Enhanced feature clustering method based on ant colony optimization for feature selection

Hassan Almazini [a,1,*], Ku Ruhana Ku-Mahamud [b,c,2], Hussein Fouad Almazini [a,3]

[a] Department of Computer Technology Engineering, Shatt Al-Arab University College, Basra, Iraq
[b] School of Computing, University Utara Malaysia, Kedah, Malaysia
[c] Shibaura Institute of Technology, Tokyo, Japan
[1] hassan.f.abbas@sa-uc.edu.iq ; [2] ruhana@uum.edu.my; [3] hussein.f.abbas@sa-uc.edu.iq
* corresponding author

ARTICLE INFO

Article history
Received January 9, 2023
Revised February 7, 2023
Accepted February 7, 2023
Available online March 31, 2023

Keywords
Feature clustering
Correlated features
Local search
Classification
Microarray

ABSTRACT

The popular modified graph clustering ant colony optimization (ACO) algorithm (MGCACO) performs feature selection (FS) by grouping highly correlated features. However, the MGCACO has problems in local search, thus limiting the search for optimal feature subset. Hence, an enhanced feature clustering with ant colony optimization (ECACO) algorithm is proposed. The improvement constructs an ACO feature clustering method to obtain clusters of highly correlated features. The ACO feature clustering method utilizes the ability of various mechanisms, such as local and global search to provide highly correlated features. The performance of ECACO was evaluated on six benchmark datasets from the University California Irvine (UCI) repository and two deoxyribonucleic acid microarray datasets, and its performance was compared against that of five benchmark metaheuristic algorithms. The classifiers used are random forest, k-nearest neighbors, decision tree, and support vector machine. Experimental results on the UCI dataset show the superior performance of ECACO compared with other algorithms in all classifiers in terms of classification accuracy. Experiments on the microarray datasets, in general, showed that the ECACO algorithm outperforms other algorithms in terms of average classification accuracy. ECACO can be utilized for FS in classification tasks for high-dimensionality datasets in various application domains such as medical diagnosis, biological classification, and health care systems.

1. Introduction

Feature selection (FS) is important for exploratory analysis of high-dimensional data (e.g., microarray datasets) and is an effective method of unearthing previously untapped knowledge obtained from classification [1]. However, FS ignores the potential for feature correlation. Thus, the generated subsets may not be optimal for the particular task. Furthermore, FS relies on certain mathematical concepts without guaranteeing that these concepts are universally valid for all data [2].

Feature selection methods are classified into three main groups based on their relationship with the learning model: filter, wrapper, and hybrid approaches [3]. The filter method requires no learning algorithms for evaluation, and it is based on statistical feature information. Therefore, it is fast in computation and efficient [4]. Wrapper methods utilize a learning algorithm in selecting feature subset and, generally, the learning efficiency is increased at the cost of higher computational complexity. The hybrid methods make use of the benefits from wrapper and filter methods to achieve a balance, combining the efficiency of learning and the execution time. The solutions of the filter-based methods

are worth indicating, where its use of the evaluation criteria with the class label, leads to enhanced correlation among features and reduced similarity among features [5].

One of the important evaluation criteria in the correlation between features is relevance and redundancy analysis. The classical evaluation criterion that is based on the analysis of relevance and redundancy is max-relevance and min-redundancy [6]. In the processing of selection features, several mechanisms are able to reduce both relevance and redundancy. Redundancy occurs when one feature provides information similar to another feature in the dataset. Therefore, choosing the redundant feature to represent the final subset negatively affects model accuracy. Irrelevant features play a meaningless role and do not supply information to the clustering or classification of a set of data instances. In high-dimensional data, the correlation between the increasing numbers of features becomes more complicated. Consequently, redundant and irrelevant features should be analyzed more accurately. Several FS methods, such as max-relevance and min-redundancy, random subspace [7], fast correlation-based filter [8], and relevance-redundancy feature selection [9], partly consider this issue, leading to unsatisfactory results for the correlation between features and the elimination of redundant features [10]. Feature redundancy is one of the major challenges of these methods even though they take into account the correlation among features. These methods often have high computational complexity that is not effective for FS of high-dimensionality data.

Investigations on FS methods have focused on the diversification between features. Such a method can express max relevance and max diversity as two optimization problems. The basic process for these methods consists of three stages. In the first stage, the correct distance measure is selected to form the feature space, while the second stage focuses on clustering of features. In the third stage, representative features from each cluster are selected to form the desired subset of features. The FS method becomes more efficient with the utilization of cluster information [10]. However, the selection of features from each cluster presents some challenges, as with many of these works, because the clustering approach is overly sensitive in the FS algorithm models [10].

One of the most effective FS algorithms to select subsets of candidate features is based on a search technique. The search technique breaks down FS methods into heuristic search, complete, and randomized. Complete search algorithms involve searching the total search space for the best feature subset. The complete search aims to select the best subset of features in a dataset with high dimensionality, which is practically impossible within a plausible time [10]. A randomized search strategy investigates an entire search space through a finite space, and the subspace scale is based on stopping criteria, such as the maximum number of iterations and the subset size. The randomized search algorithm still appears to become stuck in a local optimum, even when it uses the setting parameters to trade between the optimality of the result and convergence speed. Such algorithms have lower computational complexity than complete search algorithms [10]. In heuristic search-based FS algorithms, one feature is removed or added from the selected feature set in each iteration. Moreover, the computational complexity of the heuristic search-based algorithm is much less than that of complete search algorithms. Most algorithms have been constructed based on heuristic searches.

Techniques based on swarm intelligence as a search technique for FS include ant colony optimization (ACO) [11], grey wolf optimizer algorithm [12], artificial bee colony [13], and whale optimization algorithm [14] . Metaheuristic swarm search algorithms are highly beneficial due to their global search capabilities in high-dimensional data. [15]. For example, ACO as a metaheuristic swarm search technique for FS is commonly utilized [1]. It has various advantages over other artificial intelligence techniques, including the ability to long-term distributed memory, perform local and global searches, and learn reinforcement mechanisms [11]. Thus, ACO is suitable for dealing with high-dimensional, noisy, irrelevant, and redundant datasets when it comes to FS.

Many real problems can be modeled into a graph form, such as the traveling salesman problem [16], graph coloring problem [17], and FS for microarray [18]. In FS for microarray, the graph-based step

creates multiple views, each containing a specific number of automatically obtained genes. This is natural in genomic data where gene groups are important in deciding alternate definitions of the microarray data with regard to diversity, thereby facilitating the collection of gene subsets that are informative genes with regard to different views.

One powerful FS method that considers the correlation between features to enhance the search for optimal feature subset is feature clustering. This method can minimize the dimensionality in high-dimensional data by grouping the most highly associated features together. One of the most effective method for clustering features is modified graph clustering-based ACO (MGCACO), which is a variation of the graph clustering-based ACO algorithm (GCACO) [19]. In comparison to well-known FS methods, the GCACO results are better for high-dimensional datasets because each iteration of the ant search process includes selecting at least one feature from each cluster. The selection process drives the injection of relatively less correlated features in a significant percentage compared with features in the following iteration that are highly correlated.

