Complex & Intelligent Systems (2022) 8:2769–2789
https://doi.org/10.1007/s40747-021-00452-4

ORIGINAL ARTICLE

Two-stage improved Grey Wolf optimization algorithm for feature selection on high-dimensional classification

Chaonan Shen[1] · Kai Zhang[2]

Received: 28 December 2020 / Accepted: 17 June 2021 / Published online: 30 July 2021

Abstract
In recent years, evolutionary algorithms have shown great advantages in the field of feature selection because of their simplicity and potential global search capability. However, most of the existing feature selection algorithms based on evolutionary computation are wrapper methods, which are computationally expensive, especially for high-dimensional biomedical data. To significantly reduce the computational cost, it is essential to study an effective evaluation method. In this paper, a two-stage improved gray wolf optimization (IGWO) algorithm for feature selection on high-dimensional data is proposed. In the first stage, a multilayer perceptron (MLP) network with group lasso regularization terms is first trained to construct an integer optimization problem using the proposed algorithm for pre-selection of features and optimization of the hidden layer structure. The dataset is compressed using the feature subset obtained in the first stage. In the second stage, a multilayer perceptron network with group lasso regularization terms is retrained using the compressed dataset, and the proposed algorithm is employed to construct the discrete optimization problem for feature selection. Meanwhile, a rapid evaluation strategy is constructed to mitigate the evaluation cost and improve the evaluation efficiency in the feature selection process. The effectiveness of the algorithm was analyzed on ten gene expression datasets. The experimental results show that the proposed algorithm not only removes almost more than 95.7% of the features in all datasets, but also has better classification accuracy on the test set. In addition, the advantages of the proposed algorithm in terms of time consumption, classification accuracy and feature subset size become more and more prominent as the dimensionality of the feature selection problem increases. This indicates that the proposed algorithm is particularly suitable for solving high-dimensional feature selection problems.

Keywords Feature selection · Improved Gray Wolf optimization (IGWO) · Multilayer perceptron (MLP) · Group lasso

Introduction

With the booming field of big data and artificial intelligence, the quest for simple and efficient models has become stronger among researchers. On one hand, there are a large amount of existing practical applications involving high-dimensional data, such as in the fields of text mining [1], genomics [2] and image retrieval [3]. However, not all data are relevant for problem solving. Sometimes, the inclusion of unrelated features may diminish the performance of the model. On the other hand, it is due to some applications, such as Internet of Things (IoT) field [4], as they have high constraints on energy consumption, model size and model latency, all must use simple and efficient learning models to meet the needs. Therefore, removing irrelevant and redundant features from the data is a highly recommended approach, which can be done by feature selection methods. The basic concept of feature selection is to reduce the number of features by eliminating unrelated or redundant features, thus simplifying the learned model, shortening the inference time of the model, and enhancing the generalization ability of the model [5].

A large amount of research work has demonstrated the effectiveness of feature selection in simplifying models and improving their performance [6]. According to the different

✉ Kai Zhang
zhangkai@wust.edu.cn

Chaonan Shen
449777215@qq.com

[1] Hubei Province Key Laboratory of Intelligent Information Processing and Real-Time Industrial System, Wuhan 430065, China

[2] The School of Computer Science and Technology, Wuhan University of Science and Technology, Wuhan 430065, Hubei, China

solution methods, feature selection can be grouped into three categories: wrapper methods [7], filter methods [8] and embedded methods [9]. Wrapper methods combine the feature selection process with the classifier training process by selecting different feature subsets to obtain different reduced data and training the classifier. The cross-validation performance of the classifier is used as the evaluation metrics of the feature subsets. In contrast, the filter method judges the importance of features based only on the relationship between the unidimensional features of the training data and the target variables. The embedding method usually adds regularization constraints to the loss function to train the classifier model, and judges the importance of features and performs feature selection by the characteristics of the classifier itself.

Although feature selection has been widely studied, yet feature selection is still a challenging task. For wrapper methods, classifier performance is used as the evaluation metric for feature subsets, and its evaluation method is straightforward and accurate, so it usually produces more accurate subsets than filtering methods. However, evaluating each feature subset requires retraining a classifier, so it is computationally expensive, especially for high-dimensional data. Therefore, this method is deficient in terms of scalability. Also, there is a risk of overfitting the training set because classifier performance metrics are used to evaluate the subsets [10]. For the filter methods, the correlation between the single-dimensional features of the data and the target variable is used to determine the importance of the features, which often has advantages in terms of time efficiency, but it is difficult to determine the appropriate number of features that can be selected without a certain amount of domain knowledge or extensive experimentation. Also, this method may not be able to identify the interaction between multiple features [11]. For the embedded methods, it is faster than the wrapper methods because it uses the ranking of features done at the same time as training the classifier model. However, it also suffers from the difficulty of determining the optimal number of features to retain. From the above description, it can be seen that different feature selection algorithms have different advantages and disadvantages. Therefore, the combination of different types of feature selection algorithms is a promising technique to solve the feature selection problem.

The gray wolf optimization (GWO) algorithm is a new population-based heuristic algorithm proposed by Mirjalili et al. [12]. It mimics the process of gray wolf predation in nature. The update of the wolf position relies mainly on learning from the leader individuals (alpha, beta and delta). With the fast convergence and easy-to-use features of the algorithm, the gray wolf optimization algorithm is widely used in power dispatch problems [13], path planning [14], Scheduling [15], feature selection [16], and other engineering fields [17]. However, it is worth noting that the extant feature selection based on the Gray Wolf optimization algorithm deals with data that are usually low-dimensional, and it is not trivial to migrate the algorithm directly to high-dimensional feature selection. This motivates us to combine Grey Wolf optimization algorithms and other data mining techniques such as machine learning techniques to cope with feature selection of high-dimensional data more efficiently.

Evolutionary computation techniques have attracted much attention in the field of feature selection because of their excellent global search capabilities [18–20]. However, there are still two major challenges in the large-scale feature selection problem. First, the search space of the problem is huge. The feature selection problem is often modeled as a discrete optimization problem. For a dataset with D features, the total number of solutions is as high as 2^D. Since the problem is solved on a large scale, the value of D is very large, and the huge search space poses a great challenge to the evolutionary algorithm, leading to a slow convergence of the algorithm. Second, the feature selection process based on the evolutionary algorithm needs to evaluate the fitness of the solution. In most of the current feature selection processes, the corresponding classifier needs to be rebuilt when evaluating the fitness of each solution, which leads to the computation of the fitness is very expensive. The evaluation process takes up the vast majority of the running time of the feature selection process. It seriously affects the efficiency of the algorithm. These two main challenges limit the application of evolutionary algorithms to high-dimensional feature selection problems.

For small samples of high-dimensional biomedical data, a feature selection algorithm combining wrapper and embedded methods is proposed in this paper to avoid the curse of dimensionality. The proposed feature selection process is divided into two stages. In the first stage, the MLP network is first trained by combining neural networks and group lasso regularization techniques. The importance of features and hidden layer neurons are evaluated and ranked according to the weights of the MLP network. Then the proposed IGWO algorithm is used to pre-select the features and optimize the structure of the hidden layer of the network, thus reducing the size of the search space in the second stage of feature selection and solving the problem caused by curse of dimensionality. In the second stage, the feature selection problem is modeled as a combinatorial optimization problem. To enhance the diversity of the algorithm, a new population update strategy and leader enhancement strategy are proposed in this paper, which greatly enhance the diversity of the algorithm. In addition, to address the drawback of expensive solution set evaluation in the wrapper method, this paper proposes a rapid evaluation strategy, which greatly reduces the evaluation cost and improves the efficiency of the feature selection process.

The overall goal of this paper is to propose a two-stage improved grey wolf optimization algorithm for the feature selection problem on high-dimensional data. The method achieves high classification accuracy with a smaller subset of features and smaller time consumption. Specifically, the main contributions of this paper are as follows.

1. To address the first challenge in the large-scale feature selection process, this paper proposes a two-stage improved grey wolf optimization algorithm. Initial filtering of features is performed by modeling the optimization problem as an integer optimization problem using the importance ranking of features in the first stage. This process greatly reduces the search space of the large-scale feature selection problem. By limiting the search space to promising regions, it makes it easier for the improved grey wolf optimization algorithm to converge in the second stage.
2. To address the second challenge in the large-scale feature selection process, the solution is quickly evaluated using a multilayer perceptron neural network. Different solutions use the same multilayer perceptron neural network for fitness evaluation, and the only difference is that different solutions correspond to different neural network weights. The cost of modifying the weights on the trained multilayer perceptron neural network is very cheap. Since our method does not require training the classifier from scratch, this fast evaluation method significantly reduces the time consumption of the fitness computation.

The rest of the paper is organized as follows. Section "Background and related work" briefly describes related work, including neural network, group lasso, gray wolf optimization algorithms, and evolutionary algorithm-based feature selection methods. In section "Proposed method", this paper develops a two-stage improved gray wolf optimization algorithm. And a fast evaluation method is also developed to improve the efficiency of the feature selection process. The details of the experiments and the parameter settings are described in section "Experimental studies". The results of the experiments and the discussion are described in section "Results and discussion". Finally, conclusions and future research directions are given in section "Conclusion".

Background and related work

In this section, firstly, the concepts and definitions of neural networks and group penalization are introduced. Next, the gray wolf optimization algorithm is briefly introduced. Then a brief review of feature selection algorithms based on evolutionary algorithms is given. Finally, the research motivation of our proposed IGWO algorithm is presented.

Neural network

A multilayer perceptron is a classical feed-forward neural network that efficiently solves nonlinear problems [21], such as classification and regression. For classification problems, it maps a set of input sample data into the category space and obtains the posterior probability that a sample belongs to a class, thus classifying the sample.

