Table 4. Weight of the term

Terms	Doc 1	Doc2	Doc 3
Class	0.5385	0.4857	0
University	0.352	0.3256	0
Book	0.2288	0	0.3132
Lecture	0	0	1.2306

Table 5. Length normalization

Terms	Doc1	Doc 2	Doc3
Class	0.7886	0.8306	0
University	0.5155	0.5568	0
Book	0.3350	0	0.2466
Lecture	0	0	0.9691

Table. 6 Cosine similarity table for Doc1, Doc2, and Doc3

	Doc1	Doc2	Doc3
Doc1	1	0.94	0.08
Doc2	0.94	1	0
Doc3	0.08	0	1

5. RESULTS AND DISCUSSION

Evaluation performance is an important step in data clustering to indicate the accuracy of the provided solutions. Three best known benchmarks used in classification and clustering field are used in the evaluation process. The first benchmark, denoted as 20NewsGroup, was also obtained from the UCI. The 20Newsgroup used in this research contains 2000 documents distributed into 20 classes where 100 documents in each class. This research has selected three classes: hardware, baseball, and electronic with a total number of 300 documents. The number of terms collected after pre-processing step is 22, which includes the most importing terms among the selected three classes. The second benchmark, which is denoted as Reuters-21578, was obtained from UCI. This benchmark includes five categories, including exchanges, people, topics, organizations, and places. Each of the categories is divided into subcategories. Four subcategories used in the evaluation include three classes with a total number of 1907 documents, in which 1650 documents belong to acq class, 37 documents belong to gas class, 94 documents belong to gold class, and 126 documents belong to sugar. These subcategories are selected because the number of documents in each class differs from the other group. Thus, the problem of finding the best configuration is difficult, especially if the benchmark contains different densities. The number of terms collected after pre-processing step is 10, which includes the most important terms among the documents of all classes. The third benchmark includes web pages plus the ratings of a single Syskill and Webert web page ratings, which are denoted as SyskillWebert benchmark also obtained UCI with four different kinds of documents, including Bands, Bio Medial, Goats, and Sheep. The last benchmark used in this research is sentence classification benchmark, which is known as the SentenceCorpus. It represents the abstract of articles collected in different publishers. In this study, the two publishers used are the Jam and Plos, where the maximum number of documents are 600. The number of terms collected after the pre-processing step is 80 terms with two classes. Table 7 describes all benchmarks used in this research, which includes 13 different kinds of class.

Table 7. Characteristics of benchmarks

Benchmark	Resources	Number of documents	Total number of documents	Number of terms	Number of classes
20 newsgroups	Baseball	100	300	22	3
	Electronics	100			
	Hardware	100			
Reuters-21578	Acq	1650	1907	10	4
	Gas	37			
	Gold	94			
	Sugar	126			
SyskillWebert	Bands	61	290	5	4
	Bio Medial	124			
	Goats	57			
	Sheep	48			
SentenceCorpus	Jdm	300	600	80	2
	Plos	300			

This research evaluates the text clustering results using the external evaluation criteria, including Rand, Purity, F-Measure, and Entropy. Rand measures the accuracy of text clustering assignment based on the similarity between the documents, as shown in (6). In (6), different metrics required, including true positive (TP), false positive (FP), false negative (FN), and final true negative (TP), should be calculated.

$$Rand = \frac{TP+TN}{TP+FP+FN+TN} \quad (7)$$

Purity refers to the percent of the total number of objects (data points) that were classified correctly in the unit range [0...1] and can be calculated as shown in (7). These criteria compute the pairs of documents placed in the same clusters. Frequent documents in the same clusters are considered better clustering assignments.

$$\text{Purity}(\Omega, \Gamma) = \frac{1}{N}\sum_k maxj\left|w_i \cap c_j\right| \quad (8)$$

Where Ω = w1, w2, …; wk is the set of clusters; and Γ = c1, c2, … , cj is the set of classes.

