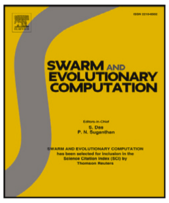
Contents lists available at ScienceDirect

Swarm and Evolutionary Computation

journal homepage: www.elsevier.com/locate/swevo



A novel importance-guided particle swarm optimization based on MLP for solving large-scale feature selection problems

Yu Xue [*], Chenyi Zhang

School of Software, Nanjing University of Information Science and Technology, Nanjing, 210000, China

ARTICLE INFO

Keywords:
Feature selection
Particle swarm optimization (PSO)
Neural network
Attention mechanism

ABSTRACT

Feature selection is a crucial data preprocessing technique that effectively reduces the dataset size and enhances the performance of machine learning models. Evolutionary computation (EC) based feature selection has become one of the most important parts of feature selection methods. However, the performance of existing EC methods significantly decrease when dealing with datasets with thousands of dimensions. To address this issue, this paper proposes a novel method called importance-guided particle swarm optimization based on MLP (IGPSO) for feature selection. IGPSO utilizes a two stage trained neural network to learn a feature importance vector, which is then used as a guiding factor for population initialization and evolution. In the two stage of learning, the positive samples are used to learn the importance of useful features while the negative samples are used to identify the invalid features. Then the importance vector is generated combining the two category information. Finally, it is used to replace the acceleration factors and inertia weight in original binary PSO, which makes the individual acceleration factor and social acceleration factor are positively correlated with the importance values, while the inertia weight is negatively correlated with the importance value. Further more, IGPSO uses the flip probability to update the individuals. Experimental results on 24 datasets demonstrate that compared to other state-of-the-art algorithms, IGPSO can significantly reduce the number of features while maintaining satisfactory classification accuracy, thus achieving high-quality feature selection effects. In particular, compared with other state-of-the-art algorithms, there is an average reduction of 0.1 in the fitness value and an average increase of 6.7% in classification accuracy on large-scale datasets.

1. Introduction

The advancement of information collection technology has led to an increase in the number of features in classification tasks, resulting in a large amount of redundant and irrelevant features within high-dimensional datasets [1]. These extraneous features can negatively impact the classification accuracy. Feature selection is a popular and efficient data preprocessing technology, which can select highly distinguishable features from original features, so as to reduce irrelevant features and redundant features. The benefits of feature selection include reducing the training time and storage space, and improving the generalization ability of the classification models [2].

Although feature selection has been a popular research topic for decades, finding optimal subsets becomes increasingly difficult for the datasets with high dimensions [3,4]. In recent decades, numerous researchers have proposed various approaches to address this issue. Traditional search methods such as sequential forward or backward floating selection cannot find satisfactory subsets, thus, researchers have turned towards evolutionary computation (EC) [5]. EC methods are not restricted by search space nor require auxiliary information. More importantly, they have superior global search capabilities that are lacking in traditional methods, making them advantageous for feature selection tasks [6].

Particle swarm optimization (PSO) is one of the EC methods and has been widely used in feature selection [7–9]. But EC based feature selection methods are suitable to be used for solving small-scale feature selection problems with tens to hundreds of dimensions [10,11]. When confronted with large-scale datasets containing thousands of dimensions, these algorithms become time consuming and cannot obtain satisfactory performance. Furthermore, the effectiveness of swarm intelligence-based algorithms is heavily influenced by population initialization [12]. Thus, Xue et al. utilized the ReliefF for population initialization in NSGA-II, resulting in a significant improvement in algorithm performance compared to the compared algorithm without special population initialization [13]. The ablation experiment conducted in [13] demonstrates that an excellent initialized population can expedite convergence and enhance the ability to identify the optimal

* Corresponding author.
E-mail addresses: xueyu@nuist.edu.cn (Y. Xue), 202212490748@nuist.edu.cn (C. Zhang).

https://doi.org/10.1016/j.swevo.2024.101760
Received 29 April 2024; Received in revised form 18 August 2024; Accepted 15 October 2024
Available online 4 November 2024

subset. Similar conclusions were also obtained in [14]. Nevertheless, existing methods primarily rely on simplistic mathematical analysis to guide population initialization, which may be effective on small-scale problems. Therefore, a new population initialization method is required for solving large-scale feature selection problems.

The balance of the exploitation and exploration is important for EC based feature selection methods [15–18]. Algorithms focusing solely on exploitation tend to overlook numerous potential areas and become trapped in local optima solutions, while some other algorithms emphasizing exploration discover multiple sub-optimal solutions without effectively identifying the optimal subset. Some algorithms adjusted the balance between the exploration and exploitation based on the frequency of position flipping [19–24]. However, in many feature selection tasks, not all features are equally important, and performing the same operation on all positions leads to a waste of computational resources, while slowing down the training process of the model.

Based on the above issues, this paper proposes a novel particle swarm optimization algorithm called IGPSO, which utilizes the neural networks to facilitate population initialization and coordinate population exploitation and exploration. The proposed approach shows greater potential in identifying superior feature subsets, which can achieve higher classification accuracy with fewer features. Specifically, the contributions of this study are as follows:

1. A novel neural network is introduced for importance vector generation. These importance vectors include the significance level of each feature and are employed for both population initialization and population evolution.
2. A two stage training method utilizing positive and negative examples is proposed. By separately training with normal samples and disordered samples using positive and negative sample sets, respectively, useful information related to features is extracted while disregarding irrelevant information.
3. An importance-guided population updating strategy is proposed, which uses the importance value to replace the acceleration factors and inertia weight in the traditional PSO algorithm. Specifically, the acceleration factor is positively correlated with the importance value, while the inertia weight is negatively correlated with the importance value. This allows for differential guidance of features based on their respective levels of importance, thereby regulating both population exploitation and exploration processes more effectively.

The remainder of this paper is organized as follows: Section 2 reviews the related work about FS methods and particle swarm optimization. Section 3 describes in detail our proposed methodology. Section 4 presents the experimental design and parameters. Section 5 reports the experimental results and analysis. Finally, Section 6 provides the conclusions of this work and suggests future directions for research.

2. Related work

In this section, we provide a brief literature review of FS methods and the PSO algorithm.

2.1. Feature selection

Feature selection plays an important role in many practical applications, Yin et al. proposed a robust multilabel feature selection method that leverages graph structure and fuzzy rough sets to consider feature interactions and dependencies, achieving superior performance and robustness [25]. Similarly, Yin et al. [26] introduced a robust MFS algorithm RMSMC considering feature multi-correlations. With the help of the fuzzy granulation mechanism, the uncertainty and ambiguity of multilabel data are characterized by fusing multiple fuzzy β coverings.

Among the typical feature selection methods, there are three main categories: filter method, wrapper method, and embedded method [27]. The filter method typically utilizes statistical measures such as correlation coefficient, mutual information, and information gain to evaluate the relationship between features [28]. Well-known filter methods include ReliefF [29] and mRMR [30]. On the other hand, the wrapper method uses a learning method to assess the selected feature subset, with higher computational cost but often achieving better performance compared to the filter method [31]. Lastly, the embedded method combines feature selection with machine learning methods to achieve optimal performance. Once the learning process is complete, the features used in the classifier are selected [13,32].

Due to global search capability and the do not require of any prior knowledge, EC methods are popular among feature selection methods [33,34]. Population initialization of the EC methods has a significant impact on the results. To address this issue, Li et al. used a feature weighting based on mutual information for population initialization, called feature weighting directed initialization [35]. Similarly, Xue et al. used the results of ReliefF as a basis for population initialization [13]. The results show that it is important to consider the initialization of the population as a crucial factor.

2.2. Particle swarm optimization

Evolutionary computation has attracted the attention of a wide range of researchers due to its powerful search capabilities [36]. There is a swarm intelligence algorithm called PSO, which has been paid attention by researchers and has been applied to many practical problems because of its simplicity, efficiency and effectiveness.

Inspired by the behavior of birds, particle swarm models have been proposed to solve continuous optimization problems. In the PSO algorithm, a group of particles represents candidate solutions, and the optimal solution is determined through the population update. Similar to other EC algorithms, PSO employs a fitness function to evaluate the quality of each particle. In addition, the optimal position of each particle itself and its neighbors are recorded as pbest (pb) and gbest (gb), respectively, which guide the particle on a directional exploration. Specifically, for a particle, the update formula of i-dimensional for velocity is as follows:

$$v_i^{t+1} = w \times v_i^t + c_1 \times r_1 \times (pb_i - x_i^t) + c_2 \times r_2 \times (gb_i - x_i^t) \tag{1}$$

where t represents the tth iteration in the population iteration process, x represents the position of the particle, w denotes the inertia weight, c_1 and c_2 are two hyperparameters representing the acceleration factor, and r_1 and r_2 are two random numbers between 0 and 1. Eq. (2) is used to update position x.

$$x_i^{t+1} = x_i^t + v_i^{t+1} \tag{2}$$

The original PSO algorithm is designed for solving continuous optimization problems but does not perform well when applied to discrete optimization problems such as feature selection. Feature selection involves denoting selection as 1 and discarding as 0. Kennedy et al. first introduced the concept of binary PSO to address this issue. Cervante et al., on the other hand, combined BPSO with information theory methods to tackle feature selection problems effectively. The main distinction between BPSO and original PSO lies in their update equation. In BPSO, Eq. (3) is used.

$$x_i^{t+1} = \begin{cases} 1, \text{if } rand() \le s(v_i^{t+1}) \\ 0, \text{otherwise} \end{cases} \tag{3}$$

where $s()$ means the sigmoid function. Since then, BPSO has been widely adopted in the field of feature selection. In the original BPSO algorithm, the sigmoid function was commonly used as a transfer function, also known as the S-shape function. Mirjalili et al. initially proposed the use of the V-shape function as a transfer function, and experimental results demonstrated that it could enhance the performance of BPSO [37]. However, due to the absence of ablation experiments, it remains uncertain which specific component is responsible for this improved performance. Drawing inspiration from quantum computing, Zheng et al. introduced quantum computing into the BPSO algorithm for the first time [38].