The MLP neural network is used as a classifier for the evaluation of solution sets. A three-layer MLP network solving a three-classification problem is used as an example for the introduction of the MLP network, as shown in Fig. 1. This neural network consists of an input layer, a hidden layer, and an output layer. Each layer consists of multiple neurons, and the neurons between adjacent layers are densely connected. For the input layer, the number of neurons is determined by the dimensionality of the input data, the number of neurons in the hidden layer q is determined by the user, and the number of neurons in the output layer is determined by the number of categories of the classification problem.

For presentation convenience, $X_i = \left(x_{i1}, x_{i2}, \dots, x_{iD}\right)$ is used to denote the i-th sample data in the dataset, where D denotes the dimensionality of the data. h_i is used to denote the i-th neuron in the hidden layer. The weight of the input layer connected to the hidden layer is called W^1, which is a matrix of $q \times D$. The i-th column of the matrix W^1 is denoted by W^1_{*i} $(i = 1, 2, \dots, D)$, which represents the weight between the i-th feature of the data and all the neurons in the hidden layer. Then for the i-th sample X_i, the output value obtained at the hidden layer is Z_i, which is a $q \times 1$ dimensional vector and is calculated as follows:

$$Z_i = g\left[\left(W^1 \otimes X_i^T\right) + B^1\right] \quad (1)$$

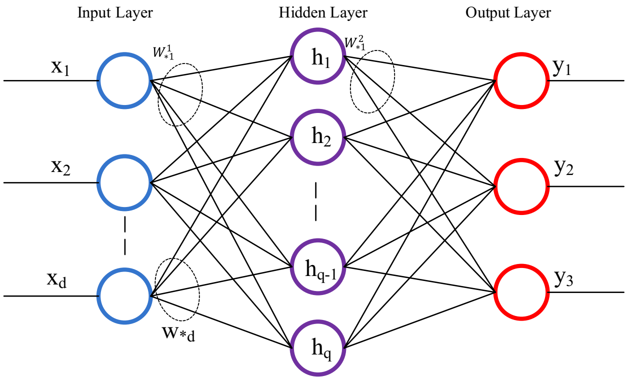

Fig. 1 The architecture of an MLP network

where X_i^T denotes the transposition of the sample X_i, with the symbol $\otimes$ denoting the multiplication of the matrix. The bias of the neurons in the input layer is denoted by B^1, which is a $q \times 1$ vector. g denotes the activation function of the input layer, and the ReLU activation function [22] is used in this paper. The obtained vector Z_i is used as the input data of the next layer.

The weight matrix between the output layer and the hidden layer is called W^2, which is a $3 \times q$ matrix. Then for the ith sample X_i, the output value obtained at the output layer is Y, which is a 3×1 dimensional vector, and is calculated as follows:

$$Y_i = f\left[\left(W^2 \otimes Z_i\right) + B^2\right] \tag{2}$$

where the symbol $\otimes$ denotes the multiplication of the matrix, and B^2 denotes the bias of the neurons in the hidden layer, which is a 3×1 dimensional vector. The activation function of the hidden layer is denoted by f. Since this paper addresses the classification problem, the softmax function is chosen for f. The output Y_i $(i = 1,\ 2,\ 3)$ represents the probability value of the sample X_i predicted as each class, and the one with the largest probability value is the final prediction result.

Consider the $\left[\left(X_1, y_1\right); \ldots ;\left(X_M, y_M\right)\right]$ dataset, where y_i is the label corresponding to the sample X_i and M denotes the size of the dataset. Using stochastic gradient descent, the weights of the neural network are trained, and the weights of the neural network are obtained by minimizing the cost function of the following equation.

$$W^* = \arg\min_{w} \left\{ \frac{1}{M} \sum_{i=1}^{M} L\left(y_i, \hat{y}_i\right) + \lambda \times \emptyset(w) \right\} \tag{3}$$

where L refers to the cross-entropy loss function and $\emptyset(w)$ is a regularization constraint added to the weight term so as to avoid overfitting the model to the training data, and the most common regularization constraint is a squared constraint imposed on the weights. The parameter $\lambda > 0$ is the regularization factor, which controls the relative impact of the empirical error and the regularization term.

Group lasso

W^1_{*i} denotes the i-th column of matrix W^1, which represents the weight between the *i-th* dimensional feature of the data and all hidden layer neurons, and is a $q \times 1$ dimensional vector. The 2-norm of this vector can characterize the importance of the i-th feature to some extent. That is, the larger the 2-norm of W^1_{*i}, the more important the i-th feature is. Formally, the importance metric of the i-th feature can be expressed by the following equation.

$$R_{xi} = \left\| W^1_{*i2} \right\| \tag{4}$$

Similarly, the importance metric of the hidden layer neuron h_i can be calculated using the following equation.

$$R_{hi} = \left\| W^2_{*i2} \right\| \tag{5}$$

Since the purpose of feature selection is to remove the unimportant feature parts and retain the important feature subsets. And Eq. (4) can characterize the importance of features, so the most important feature subset can be found by applying lasso constraints to the importance metrics of all features. The formula is as follows:

$$\emptyset_{21}(w) = \sum_{i=1}^{D} \left\| W^1_{*i2} \right\| \tag{6}$$

This way of lasso constraining the whole vector as a whole is called the group lasso technique [23]. Similarly, to remove both unimportant features and unimportant hidden layer neurons, the group lasso penalty should be applied to both the input and hidden layers of the MLP with the following equation [24].

$$\emptyset_{21}(w) = \sum_{i=1}^{D} \left\| W^1_{*i2} \right\| + \sum_{j=1}^{q} \left\| W^2_{*j2} \right\| \tag{7}$$

Gray Wolf optimization algorithm

GWO algorithm mimics the characteristics of grey wolf predation in nature. The process of predation can be considered as a process of finding the optimal solution. Without loss of generality, we consider a D-dimensional minimization optimization problem in which the smaller the fitness value of an individual, the better its performance. Suppose the size of the population is N, and $P_i = \left(p_{i1}, p_{i2}, \ldots, p_{iD}\right)$ denotes the position of the i-th individual in the population. The following equation is used to represent the predatory behavior of P_i.

$$p_{id}^{(t+1)} = p_{fd} - a^{(t)} \times r_1 \times \left| r_2 \times p_{fd} - p_{id}^{(t)} \right| \tag{8}$$

where $d = 1, 2, \ldots, D$, r_1 is a random number in the range -1 to 1 and r_2 is a random number in the range 0 to 2. The diversity of the algorithm is increased by these two random numbers, and t represents the number of current iterations. $p_{id}^{(t)}$ is the position of the i-th individual at iteration t in the d-th dimension. p_{fd} is used to represent the position of the prey (optimal solution). The value of $a^{(t)}$ decreases linearly as the iteration progresses from 2 to 0, allowing the algorithm to transition its search properties from an emphasis on exploration to an emphasis on exploitation [12].

In GWO, the population is divided into four classes of individuals: alpha, beta, delta, and omega. The populations are ranked according to their fitness values from smallest to largest. The top three individuals are referred to as alpha, beta, and delta individuals in that order, and the remaining individuals are referred to as omega. Since it is impossible to know the location of the prey (optimal solution) in advance in a real optimization problem, Eq. (8) cannot be applied in practice. An easy way to think of a solution is to approximate the position of the optimal solution by the high-performing individuals in the current population. In the GWO algorithm, the alpha, beta, and delta individuals are used to approximate the position of the optimal solution, and the other individuals update their positions by learning from alpha, beta, and delta, respectively, and the position update is represented by the following four equations.

$$p_{id1}^{(t+1)} = p_{1d}^{(t)} - a^{(t)} \times r_1 \times \left| r_2 \times p_{1d}^{(t)} - p_{id}^{(t)} \right| \tag{9}$$

$$p_{id2}^{(t+1)} = p_{2d}^{(t)} - a^{(t)} \times r_1 \times \left| r_2 \times p_{2d}^{(t)} - p_{id}^{(t)} \right| \tag{10}$$

$$p_{id3}^{(t+1)} = p_{3d}^{(t)} - a^{(t)} \times r_1 \times \left| r_2 \times p_{3d}^{(t)} - p_{id}^{(t)} \right| \tag{11}$$

$$p_{id}^{(t+1)} = \frac{p_{id1}^{(t+1)} + p_{id2}^{(t+1)} + p_{id3}^{(t+1)}}{3} \tag{12}$$

where $p_{1d}^{(t)}$, $p_{2d}^{(t)}$, and $p_{3d}^{(t)}$ denote the d-th dimensional position of alpha, beta, and delta individuals in the population, respectively. r_1, r_2 and $a^{(t)}$ are defined as in Eq. (8). $p_{id1}^{(t+1)}$,$p_{id2}^{(t+1)}$ and $p_{id3}^{(t+1)}$ denote the new positions obtained by individual P_i learning from alpha, beta, and delta individuals, respectively. The final new position learned by P_i is obtained by averaging these three new positions.

Related works

In this sub-section, feature selection algorithms based on evolutionary algorithms are reviewed. Depending on the evolutionary algorithms used, feature selection algorithms can be mainly classified into GA-based feature selection algorithms, PSO-based feature selection algorithms and GWO-based feature selection algorithms.