The other external criteria include F-measure, which measures the accuracy of separation and compactness between clusters. The algorithm is considered the best if the value of F-measure is high. However, before calculating the F-measure, we need to calculate two more measures, which are Precision and Recall, as shown in (8) and (9), respectively. Thus, the F-measure can be calculated as shown in (10).

$$Precision = \frac{TP}{TP+FP} \quad (8)$$

$$Recall = \frac{TP}{TP+FN} \quad (9)$$

$$\text{F} - \text{ Measure} = \frac{(\beta^2+1)Precision*Recall}{\beta^2 Precision+Recall} \quad (10)$$

Where β is a constant that can be used to penalize false negatives more strongly than false positives by selecting its value > 1, thereby giving more weight to Recall.

To show how documents are distributed in each cluster, this research uses other external evaluation criteria called entropy. This criterion measures the compactness of documents in each cluster. The output of the criteria is considered the best if a small value is produced by the algorithm. In (11) shows the measuring of entropy for single clustering w [27]. The total entropy of a clustering is calculated in (12).

$$H(w) = -\sum_{c\in C} P(w_c) log_2 P(w_c) \quad (11)$$

Where c is a classification in the set C of all classification, $P(w_c)$ is the probability of a data point being classified as c in cluster w, and the total entropy of a cluster is:

$$H(\Omega) = \sum_{w\in\Omega} H(w)\frac{N_w}{N} \quad (12)$$

where $\mathrm{H(w)}$ is a single cluster's entropy, $\mathrm{N_w}$ is the number of points in cluster w, and N is the total number of points.

The evaluation of the proposed algorithm performed with the best-known optimization algorithms include genetic algorithm (GA) [55], firefly algorithm (FA) [56], GWO [49], and k-means (KM) [57]. The setting of the fixed parameters is based on the literature, where the population size and number of iterations with the number of runs are the same in all optimization algorithms as shown in Table 8. The crossover operator in GA is set to 0.8, which represents the high probability of mating between the individuals. Mutation is set to a small value of 0.001, which represents the exploration of solutions. In FA, the light absorption corresponds to the fitness function of the firefly, and initial attractiveness refers to the improvement of solution in a given radius. In GWO algorithm, the coefficient is set to 2, which changes linearly. Meanwhile, in M-GWO, this coefficient has been removed from the algorithm. Only the KM algorithm is set to 1000 iterations because it converges to the local optima quickly.

The evaluation is performed in two scenarios. The first scenario shows a comparison using all algorithms (overall performance), whereas the second scenario shows a comparison between the proposed algorithm and other algorithms (algorithm vs. algorithm). The results shown in Figures 7, 8, 9, and 10 indicate that the proposed algorithm, M-GWO, produced the best results in the three benchmarks, including

Reuters-21578, in all evaluation criteria. The second benchmark, which is SentenceCorpus, also in all evaluation criteria while in 20 newsgroups in two evaluation criteria are rand and entropy metric. In SyskillWebert benchmark, M-GWO and KM produced the same ranking (i.e., 50%). The former produced the best results in F-Measure and Entropy metrics. Meanwhile, KM produced the best results in Rand and Purity metrics. The summary of this scenario shows that the proposed M-GWO algorithm is 75% better than all other algorithms, including GWO, FA, GA, and KM.

Table 8. Parameters of all algorithms

GA	FA	KM	GWO / M-GWO
Population size =50	Population size =50	Iterations =1000	Population size =50
Iterations =1000	Iterations =1000	Runs =10	Iterations =1000
Runs =10	Runs =10		Runs =10
Crossover = 0.8	Initial attractiveness (B_0) = 0		Coefficient =2/NA
Mutation rate =0.001	Light absorption (y) = 1.0		