The Louvain community detection method [20] is used for both the GCACO and MGCACO methods and determines the local maximum by maximizing a modularity function and identifying the communities with the most correlated features. When determining the communities in large networks, the greedy search method integrated into GCACO and MGCACO uses modularity maximization as an objective to obtain the superior community. These algorithms are straightforward and simple to apply [21]. Thus, MGCACO outperforms other well-known FS methods. Notwithstanding, several disadvantages exist, such as the lack of clustering highly correlated features that could reduce the finding of the optimal subset, therefore decreasing the performance of the MGCACO algorithm. Therefore, this study replaced the Louvain community detection method with an ACO-based feature clustering algorithm that utilizes the ability of various mechanisms, such as intensification and diversification for local and global optimization, to enhance the performance of the MGCACO algorithm [22].

In summary, this research's contribution is as follows:

- To propose an ACO-based feature clustering method that can be used to enhance the grouping of the highly correlated features into the same cluster. Through the use of the ACO-based feature clustering approach, the possibility of falling into the local optima is reduced because many ants search for the best solution concurrently and stochastically. Each ant will gradually form its own cluster centers for calculating heuristic values at an iteration level. Ants consider not only the pheromone levels but also the heuristic values of candidate nodes when selecting the next feature, thus enabling them to find better solutions (i.e., groups of features).
- To investigate whether the suggested method has better classification accuracy via experiments and measurements utilizing four classifiers.

This paper is structured as follows: Section 1 presents the introduction. Section 2 explains the method. Section 3 describes the results of the experiments, and Section 4 is the conclusion

2. Method

This study proposes an algorithm based on MGCACO, which is a variant of the GCACO algorithm. The MGCACO algorithm is easily trapped into the local optimum during the feature clustering stage, thus losing the diversification in the search (global search) and producing a low-quality subset of features. To guarantee the precision of the feature's subset, a metaheuristic algorithm is utilized to provide intensification and diversification in the search process. Fig. 1 shows the enhanced feature clustering with the ECACO algorithm.

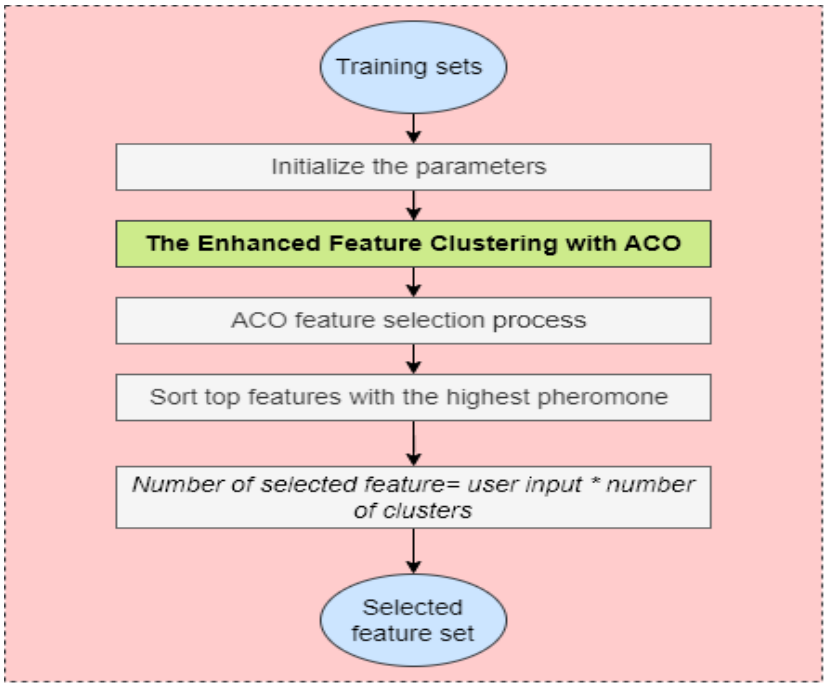

Fig. 1. ECACO algorithm

Section 2.1 presents the ECACO algorithm. Section 2.2 introduces the FS process to construct the appropriate subset of features. Section 4.3 lists in detail the different benchmark datasets classification problems. Finally, Section 2.4 illustrates the classifier and evaluation criteria.

2.1. Enhanced Feature Clustering with ACO

The ACO-based features clustering reports the application of the ant algorithm concept that aims to achieve the optimal distribution of N features into one of the K clusters, thus grouping highly correlated features into the same group. The objective is to minimize the sum of squared Euclidean distances between each feature and the cluster center by achieving the optimal distribution of N features into one of the K clusters. The ants' concept starts with strings that represent an empty solution S and the length N (number of dataset features), where each string element corresponds to a test feature. The pheromone matrix's components are all initialized to the exact quantities before the first iteration. On the basis of the quality of the generated solutions, the pheromone matrix's components are adjusted as iterations progress. For each feature of string S, the ant selects a cluster number in one of the following ways to construct solution S:

The cluster with the highest pheromone concentration is selected using probability q_0, where $q_0$is a predefined constant number in the range [0 1] and/or one of the K clusters using a probability of $(1 - q_0)$ and distribution of stochastic, indicated as p_{ij}.

The latter is referred to as biased exploration, while the first process is referred to as exploitation. The clusters with the highest pheromone concentration will be chosen in the first procedure if the random numbers q is less than q_0 (i.e., q is equal to the solution string's length and is randomly selected from a uniform distribution in the range [0 1] corresponding to these features, which are less than q_0). If the random numbers corresponding to the features q exceed q_0, then the second procedure will distribute the features to one of the clusters with a normalized pheromone probability to 1 given as in Equation (1) .

$$p_{ij} = \frac{\tau_{ij}}{\sum_{k=1}^{k} \tau_{ik}}, j = 1, \dots, k \quad (1)$$

where p_{ij} denotes the normalized pheromone probability for feature i within cluster j. The value of the objective function for a given feature clustering is used to measure the quality of the solution constructed. This objective function is calculated using the sum of squared Euclidean distances between each feature and the center of the corresponding cluster. In $\Re^n$ dimensional space, assume N features of a particular dataset {x_1,x_2,...,x_n} that should be divided into K clusters or groups. The definition of Equation (2) is the mathematical formulation of feature clustering

$$Min\ F(w,m) = \sum_{j=1}^{k}\sum_{i=1}^{N}\sum_{v=1}^{n} w_{ij} \left\|x_{iv} - m_{jv}\right\|^2 \quad (2)$$

in such a manner that

$$\sum_{j=1}^{k} w_{ij} = 1, i = 1, \dots, N \quad (3)$$

$$\sum_{i=1}^{N} w_{ij} \geq 1, j = 1, \dots, K \quad (4)$$

where $Min\ F(w,m)$ indicates the minimum value among the w as the size $N \times K$ of a weight matrix, and m as the center of cluster matrix of size $K \times n$, x_{iv} denotes the value of vth attribute of *ith* feature. Within cluster j, m_{jv} denotes the mean of all feature values for the *vth* attribute values, and w_{ij} denotes the correlated weight of feature x_i with cluster j, which may be calculated as follows:

$$w_{ij} = \begin{cases} 1\ if\ feature\ i\ is\ within\ in\ cluster\ j \\ 0\ otherwise \end{cases}$$

$$i = 1, \dots, N, j = 1, \dots, K$$

After w_{ij} is obtained, each cluster center m_j can be obtained by Equation (5)

$$m_{jv} = \frac{\sum_{i=1}^{N} w_{ij}\, x_{iv}}{\sum_{i=1}^{N} w_{iv}}, j = 1, \dots, K, v = 1, \dots, n \quad (5)$$

Thus, Equation (2) can be used to calculate the fitness value (objective function) of a given solution string S, with the center of the cluster matrix m and weight matrix w being known. Fig. 2 shows S1 constructed for N = 10 for and K = 3 as a representative of a solution string.