Genetic algorithm-based feature selection algorithms

A large number of genetic algorithm (GA)-based feature selection methods have been proposed by researchers. In [25], Hong et al. proposed an evolutionary algorithm based on speciated GA. This algorithm on one hand reduces the dimensionality of the search space on high-dimensional datasets by a special encoding form. On the other hand, it uses the niching technique to avoid the algorithm from falling into local optimum. In Ding et al. [26], proposed an optimization algorithm that hybridized genetic algorithm and competitive swarm algorithm to solve the feature selection problem. The crossover operator and variation operator from the genetic algorithm were added to the competitive swarm optimization to improve the diversity of new individuals in the algorithm and prevent premature maturation of the population. The results of the study show that the algorithm improves the computational efficiency while increasing the classification accuracy. In Amini et al. [27], proposed a two-layer feature selection method that combines a wrapper and an embedded method to select a subset of features. In the first layer, a genetic algorithm is used to search for the optimal subset of features. In the second layer, an elastic net is used to eliminate the remaining irrelevant features to reduce the number of feature subsets and the prediction error.

Particle swarm optimization-based feature selection algorithms

In recent decades, researchers have proposed a large number of feature selection methods based on particle swarm optimization algorithms. In Kennedy and Eberhart [18], proposed the discrete particle swarm algorithm (BPSO) for the feature selection problem. The particle swarm algorithm finds the best solution by mimicking the social behavior of bird flocking. For the feature selection problem, the best solution represents the best subset of features searched by the algorithm. However, traditional particle swarm algorithms do not solve large-scale optimization problems well, and the search efficiency of particle swarm algorithms decreases dramatically when the size of the optimization problem increases.

In a self-adaptive particle swarm optimization algorithm called SaPSO was proposed by Xue et al. [28], for large-scale feature selection problems. To solve the basic problem in self-adaptive-based design, an improved analytic hierarchy process method was introduced in the paper. Experimental results on 12 datasets show that the SaPSO algorithm has advantages in both classification accuracy and feature subset size. In Tran et al. [29], proposed an adaptive multigroup particle swarm algorithm. Based on the importance of features, the method divides the entire search space into a series of subspaces. During the evolution process, the subpopulations are automatically and dynamically changed according to their performance. The results of the study show the advantage of the method in high-dimensional feature selection problems. In Xue et al. [30], introduced the adaptive mechanism of policy and parameters into the particle swarm optimization algorithm and designed a feature selection algorithm called SPS-PSO. The results show that the adaptive mechanism can significantly improve the performance of the evolutionary algorithm on the feature selection problem. In Cheng et al. [19], proposed a competitive swarm optimization (CSO) algorithm, which is used to solve large-scale real-valued optimization problems. In this algorithm, the particles are randomly divided into two groups, and each group of particles competes with each other in pairs. After each competition, the winning particle goes directly to the next generation, while the losing particle learns knowledge from the winning particle to update its position. Experimental results show that the algorithm is suitable for solving large-scale optimization problems. Since the algorithm was originally developed for real-valued optimization problems. In the discrete CSO algorithm for feature selection was proposed by Gu et al. [20], To reduce the computational cost of solution set evaluation, a solution set archiving technique is introduced in the paper. Experimental results show that the CSO algorithm has a competitive advantage over some particle swarm algorithms for the feature selection problem.

Gray Wolf optimization-based feature selection algorithms

The GWO algorithm has been widely used to solve the feature selection problem due to its few control parameters, adaptive exploration behavior and simplicity of the mechanism. In Too et al. [31], proposed an opposition-based competitive gray wolf optimization algorithm (OBCGWO) to deal with the feature selection problem in electromyography (EMG) signal classification. The performance of GWO is improved by the opposition-based learning (OBL) strategy and the competitive strategy. The proposed method achieves better results in several EMG signal classifications. In Hu et al. [32], proposed a novel binary gray wolf optimization algorithm. To solve the discrete optimization problem, the paper proposed a novel parameter update formulation and transfer function. The feature selection experiments on the UCI dataset show that the proposed algorithm obtains lower classification error and fewer features compared to the original binary gray wolf optimization algorithm. In Chantar et al. [33], proposed an improved elite-based crossover binary GWO algorithm to implement the document classification task. Experimental results show that support vector machine-based feature selection techniques combined with binary GWO and elite-based crossover schemes provide better performance on the document classification task.

In general, although there have been a large number of evolutionary algorithm-based feature selection methods to solve the feature selection problem. However, there are not many studies that simultaneously solve the problems of huge search space and expensive fitness evaluation faced in large-scale feature selection problems. In this paper, a two-stage search strategy is used to solve the problem of too large a search space. A rapid fitness evaluation mechanism is used to solve the problem of high time cost of fitness evaluation. Meanwhile, GWO, as a newly proposed swarm optimization algorithm, has the advantages of simple structure, less parameters to be set and easy implementation of coding. The GWO algorithm is used as the search operator in this paper.

Proposed method

In this section, to improve the performance of the GWO algorithm, an improved GWO algorithm is proposed in this paper. It is also applied to a two-stage feature selection process. In the first stage of feature selection, the size of the feature set and the structure of the hidden layer are optimized using the proposed IGWO algorithm. This stage not only reduces the size of the feature set, but also finds the optimal number of neurons in the hidden layer. In the second stage, the feature selection process is constructed as a combinatorial optimization problem and the proposed IGWO algorithm is used to find the most meaningful features. At the end of the feature selection process, the performance of the feature selection algorithm is examined on the test set.

Improved Gray Wolf optimization algorithm

Although the GWO algorithm has been widely used in solving continuous optimization problems, however, GWO has some shortcomings, for example, in population updating, the population only uses learning from the best three solutions (leaders) to update its position, which will lead to low diversity and premature convergence [34]. Therefore, performance improvement of the GWO algorithm is necessary and will be described in detail below.

Population representation

IGWO is a population-based evolutionary optimization algorithm. The population contains N individuals, and each individual in the population is encoded as a vector of length D. The individuals are represented by the following equation.

$$x_{id} = lb_d + \text{rand} \times \left(ub_d - lb_d \right) \quad d \in [1, D] \tag{13}$$

where lb_d denotes the minimum value of the d-th dimension of the vector, ub_d denotes the maximum value of the d-th dimension of the vector, and rand denotes a random number from 0 to 1.

Position updating

In the GWO algorithm, the population only uses learning from the best three individuals (leaders) to update its position, which limits the diversity of the algorithm. To solve this problem, this paper uses a diverse leader selection strategy to increase the diversity of the algorithm. Specifically, the populations are ranked in order of fitness values from smallest to largest. Assuming that the size of the population is N, each individual generates a random integer n ranging from 1 to $\frac{N}{2}$ when choosing a learning target, and then averages the positions of the first n individuals in the population to obtain a new individual P_l. Taking this new individual P_l as its own learning target, the equation is as follows:

$$P_l = \frac{\sum_{i=1}^{n} P_i}{n} \quad n \in \left\{1, 2, \dots \frac{N}{2}\right\} \tag{14}$$

$$p_{id}^{(t+1)} = p_{fd} - a^{(t)} \times r_1 \times \left| r_2 \times p_{ld} - p_{id}^{(t)} \right| \tag{15}$$

Obviously, when the random number n is equal to 3, the result of selecting leaders in the paper is similar to the original GWO. Since individuals include randomness in selecting leaders, each individual may choose different learning objects. This strategy helps to maintain the diversity of the population, prevent early convergence of the algorithm, and increase the diversity of the algorithm, and increase the diversity of the algorithm. When the random number n is smaller, it indicates that individuals learn from the best individuals. When the random number n is larger, it indicates that individuals learn from the collective.

Adaptive learning rate

According to the proposal of the paper [35], not all individuals need to be updated in position at each iteration period. The criterion often used is that the worse the performance of an individual, the higher the probability that it will learn from other superior individuals. According to the above criterion, the learning probability of the i-th individual is defined as follows:

$$L_i = \left(\frac{i}{N}\right)^{0.5\times\log\left(\frac{D}{100}\right)} \tag{16}$$

where N represents the population size and D represents the dimensionality of the optimization problem. i is the index value of individuals in the population. The populations are sorted in ascending order according to fitness values. When i is larger, it means that the corresponding individual in the population has a larger fitness value, its performance is worse, and its learning probability L_i is larger.

Rapid evaluation strategy

A feature selection algorithm based on wrapper methods uses classifier performance as an evaluation metric for a subset of features. This evaluation method has the advantages of being straightforward and accurate. However, evaluating each feature subset requires retraining a classifier. This causes the method to be computationally expensive, especially for high-dimensional feature data. A rapid evaluation method was developed to address this problem. Specifically, the process of modifying the weights of the MLP network is considered as feature selection of the data. Formally, the modification of the MLP network can be expressed by the following equation.

$$NewW^1 = W^1 \odot G \tag{17}$$

where G is used as a mask so that the weight value at the corresponding column position of W^1 is modified to 0. The symbol $\odot$ denotes the elementwise product operator.

An example is used to illustrate this process, as shown in Fig. 2. The input matrix W^1 of the MLP is first extracted, and the input matrix W^1 is modified according to the content of the mask vector G. The mask G then represents a feature subset, such as G encoded as vector [1 0 0]. The content of this encoding indicates that the first dimensional features of the data are retained while the second and third dimensional features are discarded. And this meaning can be approximated by modifying all the second and third column values of the matrix W^1 to 0. In this encoding, the new input matrix $NewW^1$ can be obtained and the input matrix of the original MLP network is replaced with the $NewW^1$ to generate a new MLP network. It is important to note that the bias of all neurons is kept constant, since the elimination of certain features and neurons can be achieved by the operation of setting 0 to the corresponding columns of the weight matrix. Then the original data are input to the modified MLP network

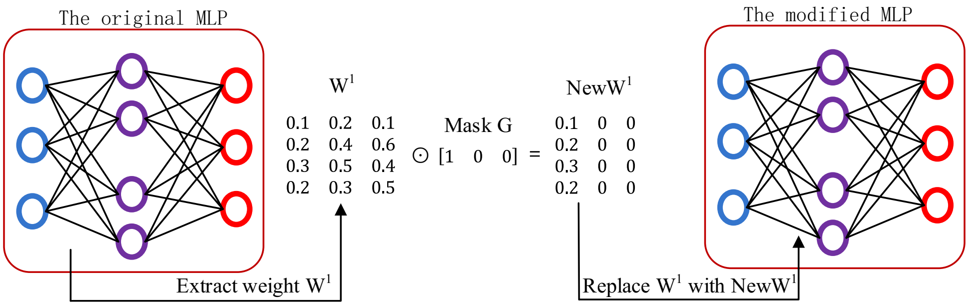

Fig. 2 Example of modifying MLP network weights

to calculate the classifier performance. With this strategy, all feature subsets can calculate performance metrics on the same classifier model without retraining the network. This strategy allows a rapid evaluation of the solution set to improve the efficiency of the feature selection process. Similarly, the hidden layer of the neural network can be compressed. Formally, the compression of the hidden layer can be expressed by the following equation.

$$NewW^2 = W^2 \odot O \tag{18}$$

where O is used as a mask so that the weight value at the corresponding column position of W^2 is modified to 0. By this operation, an action similar to the elimination of the corresponding neuron is thus reached.