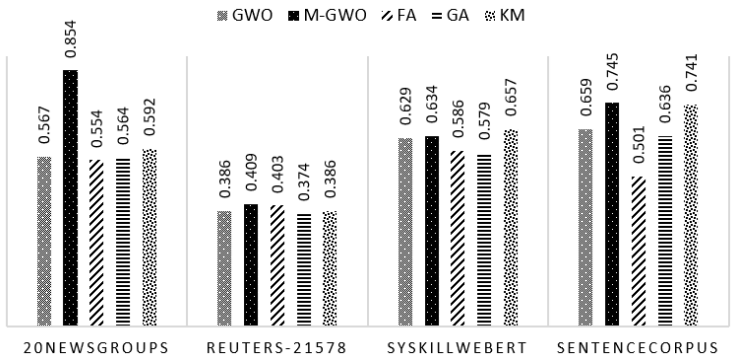

Figure 7. Quality metric of clustering (rand metric) of M-GWO vs. best known algorithms

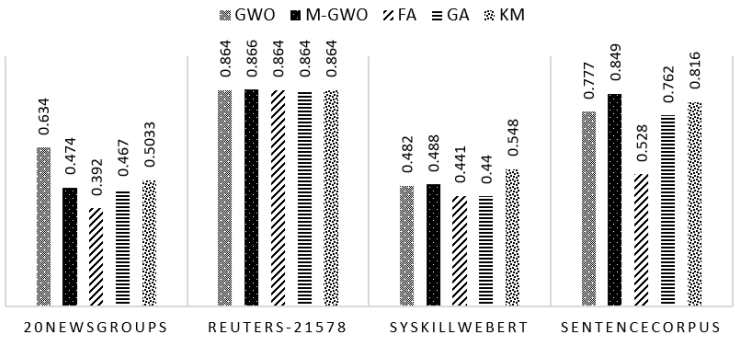

Figure 8. Quality metric of clustering (purity metric) of M-GWO vs. best known algorithms

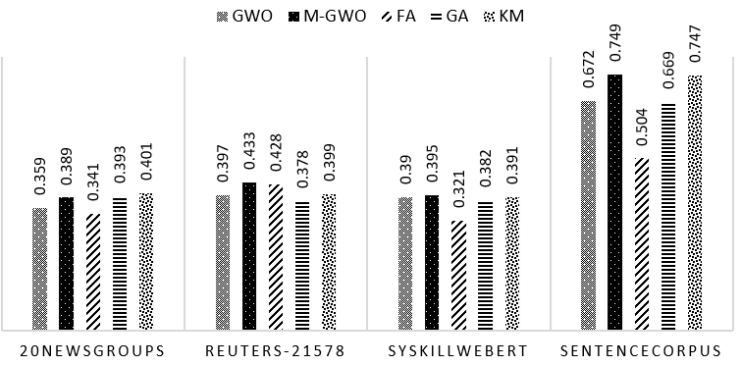

Figure 9. Quality metric of clustering (F- Measure metric) of M-GWO vs. best known algorithms

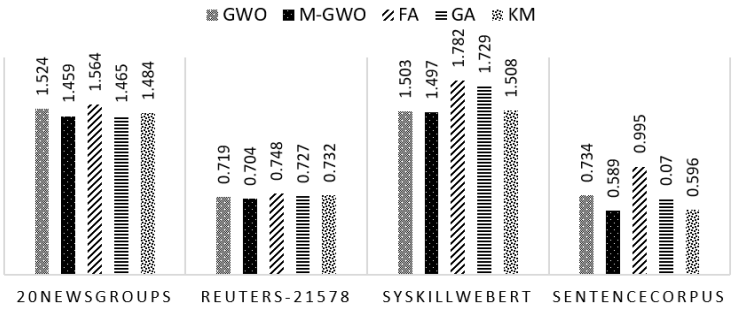

Figure 10. Quality metric of clustering (entropy metric) of M-GWO vs. best known algorithms