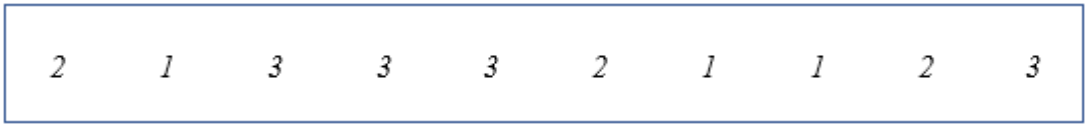

Fig. 2. Solution representation

String cluster number 2 is assigned to the first feature, cluster number 1 is assigned to the second feature, and so on. The heuristic information of the features is not available. Thus, the local search method will enhance some of the obtained solutions. For this purpose, a local search procedure is utilized on a small percentage of L solutions (the best 20% of all solutions). To conduct local search, ascending order is used to sort the fitness values of the population's selected subset. Then, a fundamental local search procedure is employed on the highest L solutions in terms of the objective function (i.e., the minimum in terms of fitness values). Each feature in the solution string has its cluster number adjusted during the local search operation with a particular threshold probability between (0,1). Thus, a random number is generated between (0,1) for each feature. Then, if the random number corresponding to the feature is less than the threshold probability, then the feature is assigned a different cluster number with a similar probability to the rest of the clusters by generating a random number. This process is illustrated in Fig. 2. For the threshold p = 0.01 and using the topmost solution string, the generated random value that corresponds to the sixth feature should be less than the threshold that indicates the number of cluster 2. Consequently, it must be randomly assigned to either cluster number 1 or 3 with an equal

probability. After the local search is completed, Equation (2) is used to determine the fitness values for the newly created solutions, where only certain solutions that have improved fitness will be accepted and replaced. Fig. 3 displays the procedure of the local search method for ACO-based feature clustering.

With the probability threshold of the local search p_{ls} in [0 1], a neighbor of S_k,
$k = 1, \ldots, L$ is produced as:

```
k =1
Generate temporary solution S_t, S_t(i) = S_k(i), i = 1,...,N.
For each feature i of S_t
 if r ≤ p_ls // r a random number in (0, 1)
        Randomly select j in the range (1, number of clusters), S_k(i) ≠ j, S_t(i) = j
        Calculate cluster centres of S_t(i), Equation(5)
        Calculate Fitness of S_t(i), Eq. (13) as F_t
        If F_t < F_k
            S_k(i) = S_t(i) and F_k < F_t
            k = k + 1; If k ≤ L go to step (2), else stop
            If k ≤ L go to step (2), else stop
End For
```

Fig. 3. Pseudocode of the local search for ACO-based feature clustering method

After the local search process is finished, the matrix of the pheromone is updated. This pheromone update rule reflects the value of the dynamic information produced by the ant's search process. Therefore, the pheromone updating rule is an adaptive memory, storing information obtained by previously discovered superior solutions and updating it at the end of each iteration. Thus, at iteration level *t*, the best L solutions among all the solutions obtained by the ants employ the pheromone updating rule based on the specified criteria (Equation [2]). These L ants simulate the pheromone trail deposition of real ants by assigning some real numbers τ_{ij} that are connected with solution attributes. The pheromone trail, $\tau_{ij}(t+1)$, is updated utilizing the rule in Equation (6)

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \sum_{l=1}^{L} \Delta\tau_{ij}^{l} \tag{6}$$

$$i = 1, \ldots, N, j = 1, \ldots, K$$

where the ϱ value is between [0 1], $(1-\varrho)$ reflects the parameter of pheromone decay to prevent the buildup of a specific parameter value permanently. $\Delta\boldsymbol{\tau}_{\boldsymbol{ij}}^{\boldsymbol{l}}$ is the additional pheromone boost and is equal to 1/*F*_l, if cluster j is allocated to the ith feature of the solution produced by ant 1 and zero otherwise. An optimal solution is one that minimizes the value of the objective function. In each iteration, if the solution produced as the current iteration's best solution has a lower objective function value than the best solution in memory, then the best solution's value is updated in memory. Consequently, the algorithm executes three steps at every iteration stage. (1) The information of the updated pheromone trail from the previous iteration is utilized to construct new solutions by ants, (2) enhancing the newly constructed solutions are enhanced via a local search, and (3) the pheromone updating rule is applied. These three steps are performed repeatedly up to a specified number of iterations. Then, the solution with the highest fitness value (lowest objective function value) takes the place of the optimal separation of features in a particular dataset into various clusters. The pseudocode depicted in Fig. 4 is the ACO-based feature clustering method.

In this pseudocode, the number of clusters, , $K_{clusters}$,, is fixed and assigned to 3, and the size of the population, $Population_{size}$, is 50

```
Input: Algorithm_parametrs , K_clusters, Data_features
Output: SbestClustering
Initialize Population, Pheromone matrix
While ≠StopCondition ( ) do
        For i = 1 to Population_size
                Construct solution using pheromone trail
                Compute weights of all test features, and cluster centres
                Compute clustering metric and assign it as objective function value of solution
        End For
        Select best solution out of all solutions using objective function values
        Compute the local search on the selected best solutions //(Fig .3)
        Use the best solution to update the pheromone trail matrix.
End While
```

Fig. 4. Pseudocode of ACO-based feature clustering method

2.2. Feature Selection Process

To establish the FS process, the pheromone initialization requires a vector to implement the amount of quality allocated to each feature represented as τ_n = {τ_1,τ_2,τ_3,...,τ_n}. The relevance of features to classes is utilized to calculate the pheromone initialization at time t=0. More relevant features are given more priority. Therefore, the classes and the mutual information, *MI*, between any feature, f_d, , are calculated for this purpose as in Equation (7) [19].

$$MI\ (d) = \sum_m \sum_{k=1}^{c} p(F_d\,[m].\,k)\ log\ \frac{p(F_d[m].p(k)}{p(F_d\,[m].k)} \tag{7}$$

where $F_d[m]$ is a value of discretizing feature *dth* for bin *m*=[1,2,....,M], M (bins number) in the MGCACO algorithm is set to 10. K is the class number, and *p(k)* is the probability of the *kth* class.

Starting with a set of clusters produced based on the ECACO, each ant in the ACO algorithm starts constructing the feature subset. As a result, in each iteration of a completely connected undirected graph, the *jth* ant picks at random at least one attribute from the cluster in its route. Then, the ant chooses the next feature from a unique cluster by employing probabilistic decision rules that depend on a specified parameter's value ε in the range [0 1], using greedy and probability processes as basis in making a decision. Otherwise, the ant stays in the present cluster. If q_0 is smaller than q, then, using the greedy strategy as in Equation (8), the jth ant chooses the next feature [19].

$$F_{next} = arg\ max_{F_d \in UF_m^j} \{\tau^i(F_d))^{\alpha}.\pi(F_d, VF_j)^{\beta}\}\ q > q_0 \tag{8}$$

where q is a predefined parameter in the range [0 1], q_0, is a random number between [0 1], UF_m^j_is the collection of features in the current cluster (mth cluster) that have not yet been accessed by the *jth* ant, $\tau^i$is the value of the pheromone quantity associated with the feature (F_d), VF_j denotes the previously chosen features (visited features), and $\pi(F_d, VF_j)$ denotes the heuristic information function, which is defined as in Equation (9) [19].