2.3. PSO-based feature selection

Nguyen et al. introduced Stick BPSO by reformulating momentum as stickiness and velocity as flipping probability [19]. The stickiness factor considers flipped particle states; particles that have not flipped for an extended period are more likely to flip, thus enhancing population activity. However, solely considering unconstrained flipping in time dimensions will expand potential subset space and increase computational costs while hindering population convergence. Chen et al. employ PSO with an evolutionary multitasking paradigm to solve feature selection problems [39]. The first task involves selecting from all the original features, while the second task involves selecting only from the top-ranked features. Zhang et al. applied BPSO to address spam detection issues and mitigate population prematurity through mutation operators, thereby enhancing algorithm performance [40]. Experimental data indicated superior performance compared to other compared algorithms. Tran et al. proposed VLPSO, which has a variable and dynamic length on high dimension features [41]. The results shown that VLPSO can achieve better classification accuracy in a shorter time. In addition, the competitive swarm optimization (CSO) algorithm is proposed in order to explore novel potential algorithms [42]. Nguyen et al. used performance constraints combined with Relief to improve the population diversity and local search efficiency of CSO [43]. Besides, SVM was used as a surrogate model to accelerate the evaluation process. The results shown that the introduction of an adaptive performance constraint can force particles to learn from other high-quality particles, thus guiding the population towards a more promising search area. However, this step makes the population more inclined to exploit, thus the particles are more likely to fall into local optimality. From the above review, it can be seen that existing PSO based feature selection algorithms often rely on mathematical methods or pure evolutionary computation methods to guide population initialization and update strategic, which leads to these algorithms being unable to lead satisfactory initialization when facing large-scale datasets and unable to have appropriate theories to guide population updates in this case. In this work, we proposed the focal neural network to guide population initialization and particles updating.

3. Proposed method

In this section, we first introduce the overall architecture of IGPSO and then present the details of each element composing the framework.

3.1. Overall framework

The overall feature selection framework is shown in Fig. 1. The framework consists of a two-stage trained neural network called focal neural network, and an importance-guided particle swarm optimization algorithm. The purpose of the focal neural network is to learn the importance of all features in the different two stages, the first stage is dedicated to finding effective features and removing invalid features, then the second stage is to identify redundant information. After that, the vector **ip** was generated, which can comprehensively reflect the importance of the features. The generation process of **ip** was described in Eq. (6). In the population updating, the importance-guided particle swarm optimization algorithm not only considers the relationship among the current position, the individual optimal solution and the global optimal solution, but also considers the importance of the features, so as to guide the population to find the optimal solution efficiently.

3.2. Focal neural network

3.2.1. Focal module

In a real-world dataset, there are many invalid features that are “unimportant” or even “harmful” to the machine learning methods. The role of the focal neural network is to focus on the “important” features, so that the machine learning methods can execute tasks faster and better by using the valid information in the dataset.

The focal module is composed of the following parts: a data normalization layer, an attention vector **A**, and a MLP with two hidden layers. At the beginning of training, every element in **A** is initialized to 1 to ensure that each feature is equal in the beginning. Assuming a_i is the ith element in **A**. For any a_i, the a_i^* is obtained by a sigmoid function as Eq. (4). After mapping each element of **A**, $\mathbf{A}^*$ is generated.

$$a_i^* = \frac{1}{(1+e^{-a_i})}. \tag{4}$$

After the normalization of the dataset, the dataset **X** is fed into the focal module for training. Each instance **x** can be seen as a vector:

$$\mathbf{x} = (x_1, x_2, \dots, x_D)$$

where each component represents a feature. D means the original number of features in the dataset.

First, the dataset **X** is multiplied bit by bit with the attention vector $\mathbf{A}^*$ to obtain vector $\mathbf{X}^\mathbf{A}$, which is used as the input of the neural network.

$$\mathbf{X}^\mathbf{A} = \mathbf{X} \odot \mathbf{A}^* \tag{5}$$

where “⊙” indicated the Hadamard product.

Then, in the process of the neural network training, the parameters in **A** are continuously updated through the backpropagation algorithm, so that **A** can finally learn the importance of different features.

Finally, the importance vector **ip** is obtained by mapping vector $\mathbf{A}^*$, and the data is scaled to the range of [0.1 0.9] by the function, which is shown in Eq. (6).

$$ip_i = \frac{a_i^* - a_{min}^*}{a_{max}^* - a_{min}^*} * 0.8 + 0.1 \tag{6}$$

where a_i^* represents the ith element in $\mathbf{A}^*$, and a_{min}^* and a_{max}^* mean the maximum and minimum values in $\mathbf{A}^*$, respectively. The reason for scaling the elements in the **ip** to [0.1 0.9] is that it makes it possible for each feature to be selected, which helps to give more diversity to the particle swarm optimization in the initialization stage and the population updating.

3.2.2. Neural network training

After normalizing the real-world dataset to the input dataset **X**, 70% of the samples are used as the training set, and the remaining of the samples are used as the validation set. The samples in the training set are further divided into positive and negative samples, where the positive samples account for k% of the training set, the remaining (1-k%) of the training set are used as the negative samples, and the labels of the negative samples are replaced by wrong labels. In this study, k is set to 70. Different choices of k are shown in ablation study 5.2.1.

In training, we find that the direct training has certain defects, the model is easy to overfit and the accuracy cannot satisfy us in some cases. We found that the use of negative samples for training can well solve the problem of overfitting and improve the accuracy. The reason for this may be that when only positive samples are used for training, the model will automatically learn useful information and ignore invalid information, but for redundant information, the model will still consider them helpful, and training with negative samples can help the model identify the redundant information.

Two focal modules named FM_{pos} and FM_{neg} are initialized before the training. Firstly, the positive sample set is utilized as the training set for FM_{pos}, after that, the output $\mathbf{A}^*_{\mathbf{pos}}$ of FM_{pos} is obtained.

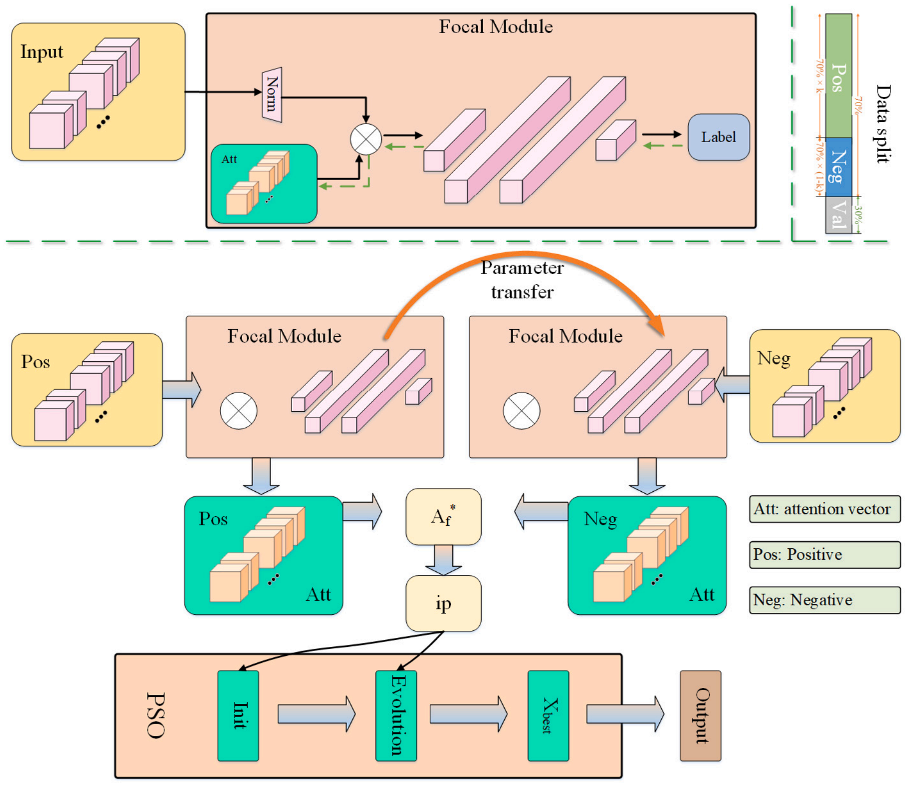


Fig. 1. Overall architecture of IGPSO.