Four comparisons in the second scenario are shown in Figures 7, 8, 9, and 10 based on the evaluation criteria, including Rand, Purity, F-Measure, and Entropy. The first comparison is performed between M-GWO and all algorithms based on the Rand metric produced by each algorithm in all benchmarks. Figure 7 indicates that the M-GWO produced the best results in 20 newsgroups, Reuters-21578, and SentenceCorpus, suggesting that it is 75% better than all other algorithms. The second comparison based on Purity metric is shown in Figure 8, suggesting that the M-GWO produced the best results in Reuters-21578 and SentenceCorpus (50%), whereas GWO and KM only produced the best result in 1 benchmark, namely 20 newsgroups (25%) and SyskillWebert (25%), respectively. The F-Measure metric in Figure 9 shows that the best result is produced by the M-GWO algorithm in Reuters-21578, SyskillWebert, and SentenceCorpus (25%). Meanwhile, only 20 newsgroup benchmarks were produced by the KM algorithm. The Entropy metric shown in Figure 10 indicates that the M-GWO algorithm produced the best result in all benchmarks, suggesting that the M-GWO algorithm can produce better compact text clustering in each cluster.

The results produced based on all external evaluation metrics, including Rand, Purity, F-Measure, and Entropy, indicated that the proposed algorithm can construct clustering solutions that are compact and well separated. This result is achieved because the proposed algorithm can handle datasets with outlier and noise, thereby minimizing the errors produced. Furthermore, the results show that KM algorithm produced better results than the proposed algorithm in some benchmarks because the number of iterations in the side algorithm are set to 1000 iterations, thereby proving higher probability to find more clustering results.

6. CONCLUSION

This research aims to solve the problem of GWO algorithm in text clustering when the algorithm is sensitive to outlier objects and noise data. However, text clustering needs more consideration because the calculation of the similarity between the documents are based on the extracted terms in all documents. Thus, some documents that lack important terms that appears in most documents are considered outlier documents. This result is achieved when the algorithm produced the text clustering according to centroid-based properties, which easily affect by outlier documents and noise data. This drawback led to the proposal of M-GWO with two contributions, namely, medoid-based clustering instated using centroid-based clustering and medoid recalculation process. The first improvement re-generated the text clustering assignment based on medoids (data instance) and not based on centroids, in which the properties of the search space are required to determine the central point. The second modification is the medoid recalculation process, which improves the search process of the M-GWO algorithm by swapping one medoid in each iteration. The swap operation is performed between each agent and the main three wolves in the population. Both modifications enhance the algorithm when the outlier document appears and make the algorithm more robust to noise effect.

This improvement can enhance the search process to find only the best set of medoids and improve the exploitation search to discover more optimal text clustering solutions in the local regains of the search space. However, the algorithm still has a limitation. For example, the algorithm cannot find the best number of clusters. Furthermore, the exploration search needs to be improved by looking not only for the best medoids in the best three wolves but also by exploring the search space such that some medoids are integrated randomly. Future studies will focus on improving the algorithm by adding a new parameter called outlier factor to calculate and score each document given their anomaly degrees. This approach will enhance the algorithm through the removal of those documents, thereby increasing the accuracy of text clustering.

ACKNOWLEDGEMENTS

The author would like to express his thanks to Shatt Al-Arab University College, for the financial assistance.

REFERENCES

[1] A. M. Jabbar, R. Sagban, and K. R. Ku-Mahamud, "Balancing exploration and exploitation in acs algorithms for data clustering," *Journal of Theoretical and Applied Information Technology*, vol. 97, no. 16, pp. 4320–4333, 2019.

[2] H. N. K. Al-Behadili, R. Sagban, and K. R. Ku-Mahamud, "Adaptive parameter control strategy for ant-miner classification algorithm," *Indonesian Journal of Electrical Engineering and Informatics*, vol. 8, no. 1, pp. 149–162, 2020, doi: 10.11591/ijeei.v8i1.1423.