Fitness function

Two aspects are considered in the design of the fitness function in this paper. On one hand, the accuracy of the classification results is considered. Since the data classified in this paper are high-dimensional unbalanced data, balanced accuracy is used in this paper. The higher the balance accuracy is, the smaller the fitness value is. On the other hand, the sparsity of features needs to be considered, and the sparsity of features refers to the proportion of the selected features to the total number of features. The fewer features are selected, the smaller the fitness value. The fitness function is shown in the following equation.

$$\text{fitness} = (-\alpha) \times w + \frac{d}{D} \times (1 - w) \tag{19}$$

where d denotes the number of features selected, D is the total number of features, and α is the average of the balanced accuracy of the classifier on the validation set. w is a control parameter that controls the interaction of sparsity and balanced accuracy. w has values ranging from 0 to 1. Larger values of w indicate that the model places more emphasis on the classification performance. It should be noted that this paper constructs a minimization optimization problem, i.e., the smaller the value of the fitness of the individual, the higher the quality of the individual.

The K-fold protocol ($K = 5$) is used for cross-validation to alleviate the overfitting problem during feature selection. Specifically, the training set is divided equally into five parts, with one part serving as the validation set and the remaining four parts as the learning set. The process is repeated for all five parts. The learning set is used for MLP training, and the validation set is used to characterize the generalization performance of the MLP network. The average of the balanced classification accuracy of the MLP network on the validation set is denoted as α, and α is calculated as follows:

$$\alpha = \frac{1}{5}\sum_{i=1}^{5} \text{balanced accuracy}_i \tag{20}$$

where the balanced accuracy is calculated as follows:

$$\text{balanced accuracy} = \frac{1}{c}\sum_{i=1}^{c} \text{accuracy}_i \tag{21}$$

where c is the number of categories of the classification problem and $accuracy_i$ is the accuracy rate of the i-th category.

Feature selection in the first stage

Due to the high dimensionality of the high-dimensional feature selection problem, directly modeling the feature selection problem as a combinatorial problem will lead to a huge search space for the optimization problem, which is not conducive to searching for the optimal solution. Therefore, in the first stage of feature selection, a two-dimensional integer optimization problem is constructed to perform a coarse search of the search space. This will be described in detail below.

Population representation

In the first stage, the feature subset length Len and the number of neurons q in the hidden layer need to be optimized in this stage, and the proposed IGWO algorithm is used to optimize these two parameters simultaneously. Each individual in the population is represented as shown in Eq. (22).

$$\begin{cases} p_{i1} \in [1, U] & \text{Number of features selected} \\ p_{i2} \in [10, M] & \text{Number of selected hidden layer neurons} \end{cases} \tag{22}$$

where p_{i1} denotes the first dimension of the i-th individual in the population, which is the number of features to be retained. p_{i1} is an integer from 1 to U, where U is the maximum value of dimensionality in the integer optimization problem. p_{i2} denotes the second dimension of the i-th individual in the population, which is the number of hidden layer neurons to be retained.p_{i2} is an integer from 10 to M, where M is the number of samples.

As an illustrative example, Fig. 3 shows how the individuals in the population are used to modify the MLP network in the first stage of the algorithm. For the convenience of description, for the input matrix W^1, it is assumed that the importance of the represented features decreases as the number of columns of W^1 increases. For the output matrix W^2, it is assumed that the importance of the represented hidden layer neurons decreases as the number of columns of W^2 increases. Taking $P_1 = [1, 2]$ as an example, first because P_{11} takes the value of 1, this means that only the top one most

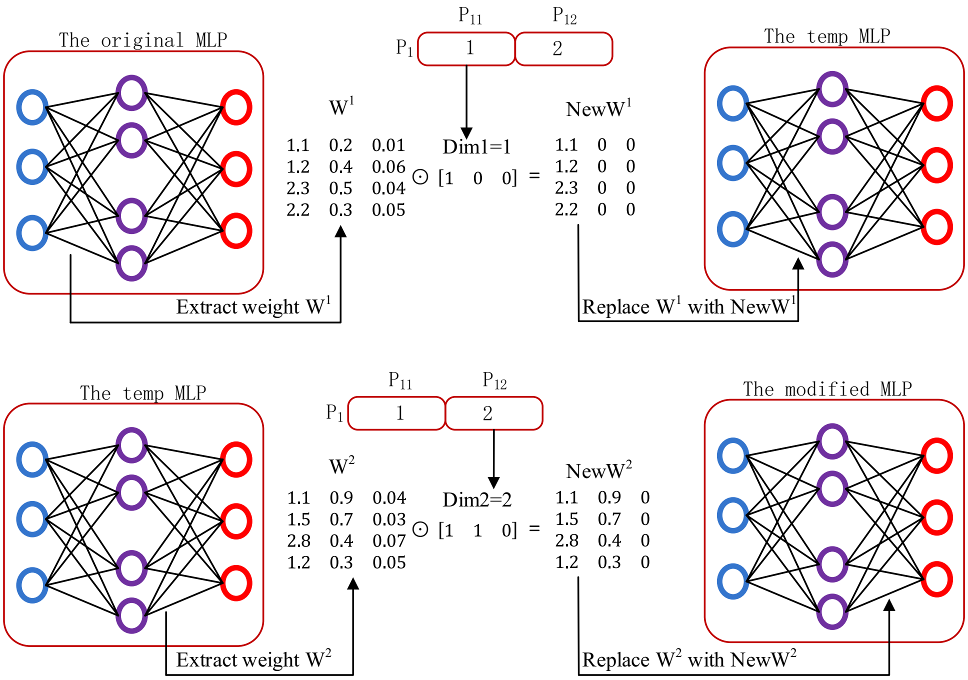

Fig. 3 Example of modifying the MLP network using the solution

important feature is retained. The matrix W^1 is extracted from the original MLP network and the input matrix is updated to obtain the new MLP network named temp using Eq. (17). Since P_{12} takes the value of 2, this means that only the top two most important neurons are retained. The matrix W^2 is extracted from temp's MLP network and the output matrix is updated by Eq. (18) to obtain the modified MLP network. Finally, the fitness of P_1 was evaluated using the modified MLP network.

Improved Gray Wolf optimization algorithm for integer optimization

At this stage, the IGWO algorithm is used for integer optimization. For ease of representation, the algorithm is referred to as the IGWO1 algorithm. The whole algorithm consists of two parts. In part 1, the training set is first divided into a learning set and a validation set, where the learning set is used to train the MLP neural network and the validation set is used to test the performance metrics of features subset. In training the MLP network, the number of neurons in the hidden layer is taken as the maximum, i.e., the number of samples. To be able to optimize both the number of features and the number of neurons in the hidden layer, the group lasso constraint is imposed on both the input layer weights and the hidden layer weights. The model is trained using stochastic gradient descent to obtain a redundant MLP network named NN. After obtaining the MLP network, the importance metrics R_1 and R_2 of features and neurons are calculated according to Eqs. (4) and (5), respectively. Then, the population is initialized using Eq. (13), while the learning rate of each individual in the population is calculated using Eq. (16).

The part 2 of Algorithm 1 is an iterative process. The fitness values of the individuals need to be calculated at each iteration. It should be noted that the population obtained by initializing using Eq. (13) is encoded in a real number mode, while the two parameters to be optimized are in integer mode. The algorithm is illustrated using the i-th individual P_i in the population as an example. Firstly, the downward rounding operation of P_i is needed to obtain $tempP_i$. $tempp_{i1}$ and $tempp_{i2}$ are used to denote the first and second dimensions of $tempP_i$, respectively. Then a copy of NN, $tempNN$, is generated and modified to simulate the selection process of features. Specifically, according to the R_1 metric, only the top $tempp_{i1}$ most important features of the network $tempNN$ are retained using Eq. (17). According to the R_2 metric, only the top $tempp_{i2}$ most important hidden layer neurons of the network $tempNN$ are retained using Eq. (18). After that, the validation set is input to the modified $tempNN$ network and the fitness value of individual P_i is calculated using Eq. (19).

When the fitness values of all individuals in the population are calculated, the individual with the lowest fitness value is alpha. Alpha is the first individual in the population when the fitness values are sorted in ascending order. After all the above tasks are completed, the individual position update operation is started. During each individual update, a random number θ from 0 to 1 is generated. if θ is less than the learning rate L_i, the position of the individual P_i is updated using Eqs. (14) and (15). The algorithm repeats iterations until the maximum number of iterations is satisfied.

At the end of the algorithm, based on the alpha value of the last generation and the feature importance metric R_1, the algorithm outputs the feature index to be retained and the optimal number of hidden layer neurons.