$$\pi(F_d, VF_j) = F - Score(F_d) - \frac{1}{N_{VF_j}} \sum_{m=1}^{N_{VF_j}} w(F_d, F_m) \tag{9}$$

where N_{VF_j} is the size of VF, and $w(F_d, F_m)$ denotes the Pearson correlation values [23] between feature F_d and F_m that were obtained from the visited features (VF_j) through all previous clusters by the *jth* ant computed as in Equation (10).

$$w(F_d, F_m) = \left| \frac{(F_d - \bar{F}_m) - (F_d - \bar{F}_m)}{\sqrt{(F_d - \bar{F}_d)^2 (F_m - \bar{F}_m)^2}} \right| \quad (10)$$

where $\bar{F}_d$ and $\bar{F}_m$ denote the vectors of features, and F_d and F_m are completely uncorrelated if $w(F_d, F_m)$ has a value equal to 0 or correlated if $w(F_d, F_m)$ has a value equal to 1. The similarity values among features are identical to each other in most cases. Thus, the influence of outliers is reduced without excluding them from the distribution of weights by using a nonlinear normalization method called softmax scaling to scale the weight $w(F_d, F_m)$ of the edge to the range [0 1] as in Equation (11) [19]

$$\hat{w}_{dm} = \frac{1}{1 + exp\left(\frac{w_{dm} - \bar{w}}{\sigma}\right)} \quad (11)$$

where $\hat{w}_{dm}$ denotes the normalized value of w_{dm} between F_d and F_m features, and $\bar{w}$ and σ are the mean and variance, respectively, of all w_{dm} values.

In addition, $F - Score(F_d)$ in Equation (9) indicates the feature score F_d based on the class relevance [19], calculated as in Equation (12).

$$\mathrm{F - Score}(F_d) = \frac{\sum_{k=1}^{K} n_k \cdot (\mu_k^d - \mu^d)^2}{\sum_{k=1}^{K} n_k \cdot (\sigma_k^d)^2} \quad (12)$$

where K indicates the classes number of the dataset; μ_k^d and σ_k^d are the variance and the mean of the kth class with σ_k^d samples (n_k is the samples number in class k), respectively; and μ^d denotes the mean of the patterns in the dth feature vector. The F-score values are normalized by the softmax scaling in ranges 0 and 1 as in Equation (11). Features with a larger F-score have a greater discrimination power.

As a result, in the greedy method, the ants select features that have the maximum dependency and the minimum parity to the features that were previously selected for the target class. If q_0 is greater than q, then a probabilistic method for every feature in the current cluster that has not yet been visited $(F_d \in UF_m^j)$ is defined as in Equation (13) [19]

$$P(F_d) = \frac{[\tau^i(F_d)]^{\alpha} [\pi(F_d, VF_j)]^{\beta}}{\sum_{F_d \in U_m^j} [\tau^i(F_d)]^{\alpha} [\pi(F_d, VF_j)]^{\beta}} \; For\; F_d \in UF_m^j \qquad q < q_0 \quad (13)$$

where β and α denote the value of important fixed parameter between (0,1) for the heuristic information and the pheromone, respectively. $\tau f^i(F_d)$ is the quantity or quality attached with each feature F_d, while $\pi(F_d, VF_j)$denotes the heuristic information obtained by Equation (9). As a result, the following feature would be chosen according to the roulette wheel rule. Fig. 5 shows the search for the best feature subset from the clusters using ACO.

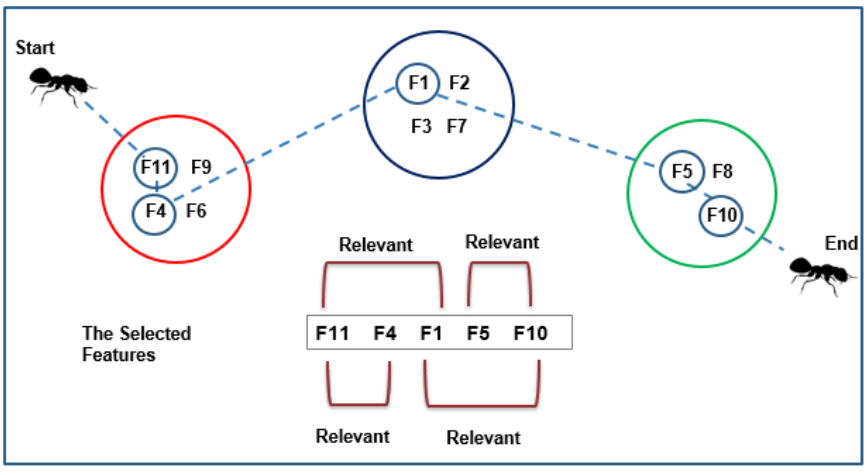

Fig. 5. Ant selection process

Once all the ants have finished traveling routes in the graph, the quantity of pheromones for per selected feature $(i.e., \tau^i(f_d))$ is updated. This step is performed before each iteration i ends. Applying the updating rule as in Equation (14) allows the quantity of pheromones for per feature to be updated [19]

$$\tau^{i+1}(F_d) = (1-\rho).\tau^i(F_d) + \sum_{j=1}^{A} \Delta_j^i(F_d) \quad (14)$$

where the value of ϱ is a certain parameter of pheromone decay to prevent limitless accumulation, and $\tau^i(F_d)$ and $\tau^{i+1}(F_d)$ indicate the amounts of pheromone on feature F_i at times $(t)$and $(t+1)$, respectively. A is the ant number, the additional pheromone boost to feature F_i via ant j is j is $\Delta_j^i(F_d))$, which is calculated from the use of MDA [19] as in Equation (15)

$$\Delta_j^i(F_d) = \begin{cases} \gamma_j^i, F_d \in FS_j^i \\ 0, F_d \notin FS_j^i \end{cases} \quad (15)$$

where FS_j^i is the feature that the jth ant selected in the ith iteration, γ_j^i is indicated as MDA [19] corresponding in the ith iteration to the jth selected subset assuming that N_{fs} features are to be chosen. All the features are sorted at each iteration by using Equation (16) [19]. N_{fs} features are selected and the values of MDA identical to these features are calculated using Equation (17) [19]. The pheromone values are adjusted only if the MDA values have improved from the previous iteration; otherwise, no change is applied.

$$Sf(d) = \tau f(d).Fscore(f_d)\ d = 1,2,\dots,D \quad (16)$$

$$Fitness = MDA(Feature(1,\dots,N_{fs})) \quad (17)$$

The procedures are performed repeatedly until the iterations are finished. Lastly, the final subset is identified by sorting the features in descending order according to their pheromone values. The feature number that will be selected is calculated by multiplying a fixed feature number, ω, with the cluster number, K. Fig. 6 provides a low-level description of the FS process.

```
Initialize Pheromone
Initialize the parameters value
While stopping criterion not satisfied do
    Position each ant in a random feature
        For each ant do
            Calculate the probability of each feature
            Select feature by applying the state transition rule
            Feature subset construction by using the ε value
            Collect the fitness by evaluating the quality of the feature subset
            Update pheromone for the selected feature
        Until every ant has built a solution
End while
```

Fig. 6. Pseudocode of ACO-based feature selection

2.3. Datasets

Experiments are conducted on different benchmark classification problems (i.e., UCI and microarray datasets) to demonstrate the efficacy of the proposed method. Table 1 summarizes the characteristics of the datasets. The UCI machine learning repository has comprehensive descriptions available in [24], while the microarray datasets are accessible in [25]. Missing values for each attribute are replaced with the mean of the available data. Softmax scaling is a nonlinear normalization method [19] that was utilized to scale the datasets into the range [0 1]. Each dataset was partitioned into two sets, with one-third for testing and two-thirds for training. The testing set was used to identify the accuracy of the chosen features, while the training set was used to specify the feature subset.