[3] H. N. K. Al-Behadili, K. R. Ku-Mahamud, and R. Sagban, "Hybrid ant colony optimization and iterated local search for rules-based classification," *J. Theor. Appl. Inf. Technol.*, vol. 98, no. 4, pp. 657–671, 2020.

[4] H. N. K. AL-Behadili, K. R. Ku-Mahamud, and R. Sagban, "Hybrid ant colony optimization and genetic algorithm for rule induction," *J. Comput. Sci.*, vol. 16, no. 7, pp. 1019–1028, 2020, doi: 10.3844/jcssp.2020.1019.1028.

[5] A. M. F. Al-Sbou, "A survey of arabic text classification models," *International Journal of Informatics and Communication Technology*, vol. 8, no. 1, pp. 25-28, 2019, doi: 10.11591/ijict.v8i1.pp25-28.

[6] A. A. hussian Hassan, W. M. Shah, M. F. I. Othman, and H. A. H. Hassan, "Evaluate the performance of K-Means and the fuzzy C-Means algorithms to formation balanced clusters in wireless sensor networks," *International Journal of Electrical and Computer Engineering*, vol. 10, no. 2, pp. 1515–1523, 2020, doi: 10.11591/ijece.v10i2.pp1515-1523.

[7] R. A. Hasan, R. A. Ibrahem Alhayali, N. D. Zaki, and A. H. Ali, "An adaptive clustering and classification algorithm for twitter data streaming in Apache Spark," *TELKOMNIKA (Telecommunication, Computing, Electronics and Control)*, vol. 17, no. 6, pp. 3086–3099, 2019, doi: 10.12928/telkomnika.v17i6.11711.

[8] A. M. Jabbar, K. R. Ku-Mahamud, and R. Sagban, "Modified ACS Centroid Memory for Data Clustering," *J. Comput. Sci.*, vol. 15, no. 10, pp. 1439–1449, 2019, doi: 10.3844/jcssp.2019.1439.1449.

[9] A. M. Jabbar, K. R. Ku-Mahamud, and R. Sagban, "An improved ACS algorithm for data clustering," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 17, no. 3, pp. 1506–1515, 2019, doi: 10.11591/ijeecs.v17.i3.pp1506-1515.

[10] H. Singh, Y. Kumar, and S. Kumar, "A new meta-heuristic algorithm based on chemical reactions for partitional clustering problems," *Evol. Intell.*, vol. 12, no. 2, pp. 241-252, 2019, doi: 10.1007/s12065-019-00221-w.

[11] S. Zhou and Z. Xu, "A novel internal validity index based on the cluster centre and the nearest neighbour cluster," *Appl. Soft Comput. J.*, vol. 71, pp. 78–88, 2018, doi: 10.1016/j.asoc.2018.06.033.

[12] P. Fränti and S. Sieranoja, "How much can k-means be improved by using better initialization and repeats?," *Pattern Recognit.*, vol. 93, pp. 95–112, 2019, doi: 10.1016/j.patcog.2019.04.014.

[13] E. G. Mansoori, "GACH: A grid-based algorithm for hierarchical clustering of high-dimensional data," *Soft Computing*, vol. 18, no. 5, pp. 905-922, 2014, doi: 10.1007/s00500-013-1105-8.

[14] W. Liao, "A Grid-based Clustering Algorithm using Adaptive Mesh Refinement," *Proc. 7th Work. Min. Sci. Eng. Datasets*, vol. 22, 2004, pp. 61-69.

[15] C. Bouveyron and C. Brunet-Saumard, "Model-based clustering of high-dimensional data: A review," *Comput. Stat. Data Anal.*, vol. 71, pp. 52–78, 2014, doi: 10.1016/j.csda.2012.12.008.

[16] G. Karthikeyan and P. Balasubramanie, "Optimal feature based density clustering for outlier detection," *International Journal of Civil Engineering and Technology*, vol. 8, no. 9, pp. 520–538, 2017.

