Computer Modeling in Engineering & Sciences CMES, vol.121, no.2, pp.569-592, 2019

Feature Selection with a Local Search Strategy Based on the Forest Optimization Algorithm

Tinghuai Ma[1,*], Honghao Zhou[1], Dongdong Jia[1], Abdullah Al-Dhelaan[2], Mohammed Al-Dhelaan[2] and Yuan Tian[3]

Abstract: Feature selection has been widely used in data mining and machine learning. Its objective is to select a minimal subset of features according to some reasonable criteria so as to solve the original task more quickly. In this article, a feature selection algorithm with local search strategy based on the forest optimization algorithm, namely FSLSFOA, is proposed. The novel local search strategy in local seeding process guarantees the quality of the feature subset in the forest. Next, the fitness function is improved, which not only considers the classification accuracy, but also considers the size of the feature subset. To avoid falling into local optimum, a novel global seeding method is attempted, which selects trees on the bottom of candidate set and gives the algorithm more diversities. Finally, FSLSFOA is compared with four feature selection methods to verify its effectiveness. Most of the results are superior to these comparative methods.

Keywords: Feature selection, local search strategy, forest optimization, fitness function.

1 Introduction

Feature selection methods are widely used in data mining and machine learning in order to improve the accuracy of classifiers and accelerate the training speed of models with large amounts of input features [Migdady (2013); Rong, Ma, Cao et al. (2019)]. From the original feature set, it allows us to eliminate irrelevant and strongly correlated redundant features and so to extract those valuable features [Paisitkriangkrai, Shen and Hengel (2016); Ma, Shao, Hao et al. (2018)].

Feature selection is a difficult task due mainly to a large search space, where the total number of possible solutions is 2^n for a dataset with n features [Xue, Zhang, Browne et al. (2016)]. It aims to decide whether to choose the feature in the dataset or not. The task becomes more challenging while n is increasing in many areas with the advances in the

[1]College of Computer and Software, Nanjing University of information science & Technology, Nanjing, China.
[2]College of Computer and Information Sciences, KingSaud University, Riyadh, Saudi Arabia.
[3]Nanjing Institute of Technology, Nanjing, China.
*Corresponding Author: Tinghuai Ma. Email: thma@nuist.edu.cn.

data collection techniques [Tzanis (2017)]. As we all known, an exhaustive search for the best feature subset of a given dataset is practically impossible in most situations, even for a moderate-sized dataset [Wipf and Nagarajan (2016)]. It is not excluded that quantum computing will probably provide enough computing power that makes violent search on today's dataset possible in the future [Vandersypen and Leeuwenhoek (2017); Lv, Ma, Tang et al. (2016)]. At that time, there is no doubt that bigger datasets will be available to scientists [Stuart (2016); Zhang, Ma, Cao et al. (2015)]. An effective way to find suitable feature from huge datasets is randomly set search start point at first [Melian and Verdegay (2011)]. Then the question is how to optimize the solution rapidly and directly like gradient descent using in optimizing Deep Learning [Keskar, Mudigere, Nocedal et al. (2016)]. According to the existing cognition, random search strategies coming up with approximately best solution are still our best choices for feature selection problem [Zhu, Xu, Chen et al. (2016)].

Our previous work on feature selection research obtained a method called FSFOACD [Ma, Jia, Zhou et al. (2018); Rong, Ma, Cao et al. (2019)]. That method used the contribution degree strategy to improve the efficiency of the forest optimization algorithm. In another word, it used the measure of the relevance and redundancy to guide the search direction of the forest optimization algorithm. The search direction, to a certain extent, improves the performance of the forest optimization algorithm. However, there are still some deficiencies in FSFOACD. For example, the number of features selected by each tree in the forest in the initialization phase is randomly generated. During the initialization, if most of the random number of selected features of the trees are close to the number of features in the original dataset, the search locations of the trees in the forest will be far away from the optimal solution, which also results in the low efficiency of the algorithm at the beginning of the search. On the other hand, during the local seeding process, some neighbor trees are added to the forest, who is strictly limited by their parents. The limited number and randomness also cost this algorithm a comparatively long time to achieve a well-accepted feature subset. Only selecting one feature at a time cannot guarantee the quality of feature subsets and so the search efficiency is lower.

In this paper, a novel feature selection method with a local search strategy based on the forest optimization algorithm is proposed, namely FSLSFOA. Firstly, a feature subset size determination mechanism is used to initialize the forest optimization algorithm, which allows a small feature subset for trees in the forest during the initialization phase. In this way, more trees are in an ideal position at the beginning, since they are separated more evenly into the search space. This initialization strategy greatly improves the search efficiency of the algorithm.

Secondly, the search strategy of FSFOACD is utilized as well as improved. Each time when adding a neighbor node, the algorithm no longer selects single feature for evaluation. Instead, it selects a probabilistic number of high-quality features and an empirical number of low-quality features at a time. The new search strategy can not only maximize the quality of the feature subsets in the forest, but also improve the search efficiency of the algorithm to a great extent.