The parameters of the multi-layer perceptron in FM_{neg} are inherited from the multi-layer perceptron in FM_{pos}, while vector $\mathbf{A}$ is reinitialized. Subsequently, FM_{neg} undergoes training on the negative sample set and obtains the vector $\mathbf{A}^*_{\mathbf{neg}}$. By subtracting $\mathbf{A}^*_{\mathbf{neg}}$ from $\mathbf{A}^*_{\mathbf{pos}}$, we obtain the final attention vector $\mathbf{A}^*_{\mathbf{f}}$, then by utilizing Eq. (6) the vector $\mathbf{ip}$ is generated, which is the important vector.

The pseudo-code for two-stage training is shown in Algorithm 1.

Algorithm 1 two-stage training.

```
Input: Dataset $\mathbf{X_{Pos}}$ and Dataset $\mathbf{X_{Neg}}$.
Output: Importance vector ip
1: Take $\mathbf{X_{Pos}}$ as the input for $FM_{pos}$ to generate $\mathbf{A}^*_{\mathbf{Pos}}$
2: Save parameters of $FM_{pos}$ as Para
3: $FM_{neg}$ loads Para
4: Every element of $\mathbf{A}$ in $FM_{neg}$ is initialized to 1
5: Take $\mathbf{X_{Neg}}$ as the input for $FM_{neg}$ to generate $\mathbf{A}^*_{\mathbf{neg}}$
6: Subtract $\mathbf{A}^*_{\mathbf{Neg}}$ from $\mathbf{A}^*_{\mathbf{Pos}}$ to get $\mathbf{A}^*_{\mathbf{f}}$
7: Use $\mathbf{A}^*_{\mathbf{f}}$ to calculate ip as Eq. (6)
8: Return ip
```

3.3. Importance-guided particle swarm optimization

In a standard binary PSO, the position of the next stage is not associated with the current position. Unlike the smooth movement in the original PSO, the particles change their current position by flipping the position from 0 to 1 or from 1 to 0. This kind of probabilistic binary flip is not effectively described by velocity, on the contrary, the use of flip probability is a potential solution. Therefore, in the proposed IGPSO, instead of using the velocity vector $\mathbf{v}$, we use the flip probability vector $\mathbf{P}$ to describe the possible changes in the particles, where the value of each entry represents the flip probability of the corresponding position. Prior to this, we obtained the vector $\mathbf{ip}$ that describes the importance of the feature. Based on $\mathbf{ip}$, the important features are more inclined to exploitation, while the unimportant features are more inclined to exploration.

The velocity vector of the particle swarm algorithm is mainly composed of three parts: momentum, cognitive, and social factors. All the three factors need to be modified for the binary PSO. We replace and modify momentum, cognitive, and social factors using the relevant factors of the importance vector $\mathbf{ip}$. In addition, we replaced the acceleration factors with the random numbers. Specifically, for a particle, the flipping probability of the i-dimension P_i is updated according to the following Eq. (7):

$$P_i = (1 - ip_i) \times Rand_i + \frac{ip_i}{2} \times |pb_i - x_i| + \frac{ip_i}{2} \times |gb_i - x_i| \tag{7}$$

where $Rand_i$ is a random value between 0 and 1, ip_i means the importance value of the i-dimension.

According to the P_i, the position of the i-th dimension should be updated according to Eq. (8).

$$x_i^{t+1} = \begin{cases} 1 - x_i^t & , Rand < P_i \\ x_i^t & , otherwise \end{cases} \tag{8}$$

where $Rand$ is a random number between 0 and 1. When P_i is greater than $Rand$, the position of i-th is flipped, otherwise it remains in the same position.

For a swarm intelligence algorithm, in addition to the population updating, the initialization of the population is also a crucial part. Therefore, in IGPSO, we design a novel population initialization strategy according to the importance of features.

$$x_n^m = \begin{cases} 1 & , ip_n > rand_{mn} \\ 0 & , otherwise \end{cases} \tag{9}$$

where x_n^m means the n-dimension of the m-th particle in the space, ip_n is the n-dimension of the importance vector ip, $rand_{mn}$ is a random number between 0 and 1, if ip_n is greater than $rand_{mn}$, the n-th dimension of the m-th particle is initialized to 1, otherwise set to 0.

The pseudo-code for IGPSO is shown in Algorithm 2.

Algorithm 2 importance-guided particle swarm optimization.

```
Input: Importance vectors ip
Output: The best particle position $x_{best}$
 1: Initialization:
 2: for each particle $x$ do
 3:   for $i$=1:$D$ do
 4:     /*D means the dimension of the original dataset.*/
 5:     $x_i = 0$
 6:     if $ip_i \geq Rand$ then
 7:       /*Rand means a random number between (0,1)*/
 8:       $x_i = 1$
 9:     end if
10:   end for
11:   Evaluate particle $x$ and set pbest=$pb_i$, gbest=$gb_i$
12: end for
13: Evaluation:
14: while not satisfy stop conditions do
15:   for each particle $x$ do
16:     for $i$=1:$D$ do
17:       Calculate $P_i$ as Eq. (7)
18:       Generate $Rand$
19:       if $P_i > Rand$ then
20:         $x_i$ = 1-$x_i$
21:       end if
22:     end for
23:     Evaluate particle $x$ and update $pb_i$ and $gb_i$
24:   end for
25: end while
26: Return $x_{best}$
```

3.4. Analysis of IGPSO

3.4.1. Population updating

According to the Eq. (7), the flip probability **P** is mainly influenced by the importance vector **ip** and a random number. In the process of evolution of PSO, according to the relationship among Gbest, Pbest and the current position, the situation can be divided into four categories, which are shown in Eq. (10).

$$P_i = \begin{cases} (1 - ip_i) \times Rand_i & , x_i = pb_i = gb_i \\ (1 - ip_i) \times Rand_i + \frac{ip_i}{2} & , x_i = pb_i \neq gb_i \\ (1 - ip_i) \times Rand_i + \frac{ip_i}{2} & , x_i = gb_i \neq pb_i \\ (1 - ip_i) \times Rand_i + ip_i & , x_i \neq pb_i = gb_i \end{cases} \tag{10}$$

Eq. (10) describes the flipping probability of a single bit in a particle during evolution. When the evolution begins, it is very likely that the situation will be $x_i \neq pb_i = gb_i$, and as the particles continue to evolve until the end of the evolution process, the situation is likely to become $x_i = pb_i = gb_i$. Without taking into account the possible disturbance caused by $Rand_i$, the P_i has a maximum value of $Rand_i + (1 - Rand_i) \times ip_i$ and a minimum value of $Rand_i - Rand_i \times ip_i$. Compared with the random number $Rand$ as Eq. (8), the former has a higher probability of flipping, while the latter tends to remain unchanged.

In addition, taking into account the importance value ip_i, it can be easily concluded that particles are more inclined to explore at the beginning of evolution and more inclined to exploit at the later stage of evolution. The advantage of this setup is that the particle swarm can quickly find the most beneficial area and start exploitation in that area. Thus, the algorithm has the ability to balance exploration and exploitation.

3.4.2. The analysis of the complexity

IGPSO consists of two main parts: the importance vector generation process and the particle swarm updating process. As for the particle swarm updating process, suppose iterations M, population size S, and input dimension D, we can deduce that the time complexity of population initialization is $O(DS)$. For a particle, in each iteration, the time complexity for updating flipping probability vector **P** is $O(D)$, determining whether flipping occurs is also O(D), then updating PB and GB costs $O(1)$. Thus, the time complexity for a particle in one iteration is $O(D)$. It can be directly inferred that the time complexity of S particles in an iteration is $O(DS)$, and considering the number of population iterations M, the time complexity of the particle swarm optimization algorithm is $O(MDS)$.

The time complexity of the importance vector generation process, which is primarily for MLP training. For a dataset with N samples and D-dimensional features, the time complexity is $O(ND^2)$. Since the importance vector generation process is implemented based on Pytorch, the time consuming is much less than the particle swarm updating process, so the complexity of IGPSO can be considered as $O(MDS)$, which is comparable to the time complexity of traditional PSO.

4. Experimental setup

The datasets utilized in the experiments are described in Table 1, each accompanied by a link to web pages dedicated to their specifics. Notably, these datasets originate from different real-world scenarios and with different sizes. Categorized based on the number of features, the datasets are divided into two groups: small-scale and large-scale. The majority of these datasets were initially included in the UCI Machine Learning Repository http://archive.ics.uci.edu/ml/index.php and the scikit-feature selection repository [35]. Each dataset was split into two partitions: the training set, which comprised 70% of the original dataset and was randomly selected, and the test set, which included the remaining examples.

The neural network parameters were updated using the SGD optimizer, with a momentum of 0.9 and a weight decay of 0.00001. During training, the learning rate was gradually reduced using the cosine annealing strategy, with a minimum value of 0.001, which were popular settings.