[17] E. Liao and C. Liu, "A Hierarchical Algorithm Based on Density Peaks Clustering and Ant Colony Optimization for Traveling Salesman Problem," in *IEEE Access*, vol. 6, pp. 38921-38933, 2018, doi: 10.1109/ACCESS.2018.2853129.

[18] M. D. Arimbi, A. Bustamam, and D. Lestari, "Implementation of Hybrid Clustering Based on Partitioning Around Medoids Algorithm and Divisive Analysis on Human Papillomavirus DNA," *4th International Symposium on Biomathematics, SYMOMATH 2016*, 2017, doi: 10.1063/1.4978974.

[19] Z. Chen and S. Xia, "K-means Clustering Algorithm with Improved Initial Center," *2009 Second International Workshop on Knowledge Discovery and Data Mining*, 2009, pp. 790-792, doi: 10.1109/WKDD.2009.210.

[20] A. Kaur, P. Kumar, and P. Kumar, "Effect of noise on the performance of clustering techniques," *2010 Int. Conf. Networking and Information Technology*, 2010, pp. 504-506, doi: 10.1109/ICNIT.2010.5508461.

[21] S. K. Popat and M. Emmanuel, "Review and Comparative Study of Clustering Techniques," *Int. J. Comput. Sci. Inf. Technol.*, vol. 5, no. 1, pp. 805–812, 2014.

[22] K. El Bouchefry and R. S. de Souza, "Learning in Big Data: Introduction to Machine Learning," *Know. Disc. Big Data from Astronomy and Earth Observation*, pp. 225-249, 2020, doi: 10.1016/B978-0-12-819154-5.00023-0.

[23] P. Van Der Spek and S. Klusener, "Applying a dynamic threshold to improve cluster detection of LSI," *Sci. Comput. Program.*, vol. 76, no. 12, pp. 1261–1274, 2011, doi: 10.1016/j.scico.2010.12.004.

[24] N. A. Mokhtar, Y. Z. Zubairi, and A. G. Hussin, "A clustering approach to detect multiple outliers in linear functional relationship model for circular data," *J. Appl. Stat.*, vol. 45, no. 6, pp. 1041-1051, 2017, doi: 10.1063/1.4982835.

[25] J. A. S. Almeida, L. M. S. Barbosa, A. A. C. C. Pais, and S. J. Formosinho, "Improving hierarchical cluster analysis: A new method with outlier detection and automatic clustering," *Chemom. Intell. Lab. Syst.*, vol. 87, no. 2, pp. 208–217, 2007, doi: 10.1016/j.chemolab.2007.01.005.

[26] P. Liu, Y. Liu, X. Hou, Q. Li, and Z. Zhu, "A Text Clustering Algorithm Based on Find of Density Peaks," *2015 7th Int. Conf. Inf. Tech. Medicine and Education (ITME)*, 2015, pp. 348-352, doi: 10.1109/ITME.2015.103.

[27] M. H. Chehreghani, H. Abolhassani, and M. H. Chehreghani, "Improving density-based methods for hierarchical clustering of web pages," *Data Knowl. Eng.*, vol. 67, no. 1, pp. 30–50, 2008, doi: 10.1016/j.datak.2008.06.006.

[28] W. Zhang, T. Yoshida, X. Tang, and Q. Wang, "Text clustering using frequent itemsets," *Knowledge-Based Syst.*, vol. 23, no. 5, pp. 379–388, 2010, doi: 10.1016/j.knosys.2010.01.011.

[29] V. V. Thang, D. V. Pantiukhin, and A. I. Galushkin, "A Hybrid Clustering Algorithm: The FastDBSCAN," *2015 Int. Conf. Engineering and Telecommunication (EnT)*, 2015, pp. 69-74, doi: 10.1109/EnT.2015.31.