Algorithm1: Proposed IGWO1 algorithm

```
Input: training set; feature dimension $D$; population size $N$; maximum iteration period $T$;
Output: Optimized feature index and optimized number of hidden layer neurons;
Split training into learning and validation sets;
$q$= The number of training sets;
Train an MLP network named $NN$ with hidden layer size $q$ using the learning set;
Using Equation 4 to calculate the importance metric $R_1$ of the features;
Using Equation 5 to calculate the importance metric $R_2$ of the hidden layer neurons;
Initialization of the population using Equation 13;
$t$=0;
Using Equation 16 to set the learning probability L of individuals;
while $t < T$
    for $i$ = 1 to $N$
        $tempP_i$ =▯Pi▯;
        $tempNN$ = $NN$;
        According to the $R_1$ metric, the first $tempp_{i1}$most important features of $tempNN$ are retained using Equation 17;
        According to the $R_2$ metric, the first $tempp_{i2}$ most important hidden layer neurons of $tempNN$ are retained using Equation 18;
        Input the validation set to the modified $tempNN$ network and
        calculate the fitness value $F_i$ of $P_i$ using Equation 19;
    end for
    Update best individual $alpha$;
    The population are ranked in ascending order of fitness values;
    for $i$ = 1 to $N$
        Generating a random number $\theta$ from 0 to 1;
        if $\theta < L_i$
            Using Equation 14 and Equation 15 to update $P_i$;
        end if
    end for
    $t$= $t$+1;
end while
Based on the $alpha$ and $R_1$ values, the optimal feature index value and optimalnumber of hidden layer neurons are returned.
```

Feature selection in the second stage

Based on the optimal solution searched in the first stage of feature selection, the promising region in the search space can be obtained. Then, in the second stage, the search space is restricted to the promising region, thus reducing the search difficulty of the problem. In the second stage, the feature selection problem is modeled as a discrete optimization problem so that the promising region is finely searched. Since the size of the problem is significantly reduced, this makes it easier for the evolutionary algorithm to converge. This is described in detail below.

Population representation

The value range of each dimension of an individual is real numbers from 0 to 1, while a threshold parameter is used to determine whether a feature is selected or not, as shown in Eq. (23).

$$\begin{cases} p_{id} \geq 0.6 \text{ Retained feature} \\ p_{id} < 0.6 \text{ No retained feature} \end{cases} \tag{23}$$

where p_{id} denotes the d-th dimensional value of the i-th individual in the population. If $p_{id} \geq 0.6$ means that the corresponding feature is retained, and if $p_{id} < 0.6$ means that the corresponding feature is not retained. The dimension of an individual is equal to the number of features retained in the first stage of feature selection. In this encoding mode, each individual in the population represents a candidate solution for a feature subset, and the optimal feature subset is obtained by iterative optimization.

Proposed leader enhancement algorithm

Alpha, beta and delta individuals play an important leading role in the GWO algorithm. Individuals tend to move to a better position by learning from these leaders. To prevent the proposed IGWO algorithm from falling into a local optimum, these leaders can enhance themselves through leader enhancement strategies. Algorithm 2 describes the proposed enhancement strategy algorithm. In this algorithm, the enhancement search process is divided into local mutation search and global mutation search, both with equal probability. The maximum number of features that can be changed by local mutation is defined as β, and the maximum number of features that can be changed by the global mutation process is 10 times β. When the number of features S to be mutated is determined, S features are randomly added or deleted from the leader P_i with equal probability, thus generating a new individual $NewP_i$.

Algorithm2: Proposed leader enhancement algorithm

```
Input: particle $P_i$, Mutation of intensity $\beta$
Output: new particle $NewP_i$
Generating a random number $\theta$ from 0 to 1;
Generate a random integer $Q$, which takes values from 1 to $\beta$;
if $\theta$ >0.5
    $S = Q$;
else
    $S = 10 \times Q$;
end if
Generating a random number $\theta$ from 0 to 1;
if $\theta$ >0.5
    $S$ elements of 1 in $P_i$ are modified to 0, and a new particle
    $NewP_i$ is obtained;
else
    $S$ elements of 0 in $P_i$ are modified to 1, and a new particle
    $NewP_i$ is obtained;
end if
```

Improved Gray Wolf optimization algorithm for combinatorial optimization

At this stage, the IGWO algorithm is used for combinatorial optimization. For ease of representation, the algorithm is referred to as the IGWO2 algorithm. Algorithm 3 describes the pseudo-code of the proposed IGWO2 algorithm. The whole algorithm consists of two parts. In part 1, a new training set is generated by compressing the training set based on the feature index obtained in the first stage of feature selection. The number of dimensions of this combinatorial optimization problem is equal to the number of dimensions of the samples in the new training set. The new training set is then divided into a learning set and a validation set, where the learning set is used to train the MLP neural network and the validation set is used to test the performance metrics of features subset. In training the MLP network, the number of neurons in the hidden layer is taken to be the optimal number of hidden layers obtained by optimization in the first stage of feature selection. The group lasso constraint is imposed on the input layer weights and the model is trained using stochastic gradient descent to obtain a redundant MLP network named NN. Then, the population is initialized using Eq. (13). Also, the learning rate of each individual in the population is calculated using Eq. (16).

The part 2 of Algorithm 3 is an iterative process. It should be noted that the individuals need to be discretized before calculating the fitness value of an individual. The algorithm is illustrated by taking the i-th individual P_i in the population as an example. The mask $tempP_i$ is obtained by discretizing P_i according to Eq. 23, and then the copy $tempNN$ of NN is modified according to the value of the mask $tempP_i$ using Eq. (17). After that, the validation set is input to the modified $tempNN$ network and the fitness value of individual P_i is calculated using Eq. (19).

When the fitness values of all individuals in the population have been calculated, the fitness values are ranked in ascending order. After all the above tasks are completed, the individual position update operation is started. For the leaders, i.e., the first three individuals in the population, Algorithm 2 is used to update them. For the other individuals in the update, a random number θ from 0 to 1 is generated. if θ is less than the learning rate L_i, the position of the particle P_i is updated using Eqs. (14) and (15). The algorithm repeats iterations until the maximum number of iterations is satisfied. At the end of the algorithm, the alpha of the last generation is output.

Algorithm3: Proposed IGWO2 algorithm

```
Input: training set; feature index; population size $N$; maximum iteration period $T$; Number of hidden layers $q$
Output: $alpha$
Based on the feature index, the training set is compressed to obtain a new training set;
Split the new training set into learning and validation sets;
Train an MLP network named $NN$ with hidden layer size $q$ using the learning set;
Initialization of the population using Equation 13;
$t$=0;
Using Equation 16 to set the learning probability $L$ of individuals;
while $t < T$
    for $i$ = 1 to $N$
        According to Equation 23, the discretization operation of $P_i$ generates $tempP_i$;
        $tempNN$ = $NN$;
        Using Equation 17, the copy $tempNN$ of $NN$ is modified according to the value taken by $tempP_i$;
        Input the validation set to the modified $tempNN$ network and calculate the fitness value $F_i$ of $P_i$ using Equation 19;
    end for
    Update best individual $alpha$;
    The population are ranked in ascending order of fitness values;
    for $i$ = 1 to 3
        Using Algorithm 2 to mutate $P_i$ to generate a new individual $NewP_i$;
        Using Equation 19 to calculate the fitness value $NewF_i$ of $NewP_i$;
        if $NewF_i < F_i$
            Update $P_i$ with $NewP_i$;
        end if
    end for
    for $i$ = 4 to $N$
        Generating a random number $\theta$ from 0 to 1;
        if $\theta < L_i$ then
            Using Equation 14 and Equation 15 to update $P_i$;
        end if
    end for
    $t$= $t$+1;
end while
Return the $alpha$ as the final solution.
```

Table 1 Datasets

Datasets	Features	Instances	Class	Smallest class (%)	Largest class (%)
SRBCT	2308	83	4	13	35
DLBCL	5469	77	2	13	53
9Tumor	5726	60	9	25	75
Leukemia 1	5327	72	3	3	15
Brain tumor 1	5920	90	5	4	67
Leukemia 2	11,225	72	3	14	30
Brain tumor 2	10,367	50	4	49	51
Prostate	10,509	102	2	28	39
Lung cancer	12,600	203	5	4	16
11 Tumor	12,533	174	11	3	68

Table 2 Algorithm parameter setting

Parameter	Value
Population size N	50
Maximum iterations T	100
Control parameter w	0.99
Mutation of intensity β	10
The maximum dimension of the integer optimization U	1000
Number of training epoch	150
Regularization factor λ	0.1

Computational complexity of two-stage improved Gray Wolf optimization

The computational complexity of the two-stage IGWO algorithm depends on three main components: leader selection, population ranking, and fitness evaluation. First, each individual in the population needs to choose its own leader and learn from the leader. The computational complexity of a population consisting of N individuals in choosing a leader is $O(N)$. Second, at each iteration cycle, the population needs to be sorted according to the fitness value. The worst computational complexity to perform the sorting is $O\left(N^2\right)$. Finally, the fitness evaluation is performed on the individuals. Each time an individual is evaluated for fitness, the MLP network needs to perform a feedforward propagation, and its computational complexity is $O(D \times q + q \times c)$, where D denotes the dimensionality of the data input to the MLP network, q denotes the number of neurons in the hidden layer, and c denotes the number of categories for classification. Then the complexity of evaluation for N individuals is $O(N \times (D \times q + q \times c))$, and the iteration period of the algorithm T, then the total complexity of the algorithm is $O\left(T \times \left(N + N^2 + N \times (D \times q + q \times c)\right)\right)$.

Experimental studies

In this section, the performance of the proposed two-stage IGWO algorithm is evaluated and compared with some evolutionary algorithm-based feature selection algorithms and traditional feature selection algorithms.