For all EC-based FS algorithms, the stopping criterion was set at 10,000 evaluations, denoted as ME (Max Evaluations) = 10,000. All other configurations were consistent with the settings in their original algorithms. For IGPSO, the population size is set to 100, which is same consistent with the settings in other related articles.

All algorithms were run independently 30 times. To enhance the statistical significance of the findings, we employed a T-test at a 95% confidence level.

We chose KNN (K = 3) as the classifier in the experiment. The reason for this is twofold. Firstly, the KNN algorithm lacks feature selection ability and exhibits weak classification performance, which allows the difference between feature subsets to be fairly revealed. Secondly, it is highly susceptible to the influence of invalid and redundant features. These characteristics make it an ideal choice for evaluating the feature selection capabilities of different algorithms.

Furthermore, to further illustrate IGPSO's performance, we introduced fitness values as a third evaluation criterion. Fitness values are calculated as Eq. (11):

$$Fitness = (1 - acc) \times 0.9 + \frac{sz}{SZ} \times 0.1 \tag{11}$$

where $1 - acc$ represents the error rate of the algorithm, sz represents the number of features contained in the subset, and SZ represents the number of features in the original dataset.

5. Results and analysis

5.1. Comparison with the state-of-the-art feature selection methods

In this section, we compare the performance of IGPSO with seven other state-of-the-art feature selection methods, including mRMR [30],

Table 1
Information of Datasets.

No.	Dataset	Classes	Features	Samples
01	Vehicle	4	18	846
02	WallRobot	4	24	5456
03	German	2	24	1000
04	Guester	5	32	9873
05	Ionosphere	2	34	351
06	Chess	2	36	3196
07	Movement	15	90	360
08	HillValley	2	100	606
09	MUSK1	2	166	476
10	USP	10	256	9298
11	Madelon	2	500	2000
12	Isolet5	26	617	1559
13	MFS	10	649	2000
14	Gametes	2	1000	1600
15	QAR	2	1024	1687
16	QOT	2	1024	8992
17	ORL	40	1024	400
18	COLL20	20	1024	1440
19	Bioresponse	2	1776	3751
20	RELATHE	2	4322	1427
21	BASEHOCK	2	4862	1993
22	Brain	5	5920	90
23	Prostate2	2	10509	102
24	11Tumor	11	12533	174

ReliefF [29], RFS [44], VLPSO [41], SBPSO [19], CNNIWPSO [45], EARFS [46], and CCSO [43]. Among them, VLPSO, SBPSO and CCSO are the outstanding PSO-based algorithms proposed in recent years, CNNIWPSO and EARFS combining deep learning method with feature selection, while mRMR, ReliefF and RFS are well-known non-EC feature selection methods. For the fairness of comparison, the settings of the PSO-based methods are given in Section 4, while for the non-EC method, we chose the same number of features as the one selected by IGPSO, since they could not determine the number of features.

5.1.1. Results on small-scale datasets

Table 2 shows the accuracy of the best subsets selected by IGPSO and the seven comparison algorithms for small-scale datasets, as well as the accuracy of original dataset evaluated by the KNN classifier. The symbols (+)/(-) in this table indicate that the result obtained by the IGPSO algorithm was significantly superior/inferior to that of the comparison algorithms, while (=) signifies no significant difference between the IGPSO algorithm and the comparison algorithms. The best mean value for each dataset is highlighted in bold. The W/D/L metric indicates the number of times IGPSO demonstrates significant superiority/similarity/inferiority compared to the compared algorithms, as determined by the T-test. The RANK metric represents the average ranking of performance over all datasets.

As can be seen from Table 2, compared with the comparison algorithms, IGPSO has achieved good performance on 11 small-scale datasets, among which IGPSO has achieved the best performance on 7 datasets (01, 05, 07–11). According to the results of the T-test, IGPSO shows a significant advantage in most cases, and for the original dataset and most compared algorithms, IGPSO achieves a significant advantage on more than 10 datasets. Compared to the powerful SBPSO and CCSO algorithms, IGPSO also achieved better performance on 7 datasets and 4 datasets, respectively. Overall, IGPSO outperforms other compared algorithms on small-scale datasets. In addition, this conclusion can also be intuitively reflected in the average ranking. The average ranking of the IGPSO algorithm on 11 small-scale datasets is 1.64, far exceeding the second-place CCSO algorithm, which fully demonstrates the stability of IGPSO in different scenarios.

Table 3 illustrates the ability of IGPSO and four EC-based compared algorithms to reduce features. As can be seen from Table 3, the six algorithms have different degrees of feature reduction ability, among which CCSO obtains 3 smallest datasets (05, 06 and 07), and IGPSO obtains 6 smallest datasets. However, the size of the subset is one of the important factors affecting the classification accuracy. Therefore, we introduce the fitness function Eq. (11) to comprehensively evaluate the feature selection ability of these algorithms.

Table 4 shows the fitness scores for the best subsets generated by the different algorithms. We can intuitively find that IGPSO achieved independent optimal fitness values on 5 datasets (01, 03–05, 11), and tied for the first place with CCSO on 4 datasets (02, 08–10), and achieved 9 optimal fitness values. This demonstrates IGPSO's excellent feature selection and stability. In addition, from the results of the T-test, IGPSO showed an overwhelming advantage compared to the most other algorithms. Besides, IGPSO showed significant differences on the most datasets. Compared to EARFS, IGPSO showed a significant advantage on 8 datasets, and there was no significant difference on 3 datasets. IGPSO is better than SBPSO on 5 datasets and no significant difference on 6 datasets. We can get the conclusion that the importance vector generated by the neural network has advantages over stickiness or mathematical analysis, which can better guide the particle to update. In addition, population initialization based on the importance of features may also be responsible for this phenomenon. Overall, IGPSO outperforms most compared algorithms on small-scale datasets. It is worth noting that attention vectors are often not sufficiently trained on small-scale datasets, so IGPSO cannot fully demonstrate its power on small-scale datasets.

5.1.2. Results on large-scale datasets

In this section, we analyze the performance of IGPSO versus other algorithms on large-scale datasets. Table 5 shows the performance of IGPSO and the compared algorithms, and we can easily see that IGPSO achieves the best classification accuracy on all 11 datasets. Compared to the results in Table 2, the gap between IGPSO and the other algorithms is large, and the advantages of IGPSO on large-scale datasets are more obvious. Besides, with the increase in the size of the datasets, the difference is also gradually increasing. In addition, from the analysis of the results of the T-test, IGPSO has achieved a significant advantage on more than 9 datasets. For the average ranking, IGPSO achieves a good score of 1.36, and the average ranking of CCSO, which ranks second, is 2.00.

Table 6 shows the number of features found by the compared algorithms and IGPSO. As can be seen from this table, IGPSO exhibits strong feature selection capabilities on large-scale datasets, with only about 20% of the features retained for most datasets (11–13, 16–17, 21) and achieves an increase in classification accuracy. For datasets 15, 16, 20, and 22, the proportion of the removed features is more than 85%, especially for dataset 20, the proportion is 93.4%. Even for the dataset with the highest proportion of retained features, IGPSO removed 72.5% of features and achieved an accuracy improvement of about 10%. As a result, IGPSO performs better than the other algorithms on large-scale datasets.

Table 7 shows the performance of the other algorithms and IGPSO about fitness values. It can be seen that IGPSO achieves the best fitness value on 11 datasets. For the ReliefF and SBPSO algorithms, IGPSO obtains a significant advantage on all 13 datasets, while for mRMR, VLPSO, CNNIWPSO and CCSO, the number is 12. In addition, the average ranking of IGPSO is 1.15, which is an exciting result, while the second-placed EARFS achieves an average ranking of 2.85. The reason for this phenomenon may be that the neural networks are fully trained by the large-scale datasets, and the importance vector efficiently reflects the relationship between features, so that IGPSO can determine the region where the best subset is located and further mine the potential feature set. However, when other algorithms meet with large-scale datasets, due to the increasing number of features and the exponential increase in feature combinations, the algorithm cannot find the region where the optimal subset is located within a certain number of iterations, thus unable to achieve a balance between exploration and exploitation.

Table 2
Accuracy of the best subset selected by IGPSO and the seven compared algorithms for small-scale datasets (%).