[30] S. Zhou, Z. Xu, and F. Liu, "Method for Determining the Optimal Number of Clusters Based on Agglomerative

Hierarchical Clustering," in *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 12, pp. 3007-3017, Dec. 2017, doi: 10.1109/TNNLS.2016.2608001.
[31] K. Murugesan and J. Zhang, "Hybrid Bisect K-Means Clustering Algorithm," *2011 International Conference on Business Computing and Global Informatization*, 2011, pp. 216-219, doi: 10.1109/BCGIn.2011.62.
[32] I. Gurrutxaga *et al.*, "SEP/COP: An efficient method to find the best partition in hierarchical clustering based on a new cluster validity index," *Pattern Recognit.*, vol. 43, no. 10, pp. 3364–3373, 2010, doi: 10.1016/j.patcog.2010.04.021.
[33] R. Syal, "A Novel Hybrid Clustering Algorithm : Integrated Partitional and Hierarchical Clustering Algorithm for Categorical Data," *International Journal Computer Science & Emerging Technologies*, vol. 3, no. 5, pp. 138–146, 2012.
[34] O. Tanaseichuk *et al*., "An Efficient Hierarchical Clustering Algorithm for Large Datasets," *Austin J. Proteomics, Bioinforma. Genomics*, vol. 2, no. 1, pp. 1–6, 2015.
[35] S. V. Lahane, M. U. Kharat, and P. S. Halgaonkar, "Divisive Approach of Clustering for Educational Data," *2012 Fifth Int. Conf. Emer. Trends in Engineering and Technology*, 2012, pp. 191-195, doi: 10.1109/ICETET.2012.55.
[36] H. D. Menéndez, F. E. B. Otero, and D. Camacho, "Medoid-based clustering using ant colony optimization," *Swarm Intell.*, vol. 10, no. 2, pp. 123–145, 2016, doi: 10.1007/s11721-016-0122-5.
[37] H. D. Menéndez, F. E. B. Otero, and D. Camacho, "MACOC: A Medoid-Based ACO Clustering Algorithm," in *Swarm Intelligence*, pp. 122-133, 2014, doi: 10.1007/978-3-319-09952-1_11.
[38] J. Muthuswamy and B. Kanmani, "Optimization Based Liver Contour Extraction of Abdominal CT Images," *International Journal of Electrical and Computer Engineering*, vol. 8, no. 6, pp. 5061-5070, 2018, doi: 10.11591/ijece.v8i6.pp5061-5070.
[39] K. Rajesh and N. Visali, "Hybrid method for achieving Pareto front on economic emission dispatch," *International Journal of Electrical and Computer Engineering*, vol. 10, no. 4, pp. 3358–3366, 2020, doi: 10.11591/ijece.v10i4.pp3358-3366.
[40] A. A. Al-Arbo and R. Z. Al-Kawaz, "Implementation of combined new optimal cuckoo algorithm with a gray Wolf algorithm to solve unconstrained optimization nonlinear problems," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 19, no. 3, pp. 1582–1589, 2020, doi: 10.11591/ijeecs.v19.i3.pp1582-1589.
[41] M. Alzaqebah, N. Alrefai, E. A. E. Ahmed, S. Jawarneh, and M. K. Alsmadi, "Neighborhood search methods with moth optimization algorithm as a wrapper method for feature selection problems," *International Journal of Electrical and Computer Engineering*, vol. 10, no. 4, pp. 3672–3684, 2020, doi: 10.11591/ijece.v10i4.pp3672-3684.
[42] P. Matrenin *et al.*, "Generalized swarm intelligence algorithms with domain-specific heuristics," *IAES International Journal of Artificial Intelligence*, vol. 10, no. 1, pp. 157–165, 2021, doi: 10.11591/ijai.v10.i1.pp157-165.
[43] M. Z. M. Tumari, M. M. Zahar, and M. A. Ahmad, "Optimal tuning of a wind plant energy production based on improved grey wolf optimizer," *Bulletin of Electrical Engineering and Informatics*, vol. 10, no. 1, pp. 23–30, 2020, doi: 10.11591/eei.v10i1.2509.