Table 1. Characteristics of the datasets

Dataset Name	Dataset Type	Features Values		No. of Patterns	No. of Features	No. of Classes
		Categorical	*Numerical*			
Wine	Physical	-	√	178	13	3
Hepatitis	Biological	√	√	155	19	2
Ionosphere	Physical	-	√	351	34	2
Spambase	Computer	-	√	4601	57	2
Arrhythmia	Biological	√	√	452	279	16
Madelon	Artificial	-	√	4400	500	2
Colon	Biological	-	√	62	2000	2
Leukemia	Biological	-	√	72	7129	2

The listed datasets vary in instance numbers (from 60–4601), attribute numbers (from 13–7129), and class labels (from 2–16). The datasets are chosen from five application domains to effectively solve several challenging issues in different application domains. The biggest size (50%) belongs to the biological domain, where (25%) of these datasets are related to the microarray application domains.

The size of the dataset can be divided into four categories based on the number of attributes: small, medium, high, and very high. The least number of datasets (13%) are classified as having a high number of attributes (between 50 and 100). 25% of the datasets are categorized as small, with the number of attributes ranging from 0 to 20, and 13% of the datasets have a medium number of attributes (in the range of 20–50). The remaining datasets with the highest percentage (50%) are datasets that have 100 and more attributes

2.4. Classifier and Evaluation Criteria

Several well-known benchmark classifiers are utilized in the evaluation process to demonstrate the generality of the proposed method. The classifiers used in evaluating the proposed algorithms are random forest, k-NN, decision tree, and support vector machine.

Accuracy is an effective way of validating FS methods by using each classifier's performance. Therefore, it is utilized to evaluate the generated feature subsets and is computed using Equation (18).

$$AC = \frac{TP+TN}{TP+TN+FP+FN} \tag{18}$$

where TN, TP, FN, and FP are the confusion matrices describing the classification results (true or false). TN, where the model predicted a negative value and the actual value is negative, FN for the model predicted a negative and the result is false, TP indicates the model predicted a positive value and the actual value is positive, and FP for the model predicted a positive, and result is false. Table 2 shows the possible cases based on predictive values and actual values.

Table 2. Confusion Matrix

Predicted Values	Actual Values		
		Positive	Negative
	Positive	TP	FP
	Negative	FN	TN

3. Results and Discussion

The implementation of the proposed ACO-based features clustering method in the MGCACO algorithm is called ECACO. The initialization parameters that are used to evaluate the ACO-based features clustering and ACO-based FS parameters are listed in Table 3.

Table 3. ACO-based features clustering experimental parameters

Model	Parameter	Description	Value
ACO-based clustering	A	Ant number	50
	I	Iterations number	1000
	q_0	Exploration/exploitation	0.98
	ρ	Evaporation rate	0.01
	P_{ls}	Local search probability	0.01
ACO-based FS	A	Ant number	100
	I	Iterations number	50
	α	Importance of pheromone	1
	β	Importance of heuristic	1
	q_0	Exploration/exploitation	0.7
	ρ	Evaporation rate	0.9
	ε	Threshold to remain in current cluster	0.4
	Run	Number of runs	10

Well-known ACO-based FS algorithms were compared with the proposed algorithm, namely, GCACO, MGCACO, microarray gene selection based on ACO (MGSACO), unsupervised FS-based ACO (UFSACO), and FS method for modified binary ant system and clustering combination (FSCBAS).

3.1. Experiment on the UCI Datasets

Tables 4–7 illustrate the experimental results, which include the standard deviation (Std) average and classification accuracy (Acc) of five FS algorithms and ECACO when four classifiers were employed, namely, random forest, support vector machine, decision tree, and k-NN. The highest result is highlighted, and the figures in parentheses show the rank of the algorithms.

Table 4 illustrates that for the case of the support vector machine classifier, except for the lonosphere dataset, which ranks third, the proposed ECACO achieves the highest classification accuracy across all datasets. This result may be due to a threshold set on the estimated class probability value. In Table 5, the proposed ECACO obtains the best classification accuracy results in four out of six datasets when the k-NN classifier is employed. The k-NN classifier is known to work well with numerical data. This is the case for the Hepatitis and Madelon datasets, where the proposed ECACO ranks second. Despite this, ECACO can also select significant features in the Hepatitis dataset, which contains both types of data (numerical and categorical).

Table 4. Average classification accuracy using support vector machine classifier on UCI datasets

Dataset	# Selected features		ECACO	FSCBAS	MGCACO	GCACO	MGSACO	UFACO
Wine	6	Acc.	**98.03 (1)**	97.2 (3)	98 (2)	95.1 (5)	94.61 (6)	96.02 (4)
		Std.	**±** 1.22	± 0	± 1.72	± 2.6	± 2.52	± 2.04
Hepatitis	6	Acc.	**83.86 (1)**	81.55 (4)	83.85 (2)	83.64 (3)	81.1 (5)	80.78 (6)
		Std.	± 1.59	± 3.74	± 4.82	± 2.83	± 1.13	± 0.4
lonosphere	15	Acc.	85.79 (3)	85.62 (4)	86.24 (2)	**86.79 (1)**	81.6 (6)	84.96 (5)
		Std.	± 2.32	± 3.57	± 2.83	± 2.41	± 2.09	± 2.12
SpamBase	24	Acc.	**91.78 (1)**	85.78 (3)	90.98 (2)	84.51 (4)	81.86 (6)	83 (5)
		Std.	± 0.80	± 0.22	± 0.90	± 2.12	± 2.67	± 2.67
Arrhythmia	20	Acc.	**86.03 (1)**	68 (3)	70. 84 (2)	62 (4)	56.55 (5)	54.39 (6)
		Std.	± 1.31	± 3.63	± 3.83	± 5.2	± 1.4	± 0.29
Madelon	40	Acc.	**66.51 (1)**	61.2 (3)	58.98 (6)	64.61 (2)	60.98 (4)	60.75 (5)
		Std.	± 0.27	± 2.07	± 2.83	± 5.58	± 0.27	± 0.24

Table 5. Average classification accuracy using k-NN classifier on UCI datasets

Dataset	# Selected features		ECACO	FSCBAS	MGCACO	GCACO	MGSACO	UFACO
Wine	6	Acc. Std.	**98.36 (1)** ± 1.63	97.2 (3) ± 0	98 (2) ± 2.02	95.68 (5) ± 2.78	94.61 (6) ± 2.52	96.02 (4) ± 2.04
Hepatitis	6	Acc. Std.	81.73 (2) ± 1.01	79.36 (6) ± 3.74	**82.12 (1)** ± 6.97	81.36 (3) ± 1.33	81.1 (4) ± 1.13	80.78 (5) ± 0.41
Lonosphere	15	Acc. Std.	**86.89 (1)** ± 2.31	85.62 (3) ± 3.75	86.5 (2) ± 2.24	85.36 (4) ± 4.14	81.6 (6) ± 2.1	84.96 (5) ± 2.12
SpamBase	24	Acc. Std.	**90.29 (1)** ± 0.76	85.78 (5) ± 0.22	89.6 (2) ± 1.26	87.04 (4) ± 1.32	81.86 (6) ± 2.68	88.33 (3) ± 0.68
Arrhythmia	20	Acc. Std.	**86.03 (1)** ± 0.43	57.31 (3) ± 1.33	64.07 (2) ± 3.14	53.75 (5) ± 5.38	55.25 (4) ± 1.84	50.87 (6) ± 1.2
Madelon	40	Acc. Std.	75.45 (3) ± 1.04	59.8 (6) ± 1.34	60.94 (5) ± 2.56	**76.21 (1)** ± 1.74	75.85 (2) ± 0.87	75.13 (4) ± 0.4

Table 6 shows the decision tree classifier's classification accuracy for the algorithms. The best classification accuracy was obtained in all six datasets using the proposed ECACO algorithm. The proposed algorithm could select important features to provide high classification accuracy, including all types of feature values (numerical and categorical).