No.	Dataset	FULL	mRMR	ReliefF	RFS	VLPSO	SBPSO	CNNIWPSO	EARFS	CCSO	IGPSO
01	Vehicle	68.42(+)	61.78(+)	68.01(+)	66.56(+)	70.19(+)	71.14(+)	70.17(+)	69.53(+)	71.41(+)	**72.27**
02	WallRobot	84.89(+)	86.49(+)	82.22(+)	88.81(+)	92.82(+)	**93.91**(=)	92.58(+)	92.71(+)	93.91(=)	93.89
03	German	62.94(=)	56.83(+)	58.73(+)	59.84(+)	62.18(+)	**63.11**(=)	59.67(+)	62.46(+)	62.25(=)	62.96
04	Guester	53.12(+)	37.41(+)	51.18(+)	47.87(+)	55.19(+)	57.17(=)	54.34(+)	57.19(=)	**57.94**(−)	57.22
05	Ionosphere	82.74(+)	89.32(+)	76.01(+)	82.00(+)	87.53(+)	87.99(+)	82.42(+)	91.55(+)	90.09(+)	**91.69**
06	Chess	93.86(+)	93.72(+)	96.73(=)	90.43(+)	91.05(+)	94.84(+)	94.91(+)	95.81(+)	**97.27**(−)	96.57
07	Movement	75.48(+)	70.76(+)	63.21(+)	80.12(+)	75.30(+)	75.80(+)	80.03(+)	82.50(=)	82.30(=)	**82.52**
08	Hillvalley	59.86(+)	55.49(+)	56.88(+)	58.74(+)	60.85(+)	61.47(+)	59.07(+)	52.38(+)	62.39(=)	**62.47**
09	Musk	85.23(+)	79.73(+)	82.86(+)	79.30(+)	81.75(+)	88.90(=)	83.87(+)	85.31(+)	**89.46**(=)	88.95
10	USP	96.49(=)	66.10(+)	94.09(+)	94.35(+)	96.32(=)	96.46(=)	96.29(+)	96.37(=)	96.36(=)	**96.70**
11	Madelon	56.79(+)	50.00(+)	71.15(+)	52.31(+)	87.80(+)	61.26(+)	75.19(+)	88.55(=)	76.76(+)	**88.82**
	W/D/L	9/2/0	11/0/0	10/1/0	11/0/0	10/1/0	7/4/0	11/0/0	7/4/0	3/6/2	N/A
	RANK	6.09	8.82	7.91	8.09	5.73	3.82	6.09	4.18	2.55	1.64

Table 3
Sizes of the best subset selected by IGPSO and the seven compared algorithms for small-scale datasets.

No.	Dataset	FULL	VLPSO	SBPSO	CNNIWPSO	EARFS	CCSO	IGPSO
01	Vehicle	18.0	9.2	9.0	9.3	9.3	9.3	**7.6**
02	WallRobot	24.0	5.8	4.0	5.6	4.1	4.0	**3.8**
03	German	24.0	12.4	11.1	10.2	7.3	6.4	**5.3**
04	Guester	32.0	16.9	13.0	14.0	15.5	13.0	**10.3**
05	Ionosphere	34.0	6.7	9.4	5.4	6.5	**5.2**	6.3
06	Chess	36.0	22.3	24.1	18.4	24.0	**11.3**	14.3
07	Movement	90.0	33.6	53.5	22.7	69.6	**22.2**	31.2
08	Hillvalley	100.0	47.9	33.5	45.9	69.4	23.1	**22.7**
09	Musk	160.0	**33.0**	68.3	39.4	69.7	52.6	50.6
10	USP	256.0	155.3	103.8	142.8	88.9	76.2	**70.3**
11	Madelon	500.0	**7.0**	228.0	16.7	11.0	60.3	10.1

Table 4
Fitness values of the best subset selected by IGPSO and the seven compared algorithms on small-scale datasets.

No.	Dataset	FULL	mRMR	ReliefF	RFS	VLPSO	SBPSO	CNNIWPSO	EARFS	CCSO	IGPSO
01	Vehicle	0.38(+)	0.39(+)	0.33(+)	0.34(+)	0.32(+)	0.31(=)	0.32(+)	0.33(+)	0.31(=)	**0.29**
02	WallRobot	0.24(+)	0.14(+)	0.18(+)	0.12(+)	0.09(+)	0.07(=)	0.09(+)	0.08(+)	**0.07**(=)	**0.07**
03	German	0.43(+)	0.41(+)	0.39(+)	0.38(=)	0.39(+)	0.38(=)	0.41(+)	0.37(=)	0.37(=)	**0.36**
04	Guester	0.52(+)	0.60(+)	0.47(+)	0.50(+)	0.46(+)	0.43(=)	0.45(+)	0.43(+)	0.42(=)	**0.41**
05	Ionosphere	0.26(+)	0.11(+)	0.23(+)	0.18(+)	0.13(+)	0.14(+)	0.17(+)	0.10(=)	0.10(=)	**0.09**
06	Chess	0.16(+)	0.10(+)	0.07(=)	0.13(+)	0.14(+)	0.11(+)	0.10(+)	0.10(+)	**0.06**(−)	0.07
07	Movement	0.32(+)	0.30(+)	0.37(+)	0.21(+)	0.26(+)	0.28(+)	0.20(+)	0.23(+)	**0.18**(=)	0.19
08	Hillvalley	0.46(+)	0.42(+)	0.41(+)	0.39(+)	0.40(+)	0.38(+)	0.41(+)	0.50(+)	**0.36**(=)	**0.36**
09	Musk	0.23(+)	0.21(+)	0.19(+)	0.22(+)	0.18(+)	0.14(=)	0.17(+)	0.18(+)	**0.13**(=)	**0.13**
10	USP	0.13(+)	0.33(+)	0.08(+)	0.08(+)	0.09(+)	0.07(=)	0.09(+)	0.07(+)	**0.06**(=)	**0.06**
11	Madelon	0.49(+)	0.45(+)	0.26(+)	0.43(+)	0.11(=)	0.39(+)	0.23(+)	0.11(=)	0.22(+)	**0.10**
	W/D/L	11/0/0	11/0/0	10/1/0	10/1/0	10/1/0	5/6/0	11/0/0	8/3/0	1/9/1	N/A
	RANK	9.55	8.00	6.91	6.73	5.73	4.45	5.64	4.55	2.00	1.36

Table 5
Accuracy of the best subset selected by IGPSO and the seven compared algorithms on large-scale datasets (%).

No.	Dataset	FULL	mRMR	ReliefF	RFS	VLPSO	SBPSO	CNNIWPSO	EARFS	CCSO	IGPSO
12	Isolet5	79.49(+)	80.77(+)	82.05(+)	87.18(+)	89.04(+)	80.49(+)	79.72(+)	89.61(=)	76.76(+)	**89.64**
13	MFS	97.00(+)	96.83(+)	97.00(+)	97.33(+)	96.75(+)	97.20(+)	97.07(+)	**98.65**(=)	97.20(+)	98.57
14	Gametes	51.25(=)	50.83(+)	48.12(+)	49.37(+)	50.10(+)	49.73(+)	50.50(+)	50.46(+)	50.52(+)	**51.50**
15	QAR	63.99(+)	61.16(+)	62.71(+)	65.55(+)	63.19(+)	66.04(+)	64.89(+)	83.69(+)	67.36(+)	**93.03**
16	QOT	72.41(+)	73.26(+)	74.57(+)	72.48(+)	73.47(+)	74.09(+)	74.79(+)	86.33(+)	76.00(+)	**92.55**
17	ORL	83.33(+)	85.00(+)	79.17(+)	84.17(+)	82.21(+)	82.38(+)	81.59(+)	85.40(+)	85.69(+)	**87.95**
18	COLL20	97.89(+)	95.78(+)	96.74(+)	98.58(=)	97.87(+)	98.00(=)	98.20(=)	98.21(+)	98.29(=)	**98.62**
19	Bioresponse	73.01(+)	74.29(+)	73.51(+)	72.84(+)	75.22(+)	73.95(+)	74.22(+)	79.74(+)	74.45(+)	**82.39**
20	RELATH	81.37(+)	78.89(+)	79.02(+)	78.93(+)	79.55(+)	82.58(+)	81.01(+)	89.72(=)	82.79(+)	**89.98**
21	BASEHOCK	80.76(+)	87.78(+)	82.13(+)	76.49(+)	89.57(+)	85.68(+)	87.60(+)	91.73(=)	87.64(+)	**92.02**
22	Brain	61.11(+)	46.47(+)	63.41(+)	70.56(=)	53.12(+)	64.81(+)	64.38(+)	66.71(+)	65.85(+)	**70.63**
23	Prostate2	83.57(+)	76.39(+)	85.46(+)	88.91(+)	87.60(+)	90.37(+)	91.16(+)	95.24(=)	94.33(+)	**95.56**
24	11Tumor	80.64(+)	79.35(+)	82.49(+)	83.31(+)	82.19(+)	85.56(+)	90.67(+)	93.43(+)	93.57(+)	**94.01**
	W/D/L	12/1/0	13/0/0	13/0/0	11/2/0	13/0/0	12/1/0	13/0/0	8/5/0	12/1/0	N/A
	RANK	7.08	7.69	7.77	6.15	6.77	5.77	5.77	2.69	3.92	1.23

Table 6
Sizes of subsets selected by IGPSO and the seven algorithms on large-scale datasets.