Datasets

To evaluate the performance of our proposed method, 10 gene expression datasets are selected for performance testing of the algorithm in this paper. The characteristics of these datasets are summarized in Table 1, which shows the number of features, the number of samples, the number of classes classified, and the percentage of instances in the smallest and largest classes for each dataset. From the table, it can be seen that these datasets are characterized by high dimensional small samples, which helps to distinguish the performance differences of the algorithms. In addition, these datasets can be publicly downloaded on the http://www.gemssystem.org.

Due to the small number of samples in the gene expression dataset, ten-fold cross validation was used to create the training and test sets. As shown in Fig. 4, the data are

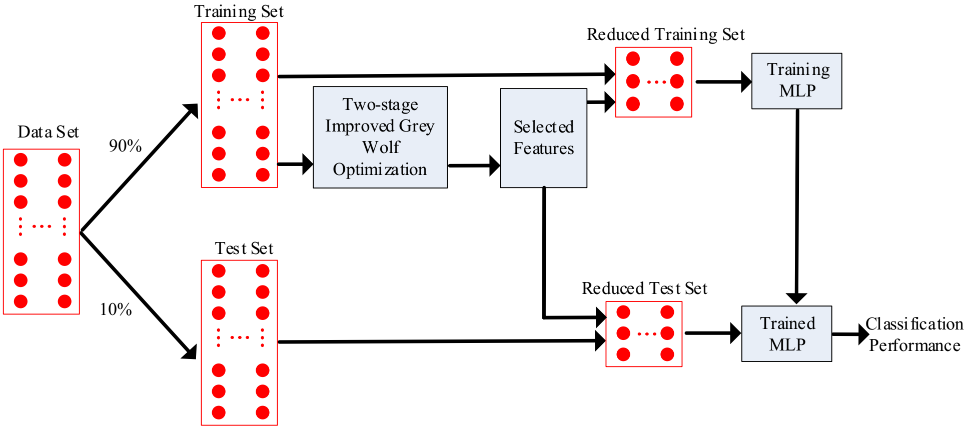

Fig. 4 Overview of feature selection in 1 of the ten-fold cross validation

Table 3 Average test results

Datasets	Algorithm	Time (m)	Feature size	Best	Mean ± Std	S
SRBCT	Full		2308.0	87.08		+
	PSO	8.2	1119.4	92.50	89.51 ± 1.56	+
	CSO	19.9	85.4	**100.00**	93.29 ± 3.52	+
	Two-stage IGWO	**4.5**	**29.5**	**100.00**	**99.14 ± 0.64**	
DLBCL	Full		5469.0	83.00		+
	PSO	47.6	2681.0	86.33	83.67 ± 1.52	+
	CSO	394.8	**30.1**	**100.00**	**94.30 ± 4.05**	=
	Two-stage IGWO	**4.6**	45.8	98.30	93.00 ± 3.60	
9 tumor	Full		5726.0	36.67		+
	PSO	39.2	2811.9	45.00	42.72 ± 1.42	+
	CSO	373.4	**220.3**	**68.33**	59.50 ± 3.72	=
	Two-stage IGWO	**4.6**	243.8	63.33	**60.28 ± 2.24**	
Leukemia 1	Full		5327.0	79.72		+
	PSO	41.2	2615.5	87.36	80.60 ± 2.55	+
	CSO	251.8	170.1	**96.81**	88.45 ± 3.90	+
	Two-stage IGWO	**4.7**	**27.6**	94.17	**93.47 ± 0.66**	
Brain tumor 1	Full		5920.0	72.08		+
	PSO	66.7	2917.2	77.08	73.73 ± 2.21	+
	CSO	462.1	207.6	**86.67**	**79.93 ± 3.09**	=
	Two-stage IGWO	**4.8**	**189.0**	82.50	78.89 ± 2.75	
Leukemia 2	Full		11,225.0	89.44		+
	PSO	120.6	5535.7	92.22	89.83 ± 1.00	+
	CSO	1845.2	88.6	**98.33**	91.72 ± 3.16	=
	Two-stage IGWO	**5.5**	**45.6**	97.22	**94.91 ± 1.80**	
Brain tumor 2	Full		10,367.0	62.50		+
	PSO	80.5	5117.2	67.08	61.99 ± 2.91	+
	CSO	950.8	**90.4**	**90.83**	**80.44 ± 6.28**	−
	Two-stage IGWO	**5.8**	182.8	79.17	74.03 ± 3.65	
Prostate	Full		10,509.0	85.33		+
	PSO	160.6	5193.7	88.33	86.00 ± 1.49	+
	CSO	2369.9	357.2	**95.17**	88.99 ± 2.68	=
	Two-stage IGWO	**5.8**	**41.8**	94.33	**92.17 ± 2.03**	
Lung cancer	Full		12,600.0	78.05		+
	PSO	574.2	6234.7	82.72	78.77 ± 1.53	+
	CSO	5565.9	226.4	93.79	87.72 ± 2.93	+
	Two-stage IGWO	**6.7**	**95.4**	**98.29**	**95.64 ± 1.43**	
11 tumor	Full		12,533.0	71.42		+
	PSO	418.5	6205.0	75.59	71.81 ± 1.75	+
	CSO	6288.6	588.6	84.47	79.52 ± 2.35	+
	Two-stage IGWO	**6.7**	**236.7**	**93.05**	**90.59 ± 1.98**	

divided into ten parts, nine of which are the training set and are used during the feature selection process. The remaining one part is used as the test set, which is never used in the feature selection process. Using the training set, the optimized feature subset is obtained by searching under the proposed two-stage algorithm. After the feature selection process is completed, the training and test sets are dimensionally compressed according to the selected features. The compressed training set is used to train the MLP neural network. The compressed test set is also tested on the trained MLP network. The performance of the feature selection method is evaluated in terms of the accuracy of the classifier on the compressed test set. It is important to note that this process needs to be repeated ten times.

Table 4 Average test results

Datasets	Algorithm	Time (s)	Feature size	Best	Mean ± Std	S
SRBCT	LFS	**25.0**	**7.1**	91.67		+
	CFS	243.3	112.3	99.17		=
	Two-stage IGWO	271.1	29.5	**100.00**	99.14 ± 0.64	
DLBCL	LFS	**56.3**	**5.9**	83.33		+
	CFS	778.4	86.3	93.00		=
	Two-stage IGWO	275.5	45.8	**98.30**	**93.00 ± 3.60**	
9Tumor	LFS	**52.9**	**9.7**	26.67		+
	CFS	341.2	44.0	56.67		+
	Two-stage IGWO	278.1	243.8	**63.33**	**60.28 ± 2.24**	
Leukemia 1	LFS	**51.9**	**5.4**	85.14		+
	CFS	778.4	79.4	92.08		+
	Two-stage IGWO	279.4	27.6	**94.17**	**93.47 ± 0.66**	
Brain tumor 1	LFS	**77.9**	**12.2**	63.33		+
	CFS	2973.0	151.9	76.67		+
	Two-stage IGWO	288.1	189	**82.50**	**78.89 ± 2.75**	
Leukemia 2	LFS	**143.4**	**4.7**	89.44		+
	CFS	5653.0	129.5	94.44		+
	Two-stage IGWO	330.5	45.6	**97.22**	**94.91 ± 1.80**	
Brain Tumor 2	LFS	**113.9**	**9.1**	77.50		=
	CFS	3182.2	101.1	77.50		=
	two-stage IGWO	345.4	182.8	**79.17**	74.03 ± 3.65	
Prostate	LFS	**158.2**	**5.9**	90.17		+
	CFS	2537.4	80.4	92.17		=
	Two-stage IGWO	350.4	41.8	**94.33**	**92.18 ± 2.03**	
Lung cancer	LFS	**358.8**	**8.5**	79.62		+
	CFS	85,179.1	517.0	93.76		+
	Two-stage IGWO	401.8	95.4	**98.29**	**95.64 ± 1.43**	
11 tumor	LFS	**309.3**	**17.3**	61.76		+
	CFS	57,340.7	361.6	80.04		+
	Two-stage IGWO	402.7	236.7	**93.05**	**90.59 ± 1.98**	

Experimental setting

In the two-stage IGWO algorithm, the importance ranking of all features is obtained using the MLP network, and the search space in the first stage is restricted to the inside of the top U most important features. Through experimental analysis, the algorithm can obtain good performance when U is set to 1000. In addition, the number of populations does not need to be too large because the search range of features is limited. The number of populations is set to 50, and the maximum number of iterations of the algorithm is 100.

w is set to 0.99 because the classification performance of the classifier is more important than the sparsity of the features. When training the MLP network, the training epoch of the MLP network is set to 150 to ensure that the weights of the MLP network reach convergence. For the regularization parameter λ, the choice is made from four typical values $\{1, 0.1, 0.01, 0.001\}$, and the best algorithm performance is achieved when λ is 0.1. The parameter settings used by the algorithm in this paper are shown in (Table 2).

Comparison algorithm

To compare with the feature selection algorithm proposed in this paper, four existing feature selection algorithms were selected. These previously published algorithms include linear forward selection (LFS) [36], correlation-based FS (CFS) [37], PSO [18], and CSO [20]. Among them, LFS and CFS are two traditional feature selection algorithms. PSO and CSO are two evolutionary computation-based feature selection algorithms. All methods operate in the same experimental environment.