Table 6. Average classification accuracy using decision tree classifier on UCI datasets

Dataset	# Selected features		ECACO	FSCBAS	MGCACO	GCACO	MGSACO	UFACO
Wine	6	Acc. Std.	**96.22 (1)** ± 1.84	93.83 (2) ± 0	93.17 (4) ± 3.37	93.57 (3) ± 2.98	92.93 (6) ± 3.47	93.09 (5) ± 3.02
Hepatitis	6	Acc. Std.	**80.79 (1)** ± 0.74	79.36 (4) ± 0	78.08 (5) ± 6.71	80.31 (2) ± 1.14	80.26 (3) ± 0.92	77.04 (6) ± 1.22
Lonosphere	15	Acc. Std.	**91.42 (1)** ± 1.62	88.32 (5) ± 1.41	90.86 (2) ± 4.21	90.54 (3) ± 1.98	86.7 (6) ± 2.14	89.06 (4) ± 1.51
SpamBase	24	Acc. Std.	**91.64 (1)** ± 0.65	89.68 (4) ± 0.22	90.88 (2) ± 0.66	89.12 (5) ± 1.12	89.83 (3) ± 1.52	89.08 (6) ± 0.51
Arrhythmia	20	Acc. Std.	**86.10 (1)** ± 0.97	59.76 (3) ± 1.37	66.71 (2) ± 3.22	56.62 (4) ± 4.13	49.94 (6) ± 2.05	50.89 (5) ± 2.32
Madelon	40	Acc. Std.	**84.17 (1)** ± 1.11	67.4 (6) ± 1.25	71.42 (5) ± 3.50	81.77 (2) ± 2.38	80.12 (3) ± 0.57	79.49 (4) ± 0.67

The results in Table 7 show that for the random forest classifier, the proposed ECACO obtains the best results in four out of six datasets, where the algorithm ranks second rank with very small differences from the best algorithms in the SpamBase and Madelon datasets. The random forest classifier is known to perform well with categorical datasets. Nonetheless, the proposed algorithm can choose meaningful features for classification even for datasets with categorical and numerical data types.

Table 7. Average classification accuracy using random forest classifier on UCI datasets

Dataset	# Selected features		ECACO	FSCBAS	MGCACO	GCACO	MGSACO	UFACO
Wine	6	Acc.	**98.19 (1)**	97.2 (3)	97.5 (2)	96.11 (4)	94.61 (6)	96.02 (5)
		Std.	± 1.71	± 0	± 2.53	± 2.32	± 2.53	± 2.04
Hepatitis	6	Acc.	**85.29 (1)**	81.39 (3)	84.43 (2)	81.3 (4)	81.1 (5)	80.78 (6)
		Std.	± 0.16	± 0	± 4.30	± 1.76	± 1.14	± 0.41
Lonosphere	15	Acc.	**94.62 (1)**	85.77 (3)	93.59 (2)	85.1 (4)	81.6 (6)	84.95 (5)
		Std.	± 1.43	± 3.57	± 1.77	± 2.41	± 2.1	± 2.12
SpamBase	24	Acc.	94.16 (2)	85.78 (5)	**94.17 (1)**	89.02 (4)	81.86 (6)	89.96 (3)
		Std.	± 0.62	± 0.23	± 0.75	± 1.98	± 2.68	± 0.54
Arrhythmia	20	Acc.	**84.21 (1)**	66.38 (3)	75.36 (2)	61.18 (4)	57.22 (5)	50.05 (6)
		Std.	± 0.42	± 1.33	± 2.24	± 2.22	± 1.6	± 2.91
Madelon	40	Acc.	83.26 (2)	61.19 (6)	76.51 (4)	72.27 (5)	**83.29 (1)**	83.24 (3)
		Std.	± 0.85	± 1.63	± 2.75	± 1.72	± 0.48	± 0.49

Table 8 depicts the summary of the averages for the classification accuracy (Acc) and standard deviation (Std) algorithms concerning the classifiers. The best result for each algorithm is highlighted. Overall, ECACO performs better than other algorithms in all classifiers regarding classification accuracy. FSCBAS has the best standard deviation for the decision tree classifier because of the parallel process of the FSCBAS algorithm, which may decrease the randomization of the solution. However, the proposed ECACO shows small competitive values compared to the other algorithms, indicating its stable performance. The accuracy results listed in Table 8 are displayed in Fig. 7.

Table 8. Results summary on UCI datasets

Classifiers		ECACO	FSCBAS	MGCACO	GCACO	MGSACO	UFACO
SVM	Acc.	**85.33**	79.89	69.67	79.44	76.11	76.65
	Std.	1.22	2.20	2.82	3.45	1.68	1.29
k-NN	Acc.	**86.46**	77.51	80.20	79.90	78.37	79.34
	Std.	1.19	1.64	1.73	3.03	1.85	1.14
DT	Acc.	**88.39**	79.72	81.85	81.98	79.96	79.77
	Std.	1.15	0.70	3.61	2.28	1.77	1.54
RF	Acc.	**89.95**	79.61	86.92	80.83	79.94	80.83
	Std.	0.86	1.15	2.39	2.06	1.75	1.14

In summary, the ECACO has the best classification accuracy. This good performance is due to the effectiveness of the enhancement on the clustering method, which is based on the ACO algorithm concept, wherein the proposed ACO intensification and diversification are changed iteratively through pheromone update, enabling the successful implementation of both global and local search. Thus, the method has managed to escape from the local optima in grouping the highly correlated features during the exploration process, thus improving the classification accuracy of the highlighted datasets shown in Tables 4–7.

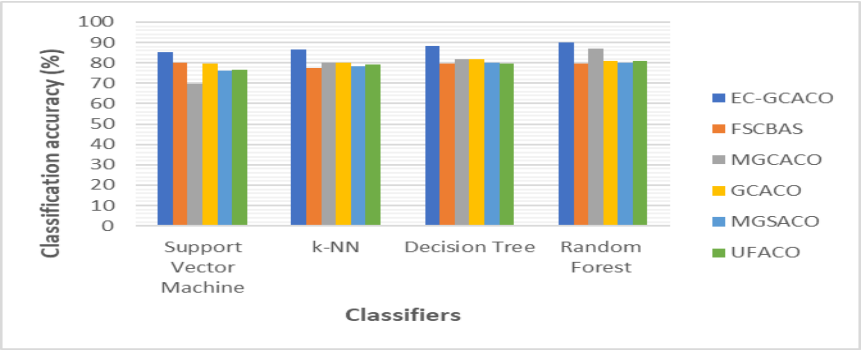

Fig. 7. Average classification accuracy for UCI datasets

3.2. Experiment on the Microarray Dataset

Tables 9 and 12 show the experimental results, which include the standard deviation (Std) and average classification accuracy (Acc) of five FS algorithms and the proposed ECACO. The classifiers used for FS were random forest, support vector machine, k-NN, and decision tree classifiers. The Colon and Leukemia datasets are classified as microarray datasets. For each dataset, the highest result is highlighted, and the figures in parentheses show the rank of the algorithms. Tables 9, 10, 11, and 12 show that the proposed ECACO achieves the best average classification accuracy in all cases, except for the case of the k-NN classifier, where the FSCBAS has the best result on the Colon dataset.