No.	Dataset	FULL	VLPSO	SBPSO	CNNIWPSO	EARFS	CCSO	IGPSO
12	Isolet5	617.0	166.9	371.6	167.1	145.7	190.7	**126.6**
13	MFS	649.0	**21.3**	268.7	121.8	151.0	170.4	133.7
14	Gametes	1000.0	305.0	489.0	298.9	301.1	263.9	**216.4**
15	QAR	1024.0	265.6	487.1	273.3	169.6	284.1	**140.2**
16	QOT	1024.0	339.8	477.7	336.4	219.5	321.9	**148.0**
17	ORL	1024.0	480.4	408.9	340.8	277.6	244.6	**200.6**
18	COLL20	1024.0	195.7	615.2	245.1	280.7	280.7	**193.7**
19	Bioresponse	1776.0	**175.5**	838.2	329.5	520.1	580.6	489.3
20	RELATHE	4322.0	**174.9**	2161.6	1416.4	317.4	1569.6	285.6
21	BASEHOCK	4862.0	**122.4**	2360.8	1203.8	1064.1	1738.6	1048.4
22	Brain	5920.0	**24.6**	2288.6	1720.0	846.5	1555.3	673.5
23	Prostate2	10509	319.3	3769.46	2788.9	227.0	2974.6	**210.6**
24	11Tumor	12533	330.1	5312.3	2573.2	383.2	3180.9	**217.4**

Table 7
Fitness values of the best subsets selected by IGPSO and the seven compared algorithms on large-scale datasets.

No.	Dataset	FULL	mRMR	ReliefF	RFS	VLPSO	SBPSO	CNNIWPSO	EARFS	CCSO	IGPSO
12	Isolet5	0.28(+)	0.19(+)	0.18(+)	0.14(+)	0.13(+)	0.24(+)	0.21(+)	0.12(+)	0.24(+)	**0.03**
13	MFS	0.13(+)	0.05(+)	0.05(+)	0.04(+)	**0.03**(=)	0.08(+)	0.05(+)	0.04(+)	0.05(+)	0.03
14	Gametes	0.54(+)	0.46(=)	0.49(+)	0.48(=)	0.48(=)	0.50(+)	0.48(=)	0.48(=)	0.47(=)	**0.46**
15	QAR	0.42(+)	0.36(+)	0.35(+)	0.32(+)	0.36(+)	0.35(+)	0.34(+)	0.16(+)	0.32(+)	**0.08**
16	QOT	0.35(+)	0.26(+)	0.24(+)	0.26(+)	0.27(+)	0.28(+)	0.26(+)	0.14(+)	0.25(+)	**0.08**
17	ORL	0.16(+)	0.15(+)	0.21(+)	0.16(+)	0.21(+)	0.20(+)	0.20(+)	0.16(+)	0.15(+)	**0.13**
18	COLL20	0.12(+)	0.06(+)	0.05(+)	**0.03**(=)	0.04(+)	0.08(+)	0.04(+)	0.04(+)	0.04(+)	0.03
19	Bioresponse	0.34(+)	0.26(+)	0.27(+)	0.27(+)	0.23(+)	0.28(+)	0.25(+)	0.21(+)	0.26(+)	**0.19**
20	RELATH	0.27(+)	0.20(+)	0.20(+)	0.20(+)	0.19(+)	0.21(+)	0.20(+)	0.10(=)	0.19(+)	**0.10**
21	BASEHOCK	0.27(+)	0.13(+)	0.18(+)	0.23(+)	0.10(+)	0.18(+)	0.14(+)	0.10(+)	0.15(+)	**0.09**
22	Brain	0.45(+)	0.49(+)	0.34(+)	0.28(=)	0.42(+)	0.36(+)	0.35(+)	0.31(+)	0.33(+)	**0.28**
23	Prostate2	0.25(+)	0.21(+)	0.13(+)	0.10(+)	0.11(+)	0.12(+)	0.11(+)	0.05(+)	0.08(+)	**0.04**
24	11Tumor	0.27(+)	0.19(+)	0.16(+)	0.15(+)	0.16(+)	0.17(+)	0.10(+)	0.06(=)	0.08(+)	**0.06**
	W/D/L	13/0/0	12/1/0	13/0/0	10/3/0	11/2/0	13/0/0	12/1/0	10/3/0	12/1/0	N/A
	RANK	9.54	6.46	6.46	5.08	5.31	8.08	5.46	2.85	4.62	1.15

5.2. Ablation study

5.2.1. Ablation experiment on the proportion k of positive samples

In this section, we explore the impact of the ratio of positive and negative samples on the performance of the algorithm. First, we selected two datasets for the ablation experiment from the small-scale datasets and the large-scale datasets, respectively. Then we set 6 proportions of 50%, 60%, 70%, 80%, 90%, and 100%. The results are shown in Fig. 2. From Fig. 2, we can see that for all datasets, with the increase of the proportion k, the classification accuracy first increases and then decreases. For the Movement and MFS datasets, the classification accuracy reaches the maximum value when $k = 70$, and for the other two datasets are 80 and 90, respectively, but the classification accuracy at $k = 70$ is still a relatively ideal result. Therefore, we chose $k = 70$ as the parameter setting for this experiment.

5.2.2. Ablation experiments for different modules

In this section, we conduct ablation experiments on different modules in IGPSO to demonstrate that the excellence of IGPSO is attributed to all the modules rather than a single module. First, we utilize the focal neural network, and use uniform sampling instead of PSO to generate subsets. The subset generation method is detailed in [46], and this algorithm is denoted as Focal-1. Second, we replace the focal neural network with ReliefF to illustrate the effectiveness of the focal neural network, and this model is named as ReliefF-PSO. Third, we eliminate the positive and negative training operation and only perform positive sample training, which is named as IGPSO-wpn, to demonstrate the effectiveness of positive and negative sample training. We use the same four datasets for this ablation experiment, and present our results in Fig. 3.

Fig. 3 illustrates the feature selection ability of the algorithms on different datasets. ReliefF-PSO performs the worst of all algorithms, which may be due to the fact that the focal neural network has a strong feature ranking ability and can better represent the relationship

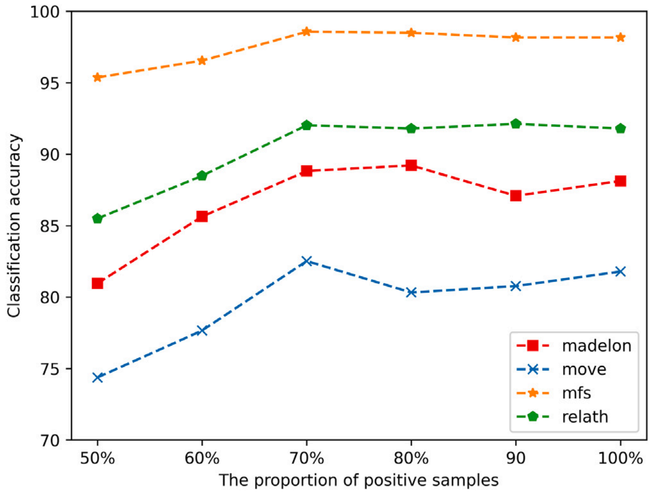


Fig. 2. The change of classification accuracy with the proportion of positive samples.

between features than ReliefF. Besides, the performance of Focal-1 is also unsatisfactory, which may be due to the fact that the non-evolutionary algorithm cannot sample the entire subset space well, thus missing some potential subsets. It is worth noting that for the MFS dataset, Focal-1 achieves the same classification accuracy as IGPSO, which may be due to the particular dataset. Finally, the performance of IGPSO-wpn is slightly worse than that of IGPSO, which illustrates the effectiveness of the positive and negative sample training methods.

5.2.3. Ablation experiments for parameter settings

In this section, we conduct ablation experiments on different parameter settings, including population size, number of iterations. First, we

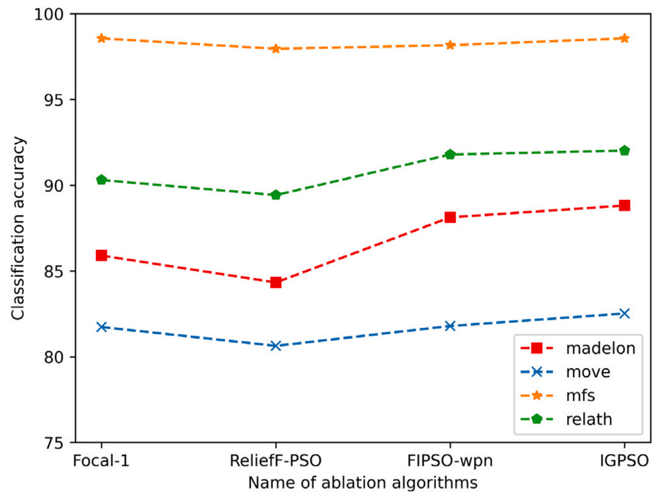


Fig. 3. Classification accuracy of different methods.