Results and discussion

In this section, the performance of the proposed algorithm is tested on ten gene expression datasets to evaluate the effectiveness of our proposed algorithm. Table 3 shows the average experimental results for the original feature set (Full), PSO, CSO and the proposed algorithm over 30 experiments. Table 4 shows the average experimental results of LFS, CFS and the proposed algorithm over 30 experiments. The experimental data include the running time of the algorithms, the feature size, and the best and average accuracy on the test set. Note that the accuracy in the table refers to the balanced accuracy calculated using Eq. (21). The running time in Table 3 is in minutes and in Table 4 the running time is in seconds. In Tables 3 and 4, the minimum time consumption, minimum feature size, highest best and average accuracy obtained on each dataset are highlighted in bold. The last column in the table shows the results of the Wilcoxon statistical test at the 5% significance level. The results of the statistical tests are indicated by each of the three symbols "+", "=", and "−". The symbols "+", "=", and "−" indicate that the proposed method is significantly better, similar, and worse than the method being compared, respectively.

Comparisons with other evolutionary algorithms

To evaluate the performance of the proposed two-stage IGWO feature selection algorithm, an algorithm comparison is performed using the full set of features and two evolutionary algorithm-based feature selection algorithms.

Two-stage improved Gray Wolf optimization versus full

As shown in Table 3, the accuracy of the proposed algorithm on all datasets is higher than the accuracy of the model constructed using all features, which indicates the effectiveness of the feature selection process in the gene expression dataset. Specifically, in terms of the best accuracy, the proposed algorithm improved the best accuracy by more than 9.00% on all datasets. The largest improvement in best precision was found on the 9Tumor dataset, where the best precision was improved by 26.66%. In terms of average accuracy, the proposed feature selection algorithm improves the average accuracy by more than 5.47% on all the datasets. The largest improvement in average accuracy is on the 9Tumor dataset, with an average accuracy improvement of 23.61%. In addition, for the size of feature subsets, the proposed algorithm eliminates more than 95.7% of the features in all datasets. The largest percentage of feature reduction is on the Prostate dataset, which reduces about 99.6% of the features.

The experimental results show that the proposed algorithm can effectively reduce the size of features while improving the accuracy of classification. This indicates that the proposed two-stage search strategy is effective. In addition, the running time of the algorithm does not increase significantly with the increase of the problem dimension, mainly because the proposed rapid evaluation method effectively reduces the evaluation cost of the solution set.

Two-stage improved Gray Wolf optimization versus particle swarm optimization

As shown in Table 3, the proposed algorithm significantly outperforms the PSO algorithm in terms of classification accuracy and the number of selected features on all datasets as well as the running time of the algorithm. Specifically, compared to the PSO algorithm, the proposed algorithm improves the best accuracy by more than 5.00% on all datasets in terms of the best accuracy. The largest best accuracy improvement is in the 9Tumor dataset, where the best accuracy is improved by 18.33%. In terms of average accuracy, the proposed algorithm improves the average accuracy by more than 5.08% on all the datasets. The largest improvement in average accuracy is on the 11Tumor dataset, where the average accuracy is improved by 18.78%. For the size of feature subsets, the number of feature selections obtained by the PSO algorithm on all datasets is 15–124 times the number of features retained by the proposed algorithm, which is much worse than the algorithm in this paper.

In PSO algorithm, the feature selection problem is directly modeled as a high-dimensional combinatorial optimization problem. Due to the high dimensionality of the problem, the algorithm can easily fall into local optimal solutions. This causes the PSO algorithm to perform worse than the algorithm in this paper. For the running time of the algorithm, the running time of the PSO algorithm is 1.8 times to 85 times longer than that of the proposed algorithm on all datasets. The reason for the enormous running time in the PSO algorithm is that the classifier needs to be reconstructed for each solution to be evaluated. The expensive computational cost limits the application of the PSO algorithm to large-scale feature selection problems.

Two-stage improved Gray Wolf optimization versus competitive swarm optimization

As shown in Table 3, the best classification accuracy of the proposed algorithm is equal to or better than that of the CSO algorithm on three-tenths of the datasets. The average accuracy on seven-tenths of the datasets is better than that of CSO, with the largest improvement in the average accuracy on the 11Tumor dataset, where the average accuracy is improved by 11.07%. For the size of feature subsets, the proposed algorithm selects fewer features on seven-tenths of the datasets. On the Prostate dataset, the CSO algorithm obtains eight times the number of feature selections than the number of features retained by the proposed algorithm. For the running time of the algorithm, the CSO algorithm runs between 4 and 938 times longer than the proposed algorithm on all datasets. Although the CSO algorithm has good performance in terms of best accuracy, the expensive computational cost limits the application of the CSO algorithm to large-scale feature selection problems.

From the above discussion, it can be seen that the proposed algorithm has better performance in most cases compared to Full, PSO, and CSO. In 30 comparisons of average classification accuracy, the proposed algorithm won 25 times, tied 4 times, and lost 1 time. This demonstrates the superiority of the two-stage search strategy designed in this paper. Initial filtering of features is performed by modeling the optimization problem as an integer optimization problem using the importance ranking of features in the first stage. This process greatly reduces the search space of the large-scale feature selection problem. By limiting the search space to promising regions, it makes it easier for the IGWO algorithm to converge in the second stage. Thus, the proposed algorithm has this better performance.

The running time of the algorithm is a key factor limiting the application of evolutionary algorithm-based feature selection methods to large-scale feature selection problems. In terms of the running time of the algorithm, with the help of the fast evaluation strategy for fitness values proposed in this paper, it has an overwhelming advantage over other algorithms in terms of running time. On one hand, this is because different solutions use the same MLP for fitness evaluation, the only difference being that different solutions correspond to different MLP weights. The weights of the trained MLP are very inexpensive to modify. Since our method does not require training the classifier from scratch, this fast evaluation method greatly reduces the time consumption of the fitness computation. On the other hand, the inference of the neural network can be easily accelerated with the help of GPUs, which makes the computation of fitness even cheaper. The running time of the PSO algorithm is 1.8 times to 85 times longer than the proposed algorithm on all datasets, and the running time of the CSO algorithm is 4 times to 938 times longer than the proposed algorithm. This indicates that the comparison algorithms are all much worse than the proposed algorithms in terms of algorithm running efficiency. In addition, the advantages of the proposed algorithm in terms of time consumption, classification accuracy and feature subset size become more and more prominent as the dimensionality of the feature selection problem increases. This indicates that the proposed algorithm is particularly suitable for solving large-scale feature selection problems.

Comparisons with traditional methods

To evaluate the performance advantages and disadvantages between the proposed two-stage IGWO feature selection algorithm and the traditional feature selection algorithm, an algorithm comparison is performed using two traditional feature selection algorithms.

Two-stage improved Gray Wolf optimization versus linear forward selection

As shown in Table 4, the proposed algorithm outperforms the LFS algorithm in terms of the best accuracy on all datasets. The improved best accuracies are all above 1.67%. The largest improvement in the best accuracy is on the 9Tumor dataset, where the best accuracy is improved by 36.66%. In terms of average accuracy, the proposed algorithm achieves better accuracy than LFS on nine-tenths of the datasets. Only on the Brain2 dataset, the average accuracy of LFS is higher than the algorithm proposed by LFS. The LFS algorithm has an advantage in the running time of the algorithm, which is caused by the premature convergence of the LFS algorithm in the search process and thus falling into a local optimum solution.

Two-stage improved Gray Wolf optimization versus correlation-based FS

As shown in Table 4, the proposed algorithm outperforms the CFS algorithm in terms of classification accuracy on all datasets. The largest improvement in the best accuracy is on the 11Tumor dataset, where the best accuracy is improved by 13.01%. In terms of average accuracy, the proposed algorithm achieves classification accuracies equal to or better than LFS on eight tenths of the datasets. For the size of the feature subset, the proposed algorithm selects fewer features on seven tenths of the dataset, where the proposed algorithm retains only 1/5 times the number of features retained by the CFS algorithm on the Lung Cancer dataset. In terms of algorithm running time, the CFS algorithm has a slightly shorter running time than the proposed algorithm on the SRBCT dataset. The running time of the CFS algorithm is longer than that of the proposed algorithm on all the remaining datasets. It should be noted that the running time of the CFS algorithm increases substantially as the dimensionality of the problem increases. In the Lung Cancer dataset, its running time is 211 times longer than that of the proposed algorithm, which indicates that the proposed algorithm is more suitable for large-scale feature selection problems than CFS.

From the above discussion, it can be seen that the proposed algorithm wins 15 times and ties 5 times in the comparison of 20 average accuracies with 2 traditional methods. The results show that for most large-scale feature selection problems, the proposed algorithm is able to achieve a smaller subset of features and better classification accuracy with efficient running time compared to traditional methods.

The potential applications of two-stage improved Gray Wolf optimization

With a two-stage search strategy and a fast fitness evaluation mechanism, our proposed method is particularly suitable for solving large-scale feature selection problems. An example is data analysis of gene expression data. Gene expression data analysis is an important tool in cancer diagnosis. Although the dimensionality of genes is very high, only a small fraction of the dimensionality plays a role in classification. Therefore, searching for a subset of genes related to cancer among all characteristic genes is a key aspect in cancer gene research [38]. The algorithm proposed in this paper can effectively cope with this problem. Besides, many machine learning fields, such as text mining [1] and image retrieval [2], require data analysis of large amount of high-dimensional data, and the method proposed in this paper can also be used to solve such problems.

The limitation of two-stage improved Gray Wolf optimization

Although the proposed method is suitable for solving large-scale feature selection problems. However, the advantage of our algorithm gradually becomes smaller in small-scale feature selection problems. On one hand, it is because the search space of the small-scale feature selection problem is small, and it is not too difficult to model the problem directly as a discrete optimization problem for solving. On the other hand, it is because of the small size of the features, it is not too expensive to perform the fitness evaluation directly like using k-nearest neighbor [39], Support Vector Machine [40] and other classifiers. In addition, although the proposed algorithm improves the performance of the classifier, the existing experimental results are still far from the desired goal, especially for the feature selection problem dealing with class imbalance. However, this type of problem is also a major challenge to be solved in the field of feature selection.