Table 9. Average classification accuracy using support vector machine classifier on microarray datasets

Dataset	# Selected features		ECACO	FSCBAS	MGCACO	GCACO	MGSACO	UFACO
Colon	40	Acc.	**90.00 (1)**	87.59 (2)	84.76 (3)	81.99 (6)	83.88 (4)	82.1 (5)
		Std.	± 1.51	± 1.31	± 6.31	± 2.23	± 2.94	± 3.26
Leukemia	40	Acc.	**98.33 (1)**	95.42 (3)	95.83 (2)	93.67 (4)	90.14 (5)	89.45 (6)
		Std.	± 0.90	± 0.67	± 3.22	± 2.35	± 5.17	± 3.28

Table 10. Average classification accuracy using k-NN classifier on microarray datasets

Dataset	# Selected features		ECACO	FSCBAS	MGCACO	GCACO	MGSACO	UFACO
Colon	40	Acc.	87.14 (2)	**87.59 (1)**	81.43 (4)	81.13 (5)	82.1 (3)	80.65 (6)
		Std.	± 2.04	**± 2.52**	± 5.01	± 4.32	± 2.79	± 2.28
Leukemia	40	Acc.	**98.75 (1)**	95.42 (2)	93.75 (3)	88.02 (5)	88.06 (4)	87.92 (6)
		Std.	± 1.66	± 1.64	± 9.63	± 3.11	± 2.27	± 1.45

Table 11. Average classification accuracy using decision tree classifier on microarray datasets

Dataset	# Selected features		ECACO	FSCBAS	MGCACO	GCACO	MGSACO	UFACO
Colon	40	Acc.	**87.61 (1)**	86.13 (2)	78.09 (6)	84.23 (4)	85.17 (3)	82.91 (5)
		Std.	± 2.80	± 0.83	± 7.73	± 2.39	± 2.82	± 4.18
Leukemia	40	Acc.	**93.75 (1)**	93.62 (2)	90.83 (3)	85.74 (5)	90.28 (4)	77.92 (6)
		Std.	± 0.08	± 0	± 3.63	± 1.52	± 4.29	± 2.96

Table 12. Average classification accuracy using random forest classifier on microarray datasets

Dataset	# Selected features		ECACO	FSCBAS	MGCACO	GCACO	MGSACO	UFACO
Colon	40	Acc.	**88.57 (1)**	87.59 (2)	84.29 (3)	81.22 (6)	82.91 (4)	81.78 (5)
		Std.	**± 2.33**	± 2.53	± 4.28	± 3.95	± 2.18	± 4.93
Leukemia	40	Acc.	**99.76 (1)**	95.42 (3)	98.75 (2)	89.69 (4)	89.36 (5)	86.12 (6)
		Std.	**± 0.46**	± 0.67	± 1.90	± 3.61	± 5.16	± 3.53

Table 13 summarizes the results presented in Tables 9–12, where the best result for each classifier is highlighted. Overall, the ECACO algorithm outperforms the other algorithms regarding average classification accuracy. Compared with the other algorithms, the proposed ECACO shows small competitive values in terms of standard deviation, indicating its stable performance. Fig. 8 shows the average classification accuracy for all classifiers. In general, the proposed algorithm outperforms the other algorithms in all classifiers.

Table 13. Average classification accuracy, standard deviation, and performance rank on microarray datasets

Classifiers		ECACO	FSCBAS	MGCACO	GCACO	MGSACO	UFACO
SVM	Acc.	**94.16**	91.50	90.29	87.83	87.01	85.77
	Std.	1.20	0.99	4.765	2.29	4.05	3.27
k-NN	Acc.	**92.94**	91.50	87.58	84.57	85.08	84.28
	Std.	1.85	2.08	7.23	3.71	2.53	1.86
DT	Acc.	**90.68**	89.87	84.46	84.98	85.20	78.70
	Std.	1.44	0.41	5.68	3.91	3.55	3.57
RF	Acc.	**94.16**	91.50	91.51	85.45	86.13	83.95
	Std.	1.39	1.6	3.09	6.78	3.67	4.23

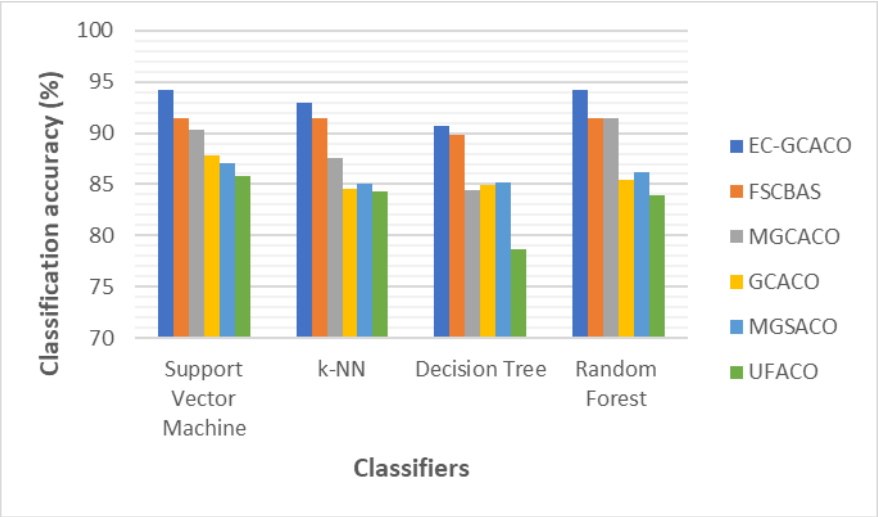

Fig. 8. Average classification accuracy of microarray data

4. Conclusion

The experimental findings demonstrate that the proposed ECACO surpasses other FS algorithms in terms of classification performance in both benchmark and microarray datasets. With the ACO-based feature clustering method integrated with the proposed ECACO, the number of selected features was reduced significantly without losing important information, resulting in high classification accuracy in handling the problem of high-dimensionality datasets. Experimental results show that the enhanced feature clustering method can effectively perform well in space, successfully escape from the local optima while grouping the highly correlated features during exploration, and improve classification accuracy. In summary, the proposed ECACO algorithm can minimize high-dimensionality datasets while maintaining acceptable classification accuracy. However, ACO-based feature clustering did not obtain the best result in some datasets (i.e., Lonosphere, Madelon, and Hepatitis), especially when the support vector machine and k-NN classifiers were used. Thus, the appropriate number of clusters should be specified to overcome this limitation. A more adaptive version should be developed, where the appropriate number of clusters is adapted automatically rather than being predefined.

Acknowledgment

The authors wish to express gratitude to the Shatt Al-Arab University College for giving encouragement and full support in providing the facilities to complete this work.

Declarations

Author contribution. All authors contributed equally to the main contributor to this paper. All authors read and approved the final paper.

Funding statement. This proposed work did not receive any funding.

Conflict of interest. The authors declare no conflict of interest.
Additional information. No additional information is available for this paper.