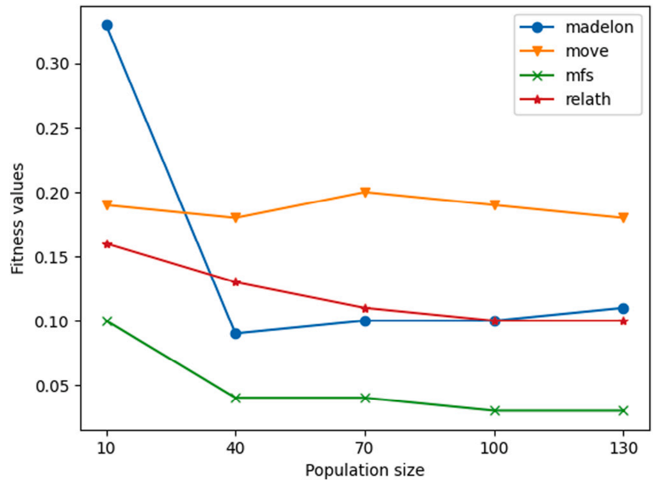


Fig. 4. The fitness values of different population size.

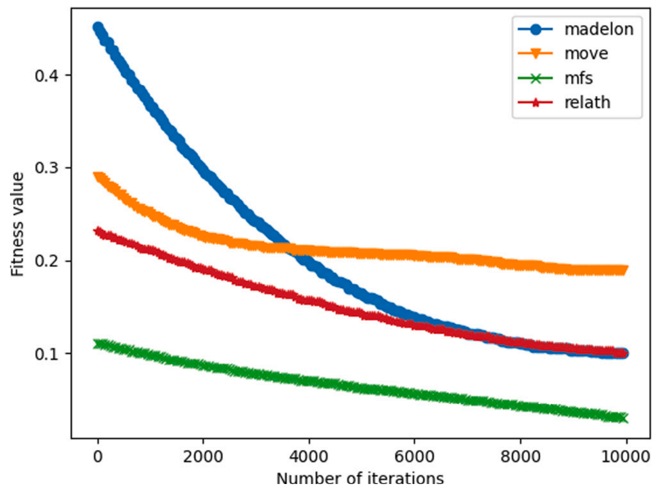


Fig. 5. The change of fitness value with the increase of the number of iterations.

select five different population sizes as the parameters of the model, and compare the fitness value of the model in different settings. The result is shown in Fig. 4. Then, we set the ME from 0 to 10000, then recorded the variation of fitness value with the number of iterations, which is shown in Fig. 5.

It can be seen from Fig. 4 that for most datasets, $ps = 10$ is not a satisfactory choice. When the ps is between 40 and 130, the results are different, which may be related to the characteristics of the datasets themselves. But it is worth mentioning that when the ps is between 40 and 130, the overall performance of the algorithm is satisfactory. The reason for this phenomenon may be that when the population size is too small, the particles is more likely to fall into the local optimal solution, so that the satisfactory results cannot be obtained. In order to ensure the fairness of the experiments, we set the ps to 100 in the comparison with other methods, just as the settings in other PSO-based methods.

Fig. 5 shows the variation of fitness value with the number of iterations. Obviously, the fitness value decreases when the number of iterations increases. In particular, at the beginning of the iteration, the fitness value decreases faster, and as the number of iterations increases, the decline slows down. However, it is important to note that the time cost increases as the number of iterations increases. Therefore, in order to save the time consuming on the one hand, and to maintain the fairly comparison with other algorithms on the other hand, we choose $ME = 10000$ in the comparison experiments.

6. Conclusions and future work

In this paper, a novel importance-guided particle swarm optimization with MLP is proposed to address the challenge of large-scale feature selection. The method utilizes positive and negative datasets to train an importance vectors by a focal neural network, thereby capturing the importance associated with features. The IGPSO algorithm is employed to coordinate population exploitation and exploration, enabling rapid convergence of the particle swarm in the early stages of iteration while focusing on potential space in later stages. IGPSO has been evaluated on both large-scale and small-scale datasets, demonstrating superior performance compared to state-of-the-art FS algorithms on 24 datasets, particularly for large-scale datasets. However, a limitation of this approach lies in the occupation of video memory during focal neural network training. Moreover, the instability of neural network training will also affect the results of the method. In future research, we will explore new neural network models and investigate other EC algorithms to develop stronger feature selection abilities. For example, KAN model has the potential to replace MLP for importance vector training. In addition, there are many excellent EC algorithms, such as DE, ACO, etc., which can be used to improve the proposed method.

CRediT authorship contribution statement

Yu Xue: Supervision, Methodology, Funding acquisition. **Chenyi Zhang:** Writing – review & editing, Writing – original draft, Validation, Methodology.

Declaration of competing 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.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (62376127, 61876089, 61876185, 61902281, 62376127) and the Natural Science Foundation of Jiangsu Province, China (BK20141005). This work was partially supported by the project Distinguished Professors of Jiangsu Province, China.

Data availability

Data will be made available on request.

References

[1] S. Solorio-Fernández, J.A. Carrasco-Ochoa, J.F. Martínez-Trinidad, A review of unsupervised feature selection methods, Artif. Intell. Rev. 53 (2) (2020) 907–948, http://dx.doi.org/10.1007/s10462-019-09682-y.

[2] G. Kou, P. Yang, Y. Peng, F. Xiao, Y. Chen, F.E. Alsaadi, Evaluation of feature selection methods for text classification with small datasets using multiple criteria decision-making methods, Appl. Soft Comput. 86 (2020) 105836, http://dx.doi.org/10.1016/j.asoc.2019.105836.

[3] R.R. Mostafa, A.M. Khedr, Z. Al Aghbari, I. Afyouni, I. Kamel, N. Ahmed, An adaptive hybrid mutated differential evolution feature selection method for low and high-dimensional medical datasets, Knowl.-Based Syst. 283 (2024) 111218, http://dx.doi.org/10.1016/j.knosys.2023.111218.

[4] X. Zhou, W. Yuan, Q. Gao, C. Yang, An efficient ensemble learning method based on multi-objective feature selection, Inform. Sci. (2024) 121084, http://dx.doi.org/10.1016/j.ins.2024.121084.

[5] B.H. Nguyen, B. Xue, M. Zhang, A survey on swarm intelligence approaches to feature selection in data mining, Swarm Evol. Comput. 54 (2020) 100663, http://dx.doi.org/10.1016/j.swevo.2020.100663.

[6] C. Liang, L. Wang, L. Liu, H. Zhang, F. Guo, Multi-view unsupervised feature selection with tensor robust principal component analysis and consensus graph learning, Pattern Recognit. 141 (2023) 109632, http://dx.doi.org/10.1016/j.patcog.2023.109632.

[7] Z. Zhang, L. Liu, J. Li, X. Wu, Integrating global and local feature selection for multi-label learning, ACM Trans. Knowl. Discov. Data 17 (1) (2023) http://dx.doi.org/10.1145/3532190.

[8] S. Tijjani, M.N. Ab Wahab, M.H. Mohd Noor, An enhanced particle swarm optimization with position update for optimal feature selection, Expert Syst. Appl. 247 (2024) 123337, http://dx.doi.org/10.1016/j.eswa.2024.123337.

[9] J. He, L. Qu, P. Wang, Z. Li, An oscillatory particle swarm optimization feature selection algorithm for hybrid data based on mutual information entropy, Appl. Soft Comput. 152 (2024) 111261, http://dx.doi.org/10.1016/j.asoc.2024.111261.

[10] Y. Hu, Y. Zhang, D. Gong, Multiobjective particle swarm optimization for feature selection with fuzzy cost, IEEE Trans. Cybern. 51 (2) (2021) 874–888, http://dx.doi.org/10.1109/TCYB.2020.3015756.

[11] D. Paul, A. Jain, S. Saha, J. Mathew, Multi-objective PSO based online feature selection for multi-label classification, Knowl.-Based Syst. 222 (2021) 106966, http://dx.doi.org/10.1016/j.knosys.2021.106966.

[12] P. Dhal, C. Azad, A multi-objective feature selection method using Newton's law based PSO with GWO, Appl. Soft Comput. 107 (2021) 107394, http://dx.doi.org/10.1016/j.asoc.2021.107394.

[13] Y. Xue, H. Zhu, F. Neri, A feature selection approach based on NSGA-II with relieff, Appl. Soft Comput. 134 (2023) 109987, http://dx.doi.org/10.1016/j.asoc.2023.109987.

[14] L. Qu, W. He, J. Li, H. Zhang, C. Yang, B. Xie, Explicit and size-adaptive PSO-based feature selection for classification, Swarm Evol. Comput. 77 (2023) 101249, http://dx.doi.org/10.1016/j.swevo.2023.101249.

[15] Z. Deng, T. Li, D. Deng, K. Liu, P. Zhang, S. Zhang, Z. Luo, Feature selection for label distribution learning using dual-similarity based neighborhood fuzzy entropy, Inform. Sci. 615 (2022) 385–404, http://dx.doi.org/10.1016/j.ins.2022.10.054.

[16] B. Liu, L. Wang, Y.-H. Jin, F. Tang, D.-X. Huang, Improved particle swarm optimization combined with chaos, Chaos Solitons Fractals 25 (5) (2005) 1261–1271, http://dx.doi.org/10.1016/j.chaos.2004.11.095.