Conclusion

In this paper, a two-stage IGWO algorithm is proposed to solve the feature selection problem for high-dimensional biological data. The proposed algorithm can significantly reduce the size of features while maintaining high performance metrics. Meanwhile, to solve the problem of high computational cost of wrapper-based feature selection algorithm, a rapid evaluation method is proposed using group lasso technique and MLP network. This method approximates the feature selection process by modifying the weight data of the MLP network, thus directly evaluating individuals on the modified neural network without retraining and greatly reducing the evaluation cost. In the fitness design, sparsity and classification accuracy are aggregated as a single objective to serve as an individual fitness evaluation. Then modeling the problem as a multi-objective problem is a worthy consideration. In addition to feature selection, group lasso regularization and IGWO based algorithms can be used for neural network sparse structure learning. Therefore, using IGWO algorithm to learn optimal neural network structure, which is the direction of our further research.

Funding Not applicable.

Availability of data and material The datasets used in this article can be publicly downloaded on the http://www.gemssystem.org.

Declarations

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

References

1. Bai X, Gao X, Xue B (2018) Particle swarm optimization based two-stage feature selection in text mining. In: 2018 IEEE congress on evolutionary computation (CEC), pp 1–8
2. Bermingham ML, Pong-Wong R, Spiliopoulou A et al (2015) Application of high-dimensional feature selection: evaluation for genomic prediction in man. Sci Rep 5:10312. https://doi.org/10.1038/srep10312
3. Rui Y, Huang T, Chang S (1999) Image retrieval: current techniques, promising directions, and open issues. J Vis Commun Image Represent 10:39–62
4. Egea S, Rego Mañez A, Carro B et al (2018) Intelligent IoT traffic classification using novel search strategy for fast-based-correlation feature selection in industrial environments. IEEE Internet Things J 5:1616–1624. https://doi.org/10.1109/JIOT.2017.2787959
5. Dash M (1997) Feature selection via set cover. In: Proceedings 1997 IEEE knowledge and data engineering exchange workshop, pp 165–171
6. Yang F, Mao KZ (2011) Robust feature selection for microarray data based on multicriterion fusion. IEEE/ACM Trans Comput Biol Bioinform 8:1080–1092. https://doi.org/10.1109/TCBB.2010.103
7. Guyon I, Elisseeff A (2003) An introduction to variable and feature selection. J Mach Learn Res 3:1157–1182
8. Molina LC, Belanche L, Nebot A (2002) Feature selection algorithms: a survey and experimental evaluation. In: 2002 IEEE international conference on data mining, 2002. Proceedings, pp 306–313
9. Liu H, Lei Yu (2005) Toward integrating feature selection algorithms for classification and clustering. IEEE Trans Knowl Data Eng 17:491–502. https://doi.org/10.1109/TKDE.2005.66
10. Loughrey J, Cunningham P (2005) Overfitting in wrapper-based feature subset selection: the harder you try the worse it gets. In: Bramer M, Coenen F, Allen T (eds) Research and development in intelligent systems XXI. Springer, London, pp 33–43
11. Jakulin A, Bratko I (2004) Testing the significance of attribute interactions. In: Proceedings of the twenty-first international conference on machine learning. association for computing machinery, New York, NY, USA, p 52
12. Mirjalili S, Mirjalili SM, Lewis A (2014) Grey Wolf optimizer. Adv Eng Softw 69:46–61. https://doi.org/10.1016/j.advengsoft.2013.12.007
13. Wong LI, Sulaiman MH, Mohamed MR, Hong MS (2014) Grey Wolf optimizer for solving economic dispatch problems. In: 2014 IEEE international conference on power and energy (PECon). pp 150–154
14. Tsai P-W, Nguyen T-T, Dao T-K (2017) Robot path planning optimization based on multiobjective Grey Wolf optimizer. In: Wang C-H, Jiang XH, Pan J-S, Lin JC-W (eds) Genetic and evolutionary computing. Springer International Publishing, Cham, pp 166–173
15. Lu C, Gao L, Li X, Xiao S (2017) A hybrid multi-objective grey wolf optimizer for dynamic scheduling in a real-world welding industry. Eng Appl Artif Intell 57:61–79. https://doi.org/10.1016/j.engappai.2016.10.013
16. Emary E, Zawbaa HM, Hassanien AE (2016) Binary grey wolf optimization approaches for feature selection. Neurocomputing 172:371–381. https://doi.org/10.1016/j.neucom.2015.06.083
17. Zhou J, Zhu W, Zheng Y, Li C (2016) Precise equivalent model of small hydro generator cluster and its parameter identification using improved Grey Wolf optimiser. IET Gener Transm Distrib 10:2108–2117. https://doi.org/10.1049/iet-gtd.2015.1141
18. J. Kennedy, R. C. Eberhart (1997) A discrete binary version of the particle swarm algorithm. In: 1997 IEEE International conference on systems, man, and cybernetics. computational cybernetics and simulation, vol 5, pp 4104–4108
19. Cheng R, Jin Y (2015) A competitive swarm optimizer for large scale optimization. IEEE Trans Cybern 45:191–204. https://doi.org/10.1109/TCYB.2014.2322602
20. Gu S, Cheng R, Jin Y (2018) Feature selection for high-dimensional classification using a competitive swarm optimizer. Soft Comput 22:811–822. https://doi.org/10.1007/s00500-016-2385-6
21. Wang X, Wang J, Zhang K et al (2020) Convergence and objective functions of noise-injected multilayer perceptrons with hidden multipliers. Neurocomputing. https://doi.org/10.1016/j.neucom.2020.03.119
22. Agarap AF (2018) Deep learning using rectified linear units (relu). arXiv:1803.08375
23. Scardapane S, Comminiello D, Hussain A, Uncini A (2017) Group sparse regularization for deep neural networks. Neurocomputing 241:81–89. https://doi.org/10.1016/j.neucom.2017.02.029
24. Wang J, Xu C, Yang X, Zurada JM (2018) A novel Pruning algorithm for smoothing feedforward neural networks based on group Lasso method. IEEE Trans Neural Netw Learn Syst 29:2012–2024. https://doi.org/10.1109/TNNLS.2017.2748585
25. Hong J-H, Cho S-B (2006) Efficient huge-scale feature selection with speciated genetic algorithm. Pattern Recognit Lett 27:143–150. https://doi.org/10.1016/j.patrec.2005.07.009
26. Ding Y, Zhou K, Bi W (2020) Feature selection based on hybridization of genetic algorithm and competitive swarm optimizer. Soft Comput 24:11663–11672. https://doi.org/10.1007/s00500-019-04628-6
27. Amini F, Hu G (2021) A two-layer feature selection method using Genetic Algorithm and Elastic Net. Expert Syst Appl 166:114072. https://doi.org/10.1016/j.eswa.2020.114072
28. Xue Y, Xue B, Zhang M (2019) Self-adaptive particle swarm optimization for large-scale feature selection in classification. ACM Trans Knowl Discov Data. https://doi.org/10.1145/3340848
29. Tran B, Xue B, Zhang M (2019) Adaptive multi-subswarm optimisation for feature selection on high-dimensional classification. In: Proceedings of the genetic and evolutionary computation conference. association for computing machinery, New York, NY, USA, pp 481–489
30. Xue Y, Tang T, Pang W, Liu AX (2020) Self-adaptive parameter and strategy based particle swarm optimization for large-scale feature selection problems with multiple classifiers. Appl Soft Comput 88:106031. https://doi.org/10.1016/j.asoc.2019.106031

31. Too J, Abdullah AR (2020) Opposition based competitive grey wolf optimizer for EMG feature selection. Evol Intell. https://doi.org/10.1007/s12065-020-00441-5
32. Hu P, Pan J-S, Chu S-C (2020) Improved binary Grey Wolf optimizer and Its application for feature selection. Knowl-Based Syst 195:105746. https://doi.org/10.1016/j.knosys.2020.105746
33. Chantar H, Mafarja M, Alsawalqah H et al (2020) Feature selection using binary grey wolf optimizer with elite-based crossover for Arabic text classification. Neural Comput Appl 32:12201–12220. https://doi.org/10.1007/s00521-019-04368-6
34. Al-Betar MA, Awadallah MA, Faris H et al (2018) Natural selection methods for Grey Wolf Optimizer. Expert Syst Appl 113:481–498. https://doi.org/10.1016/j.eswa.2018.07.022
35. Cheng R, Jin Y (2015) A social learning particle swarm optimization algorithm for scalable optimization. Inf Sci 291:43–60. https://doi.org/10.1016/j.ins.2014.08.039
36. M. Gutlein, E. Frank, M. Hall, A. Karwath (2009) Large-scale attribute selection using wrappers. In: 2009 IEEE symposium on computational intelligence and data mining, pp 332–339
37. Hall MA (2000) Correlation-based feature selection for discrete and numeric class machine learning. In: Proceedings of the seventeenth international conference on machine learning. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp 359–366
38. Li J, Wong L (2002) Identifying good diagnostic gene groups from gene expression profiles using the concept of emerging patterns. Bioinformatics 18:725–734. https://doi.org/10.1093/bioinformatics/18.5.725
39. Chen K, Zhou F-Y, Yuan X-F (2019) Hybrid particle swarm optimization with spiral-shaped mechanism for feature selection. Expert Syst Appl 128:140–156. https://doi.org/10.1016/j.eswa.2019.03.039
40. El-Kenawy E-SM, Ibrahim A, Mirjalili S et al (2020) Novel feature selection and voting classifier algorithms for COVID-19 classification in CT images. IEEE Access 8:179317–179335. https://doi.org/10.1109/ACCESS.2020.3028012