References

[1] A. K. Shukla, D. Tripathi, B. R. Reddy, and D. Chandramohan, "A study on metaheuristics approaches for gene selection in microarray data: algorithms, applications and open challenges," *Evol. Intell.*, vol. 13, no. 3, pp. 309–329, Sep. 2020, doi: 10.1007/S12065-019-00306-6.

[2] M. Yuan, Z. Yang, and G. Ji, "Partial maximum correlation information: A new feature selection method for microarray data classification," *Neurocomputing*, vol. 323, pp. 231–243, Jan. 2019, doi: 10.1016/J.NEUCOM.2018.09.084.

[3] P. Agrawal, H. F. Abutarboush, T. Ganesh, and A. W. Mohamed, "Metaheuristic algorithms on feature selection: A survey of one decade of research (2009-2019)," *IEEE Access*, vol. 9, pp. 26766–26791, 2021, doi: 10.1109/ACCESS.2021.3056407.

[4] Z. Manbari, F. Akhlaghian Tab, and C. Salavati, "Fast unsupervised feature selection based on the improved binary ant system and mutation strategy," *Neural Comput. Appl.*, vol. 31, no. 9, pp. 4963–4982, Sep. 2019, doi: 10.1007/S00521-018-03991-Z.

[5] B. Sahu, S. Dehuri, and A. Jagadev, "A Study on the Relevance of Feature Selection Methods in Microarray Data," *Open Bioinforma. J.*, vol. 11, no. 1, pp. 117–139, Aug. 2018, doi: 10.2174/1875036201811010117.

[6] P. Bugata and P. Drotar, "On some aspects of minimum redundancy maximum relevance feature selection," *Sci. China Inf. Sci.*, vol. 63, no. 1, pp. 1–15, Jan. 2020, doi: 10.1007/S11432-019-2633-y.

[7] M. Jiang, Y. Fang, Y. Su, G. Cai, and G. Han, "Random Subspace Ensemble with Enhanced Feature for Hyperspectral Image Classification," *IEEE Geosci. Remote Sens. Lett.*, vol. 17, no. 8, pp. 1373–1377, Aug. 2020, doi: 10.1109/LGRS.2019.2948960.

[8] W. Wang, Z. Yu, C. Fu, D. Cai, and X. He, "COP: customized correlation-based Filter level pruning method for deep CNN compression," *Neurocomputing*, vol. 464, pp. 533–545, Nov. 2021, doi: 10.1016/J.NEUCOM.2021.08.098.

[9] L. Wang, S. Jiang, and S. Jiang, "A feature selection method via analysis of relevance, redundancy, and interaction," *Expert Syst. Appl.*, vol. 183, p. 115365, Nov. 2021, doi: 10.1016/J.ESWA.2021.115365.

[10] Z. Manbari, F. AkhlaghianTab, and C. Salavati, "Hybrid fast unsupervised feature selection for high-dimensional data," *Expert Syst. Appl.*, vol. 124, pp. 97–118, Jun. 2019, doi: 10.1016/J.ESWA.2019.01.016.

[11] H. Almazini and K. R. Ku-Mahamud, "Adaptive Technique for Feature Selection in Modified Graph Clustering-Based Ant Colony Optimization," *Int. J. Intell. Eng. Syst.*, vol. 14, no. 3, pp. 332–345, 2021, doi: 10.22266/ijies2021.0630.28.

[12] H. F. Almazini, K. R. Ku-Mahamud, and H. Almazini, "Heuristic Initialization Using Grey Wolf Optimizer Algorithm for Feature Selection in Intrusion Detection," *Int. J. Intell. Eng. Syst.*, vol. 16, no. 1, pp. 410–418, 2023, doi: 10.22266/ijies2023.0228.36.

[13] H. Li *et al.*, "A Hybrid Feature Selection Algorithm Based on a Discrete Artificial Bee Colony for Parkinson's Diagnosis," *ACM Trans. Internet Technol.*, vol. 21, no. 3, Jun. 2021, doi: 10.1145/3397161.

[14] M. Mafarja and S. Mirjalili, "Whale optimization approaches for wrapper feature selection," *Appl. Soft Comput.*, vol. 62, pp. 441–453, Jan. 2018, doi: 10.1016/J.ASOC.2017.11.006.

[15] M. Amoozegar and B. Minaei-Bidgoli, "Optimizing multi-objective PSO based feature selection method using a feature elitism mechanism," *Expert Syst. Appl.*, vol. 113, pp. 499–514, Dec. 2018, doi: 10.1016/J.ESWA.2018.07.013.

[16] H. F. Almazini, S. Mortada, H. F. A. Al-Mazini, H. N. K. Al-Behadili, and J. Alkenani, "Improved discrete plant propagation algorithm for solving the traveling salesman problem," *IAES Int. J. Artif. Intell.*, vol. 11, no. 1, pp. 13–22, Mar. 2022, doi: 10.11591/IJAI.V11.I1.PP13-22.

[17] T. Mostafaie, F. Modarres Khiyabani, and N. J. Navimipour, "A systematic study on meta-heuristic approaches for solving the graph coloring problem," *Comput. Oper. Res.*, vol. 120, p. 104850, Aug. 2020,

doi: 10.1016/J.COR.2019.104850.

[18] A. K. Shukla, D. Tripathi, B. R. Reddy, and D. Chandramohan, "A study on metaheuristics approaches for gene selection in microarray data: algorithms, applications and open challenges," *Evol. Intell.*, vol. 13, no. 3, pp. 309–329, Sep. 2020, doi: 10.1007/S12065-019-00306-6

[19] H. Ghimatgar, K. Kazemi, M. S. Helfroush, and A. Aarabi, "An improved feature selection algorithm based on graph clustering and ant colony optimization," *Knowledge-Based Syst.*, vol. 159, pp. 270–285, Nov. 2018, doi: 10.1016/J.KNOSYS.2018.06.025.

[20] M. Mohammadi, M. Fazlali, and M. Hosseinzadeh, "Accelerating Louvain community detection algorithm on graphic processing unit," *J. Supercomput.*, vol. 77, no. 6, pp. 6056–6077, Jun. 2021, doi: 10.1007/S11227-020-03510-9.

[21] J. Zhang, Z. Ma, Q. Sun, and J. Yan, "Research Review on Algorithms of Community Detection in Complex Networks," *J. Phys. Conf. Ser.*, vol. 1069, no. 1, p. 012124, Aug. 2018, doi: 10.1088/1742-6596/1069/1/012124.

[22] I. BenMansour, "An Effective Hybrid Ant Colony Optimization for the Knapsack Problem Using Multi-Directional Search," *SN Comput. Sci.*, vol. 4, no. 2, pp. 1–15, Mar. 2023, doi: 10.1007/S42979-022-01564-5.

[23] A. E. Hassanien, Ed., "Machine Learning Paradigms: Theory and Application," vol. 801, 2019, doi: 10.1007/978-3-030-02357-7.

[24] "Dua, D. and Graff, C. (2019) UCI Machine Learning Repository. University of California, School of Information and Computer Science, Irvine, CA. - References - Scientific Research Publishing.". Available at: https://scirp.org/reference/referencespapers.aspx?referenceid=2607575 .

[25] R. Alanni, J. Hou, H. Azzawi, and Y. Xiang, "Deep gene selection method to select genes from microarray datasets for cancer classification," *BMC Bioinformatics*, vol. 20, no. 1, pp. 1–15, Nov. 2019, doi: 10.1186/S12859-019-3161-2/FIGURES/5.