[44] I. A. Saleh, O. I. Alsaif, and M. A. Yahya, "Optimal distributed decision in wireless sensor network using gray wolf optimization," *IAES International Journal of Artificial Intelligence*, vol. 9, no. 4, pp. 646–654, 2020, doi: 10.11591/ijai.v9.i4.pp646-654.
[45] A. M. Putri, Z. Rustam, J. Pandelaki, I. Wirasati, and S. Hartini, "Acute sinusitis data classification using grey wolf optimization-based support vector machine," *IAES International Journal of Artificial Intelligence*, vol. 10, no. 2, pp. 438–445, 2021, doi: 10.11591/ijai.v10.i2.pp438-445.
[46] M. T. Le, M. T. Vo, N. T. Pham, and S. V. T. Dao, "Predicting heart failure using a wrapper-based feature selection," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 21, no. 3, pp. 1530–1539, 2021, doi: 10.11591/ijeecs.v21.i3.pp1530-1539.
[47] A. A. Alomoush, A. R. A. Alsewari, K. Z. Zamli, A. Alrosan, W. Alomoush, and K. Alissa, "Enhancing three variants of harmony search algorithm for continuous optimization problems," *International Journal of Electrical and Computer Engineering*, vol. 11, no. 3, pp. 2343–2349, 2021, doi: 10.11591/ijece.v11i3.pp2343-2349.
[48] H. M. Asgher, Y. M. Mohmad Hassim, R. Ghazali, and M. Aamir, "Improved grey wolf algorithm for optimization problems," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 22, no. 3, pp. 1573–1579, 2021, doi: 10.11591/ijeecs.v22.i3.pp1573-1579.
[49] H. Rashaideh *et al.*, "A Grey Wolf Optimizer for Text Document Clustering," *J. Intell. Syst.*, vol. 29, no. 1, pp. 814–830, 2018, doi: 10.1515/jisys-2018-0194.
[50] K. Bache and M. Lichman, UCI Machine Learning Repository, 2013.
[51] N. M. N. Mathivanan, N. A. Nor, and R. M. Janor, "Improving classification accuracy using clustering technique," *Bulletin of Electrical Engineering and Informatics*, vol. 7, no. 3, pp. 465–470, 2018, doi: 10.11591/eei.v7i3.1272.
[52] N. Katuk, K. R. Ku-Mahamud, N. H. Zakaria, and A. M. Jabbar, "A scientometric analysis of the emerging topics in general computer science," *J. Inf. Com. Tech.*, vol. 19, no. 4, pp. 583–622, 2020, doi: 10.32890/jict2020.19.4.6.
[53] K. Subhadra, M. Shashi, and A. Das, "Extended ACO based document clustering with hybrid distance metric," *2015 IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT)*, 2015, pp. 1-6, doi: 10.1109/ICECCT.2015.7226090.
[54] L. H. Patil and M. Atique, "A novel approach for feature selection method TF-IDF in document clustering," *2013 3rd IEEE Int. Advance Computing Conference (IACC)*, 2013, pp. 858-862, doi: 10.1109/IAdCC.2013.6514339.
[55] D. Mustafi, A. Mustafi, and G. Sahoo, "A novel approach to text clustering using genetic algorithm based on the nearest neighbour heuristic," *Int. J. Comput. Appl.*, 2020, doi: 10.1080/1206212X.2020.1735035.
[56] J. Senthilnath, S. N. Omkar, and V. Mani, "Clustering using firefly algorithm: Performance study," *Swarm Evol. Comput.*, vol. 1, no. 3, pp. 164–171, 2011, doi: 10.1016/j.swevo.2011.06.003.
[57] T. Niknam and B. Amiri, "An efficient hybrid approach based on PSO, ACO and k-means for cluster analysis," *Appl. Soft Comput. J.*, vol. 10, no. 1, pp. 183–197, 2010, doi: 10.1016/j.asoc.2009.07.001.