[17] K. Yu, D. Zhang, J. Liang, K. Chen, C. Yue, K. Qiao, L. Wang, A correlation-guided layered prediction approach for evolutionary dynamic multiobjective optimization, IEEE Trans. Evol. Comput. 27 (5) (2023) 1398–1412, http://dx.doi.org/10.1109/TEVC.2022.3193287.

[18] F. Zhao, H. Zhang, L. Wang, A Pareto-based discrete jaya algorithm for multiobjective carbon-efficient distributed blocking flow shop scheduling problem, IEEE Trans. Ind. Inform. 19 (8) (2023) 8588–8599, http://dx.doi.org/10.1109/TII.2022.3220860.

[19] B.H. Nguyen, B. Xue, P. Andreae, M. Zhang, A new binary particle swarm optimization approach: Momentum and dynamic balance between exploration and exploitation, IEEE Trans. Cybern. 51 (2) (2021) 589–603, http://dx.doi.org/10.1109/TCYB.2019.2944141.

[20] H. Yu, K.-Z. Gao, Z.-F. Ma, Y.-X. Pan, Improved meta-heuristics with Q-learning for solving distributed assembly permutation flowshop scheduling problems, Swarm Evol. Comput. 80 (2023) 101335, http://dx.doi.org/10.1016/j.swevo.2023.101335.

[21] Y. Ren, K. Gao, Y. Fu, H. Sang, D. Li, Z. Luo, A novel Q-learning based variable neighborhood iterative search algorithm for solving disassembly line scheduling problems, Swarm Evol. Comput. 80 (2023) 101338, http://dx.doi.org/10.1016/j.swevo.2023.101338.

[22] M. Gao, K. Gao, Z. Ma, W. Tang, Ensemble meta-heuristics and Q-learning for solving unmanned surface vessels scheduling problems, Swarm Evol. Comput. 82 (2023) 101358, http://dx.doi.org/10.1016/j.swevo.2023.101358.

[23] H. Li, K. Gao, P.-Y. Duan, J.-Q. Li, L. Zhang, An improved artificial bee colony algorithm with Q-learning for solving permutation flow-shop scheduling problems, IEEE Trans. Syst. Man Cybern.: Syst. 53 (5) (2023) 2684–2693, http://dx.doi.org/10.1109/TSMC.2022.3219380.

[24] C. Li, Y. Huang, Y. Xue, Dependence structure of gabor wavelets based on copula for face recognition, Expert Syst. Appl. 137 (2019) 453–470, http://dx.doi.org/10.1016/j.eswa.2019.05.034.

[25] T. Yin, H. Chen, J. Wan, P. Zhang, S.-J. Horng, T. Li, Exploiting feature multi-correlations for multilabel feature selection in robust multi-neighborhood fuzzy β covering space, Inf. Fusion 104 (2024) 102150, http://dx.doi.org/10.1016/j.inffus.2023.102150.

[26] T. Yin, H. Chen, Z. Yuan, J. Wan, K. Liu, S.-J. Horng, T. Li, A robust multilabel feature selection approach based on graph structure considering fuzzy dependency and feature interaction, IEEE Trans. Fuzzy Syst. 31 (12) (2023) 4516–4528, http://dx.doi.org/10.1109/TFUZZ.2023.3287193.

[27] G. Hu, B. Du, X. Wang, G. Wei, An enhanced black widow optimization algorithm for feature selection, Knowl.-Based Syst. 235 (2022) 107638, http://dx.doi.org/10.1016/j.knosys.2021.107638.

[28] B. Jiang, Y. Liu, H. Geng, Y. Wang, H. Zeng, J. Ding, A holistic feature selection method for enhanced short-term load forecasting of power system, IEEE Trans. Instrum. Meas. 72 (2023) 1–11, http://dx.doi.org/10.1109/TIM.2022.3219499.

[29] Z. Zhao, L. Wang, H. Liu, J. Ye, On similarity preserving feature selection, IEEE Trans. Knowl. Data Eng. 25 (3) (2013) 619–632, http://dx.doi.org/10.1109/TKDE.2011.222.

[30] H. Peng, F. Long, C. Ding, Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy, IEEE Trans. Pattern Anal. Mach. Intell. 27 (8) (2005) 1226–1238, http://dx.doi.org/10.1109/TPAMI.2005.159.

[31] J.-Q. Yang, Q.-T. Yang, K.-J. Du, C.-H. Chen, H. Wang, S.-W. Jeon, J. Zhang, Z.-H. Zhan, Bi-directional feature fixation-based particle swarm optimization for large-scale feature selection, IEEE Trans. Big Data 9 (3) (2023) 1004–1017, http://dx.doi.org/10.1109/TBDATA.2022.3232761.

[32] W. Ding, Y. Sun, M. Li, J. Liu, H. Ju, J. Huang, C.-T. Lin, A novel spark-based attribute reduction and neighborhood classification for rough evidence, IEEE Trans. Cybern. 54 (3) (2024) 1470–1483, http://dx.doi.org/10.1109/TCYB.2022.3208130.

[33] M. Lefoane, I. Ghafir, S. Kabir, I.-U. Awan, Unsupervised learning for feature selection: A proposed solution for botnet detection in 5G networks, IEEE Trans. Ind. Inform. 19 (1) (2023) 921–929, http://dx.doi.org/10.1109/TII.2022.3192044.

[34] V. Subrahmanyam, V. Janaki, P.S. Rao, N. Gurrapu, S.K. Mandala, R. Roshan, Internet of things(IoT) based data analysis for feature selection by hybrid swarm intelligence(SI) algorithm, in: 2024 IEEE International Conference on Interdisciplinary Approaches in Technology and Management for Social Innovation, vol. 2, 2024, pp. 1–6, http://dx.doi.org/10.1109/IATMSI60426.2024.10503278.

[35] J. Li, K. Cheng, S. Wang, F. Morstatter, R.P. Trevino, J. Tang, H. Liu, Feature selection: A data perspective, ACM Comput. Surv. 50 (6) (2017) 1–45, http://dx.doi.org/10.1145/3136625.

[36] W. Ding, C.-T. Lin, Z. Cao, Deep neuro-cognitive co-evolution for fuzzy attribute reduction by quantum leaping PSO with nearest-neighbor memeplexes, IEEE Trans. Cybern. 49 (7) (2019) 2744–2757, http://dx.doi.org/10.1109/TCYB.2018.2834390.

[37] S. Mirjalili, A. Lewis, S-shaped versus V-shaped transfer functions for binary particle swarm optimization, Swarm Evol. Comput. 9 (2013) 1–14, http://dx.doi.org/10.1016/j.swevo.2012.09.002.

[38] Y.-W. Jeong, J.-B. Park, S.-H. Jang, K.Y. Lee, A new quantum-inspired binary PSO: Application to unit commitment problems for power systems, IEEE Trans. Power Syst. 25 (3) (2010) 1486–1495, http://dx.doi.org/10.1109/TPWRS.2010.2042472.

[39] K. Chen, B. Xue, M. Zhang, F. Zhou, An evolutionary multitasking-based feature selection method for high-dimensional classification, IEEE Trans. Cybern. 52 (7) (2022) 7172–7186, http://dx.doi.org/10.1109/TCYB.2020.3042243.

[40] Y. Zhang, S. Wang, P. Phillips, G. Ji, Binary PSO with mutation operator for feature selection using decision tree applied to spam detection, Knowl.-Based Syst. 64 (2014) 22–31, http://dx.doi.org/10.1016/j.knosys.2014.03.015.

[41] B. Tran, B. Xue, M. Zhang, Variable-length particle swarm optimization for feature selection on high-dimensional classification, IEEE Trans. Evol. Comput. 23 (3) (2019) 473–487, http://dx.doi.org/10.1109/TEVC.2018.2869405.

[42] R. Cheng, Y. Jin, A competitive swarm optimizer for large scale optimization, IEEE Trans. Cybern. 45 (2) (2015) 191–204, http://dx.doi.org/10.1109/TCYB.2014.2322602.

[43] B.H. Nguyen, B. Xue, M. Zhang, A constrained competitive swarm optimizer with an SVM-based surrogate model for feature selection, IEEE Trans. Evol. Comput. 28 (1) (2024) 2–16, http://dx.doi.org/10.1109/TEVC.2022.3197427.

[44] F. Nie, H. Huang, X. Cai, C. Ding, Efficient and robust feature selection via joint l2,1-norms minimization, in: Proceedings of the 23rd International Conference on Neural Information Processing Systems, NIPS '10, vol. 2, 2010, pp. 1813–1821.

[45] Y.N. Pawan, K.B. Prakash, S. Chowdhury, Y.-C. Hu, Particle swarm optimization performance improvement using deep learning techniques, Multimedia Tools Appl. 81 (19) (2022) 27949–27968, http://dx.doi.org/10.1007/s11042-022-12966-1.

[46] Y. Xue, C. Zhang, F. Neri, M. Gabbouj, Y. Zhang, An external attention-based feature ranker for large-scale feature selection, Knowl.-Based Syst. 281 (2023) 111084, http://dx.doi.org/10.1016/j.knosys.2023.111084.