Thirdly, a new fitness function to balance the classification accuracy and the dimensionality

reduction ratio is proposed, since the core purpose of feature selection is to increase dimensionality reduction ratio on the premise of ensuring the classification accuracy. In order to verifying the effectiveness of our algorithm, 10 widely-used datasets from UCI are selected and three classifiers respectively from support vector machine (SVM) [Aburomman and Reaz (2016)], decision tree (DT) [Shirani, Habibi, Besalatpour et al. (2015)] and k-nearest neighbor (KNN) [Benmahamed, Teguar and Boubakeur (2018)]. There are 4 feature selection methods are selected for comparison, including UPFS [Dadaneh, Markid and Zakerolhosseini (2016)], ANFS [Luo, Nie, Chang et al. (2018)], PSO(4-2) [Tran, Xue and Zhang (2014b)] and FSFOACD [Ma, Jia, Zhou et al. (2018)]. The experiments show that most of the results are superior to previous feature selection algorithms we selected.

The rest of this paper is organized as follows: In Section 2, related works on feature selection are reviewed. Moreover, details about the proposed method are introduced in Section 3. In Section 4, the proposed algorithm is compared with the other existing feature selection methods. Finally, Section 5 summarizes the present study.

2 Related work

Feature selection involves two conflict objectives, which are to maximize the classification accuracy and minimize the number of features [Hancer, Xue, Zhang et al. (2017)]. Therefore, feature selection can be treated as a multi-objective problem to find a set of trade-off solutions between these two objectives [Selvam, Kumar and Siripuram (2017)].

An important problem of feature selection is the feature interaction (or epistasis) [Guyon (2003)]. There can be two-way, three-way, or complex multi-way interactions among features [Gorelick and Bertram (2010)]. A feature weakly relevant to the target concept can significantly improve the classification accuracy if it is used together with some complementary features [Anthimopoulos, Christodoulidis, Ebner et al. (2016)]. In contrast, an individually relevant feature may become redundant when used together with other features [Sun, Goodison, Li et al. (2007)].

Based on the evaluation criteria, feature selection algorithms are generally classified into two categories: 1) filter approaches and 2) wrapper approaches [Ambusaidi, He, Nanda et al. (2016); Zawbaa, Emary, Hassanien et al. (2016)]. Their main difference is that wrapper approaches include a classification/learning algorithm in feature subset evaluation, while filter algorithms are independent of any classification algorithms [Tran, Zhang and Xue (2016)]. Filters ignore the performance of the selected features, whereas wrappers not, which usually results in those wrappers getting better solutions [Yang, Liu, Zhou et al. (2013)]. Choosing filters or wrappers depends on your requirement for accuracy and time efficiency [Rodriguezgaliano, Luqueespinar, Chicaolmo et al. (2018)].

According to the filter and wrapper categories, the unsupervised learning and supervised learning methods are adopted for feature selection. Forest Optimization Algorithm (FOA) and its improvments [Ma, Jia, Zhou et al. (2018)] are supervised methods. Also, Particle Swarm Optimization (PSO) [Tran, Xue and Zhang (2014b)] method is supervised mthods. While choosing features, it will consider the classification performance. Differently, on

the other hand, UPFS [Dadaneh, Markid and Zakerolhosseini (2016)] and ANFS [Luo, Nie, Chang et al. (2018)] are belong to unsupervised method. The labels of the dataset are masked while using these two methods. Unsupervised methods are the ultimate goal of feature selection, and so they occupy an important position in this field.

2.1 FOA

Forest Optimization Algorithm (FOA) is an classic supervised learning, which was proposed by Ghaemi et al. [Ghaemi and Feizi-Derakhshi (2014)]. And we propsed the feature contribution to promote the effecient based on Forest Optimization Algorithm [Ma, Jia, Zhou et al. (2018)].

FOA as a evolutionary algorithms, which is suitable for optimization task. Ghaemi et al. found FOA can improve the classification accuracy comparing with other feature selection methods [Ghaemi and Feizi-Derakhshi (2016)]. Fernández-García et al. compared the FOA with other feature selection method on Academic Data to foresee if students will finish their degree after finishing their first year in college [Fernández-García, Iribarne, Corral et al. (2018)].

Although the FOA get the better result in above experiments, the search efficiency is still low and the algorithm complexity is relative higher. At the beginning, the random initiation of parameter can not ensure the best results. So, we will do future work to how to optimize the initiate the parameter and optimize the search strategies.

2.2 PSO

Feature selection is a combinatorial optimization problem, which can be solved by Particle Swarm Optimization (PSO) [Zhang, Wang, Sui et al. (2017)]. Xue et al. studied on multi-objective particle swarm optimization (PSO) for feature selection, and it achieved comparable results with the existing well-known multi-objective algorithms in most cases [Xue, Zhang and Browne (2013)]. In 2014, Tran et al. improved the PSO to classification problems with thousands or tens of thousands of features [Tran, Xue and Zhang (2014a)]. At same time, Tran et al. proposed three new initialization strategies and three new personal best and global best updating mechanisms in PSO to develop feature selection approaches, which named as PSO(4-2) [Tran, Xue and Zhang (2014b)].

As the particle swarm in the search process will fall into the local optimal solution, there need solution to make the particles escape from the local optimal solution, and achieve the purpose of large data optimization. Usually, PSO can not assure the global optimization results.

2.3 UPFS

Unsupervised feature selection just evalue the similarity between features and remove the redundancy features therein. Mitra et al. proposed an unsupervised feature selection algorithm suitable for data sets, large in both dimension and size [Mitra, Murthy and Pal (2002)]. Hou et al. first combined the embedding learning and feature ranking together,

and proposed joint embedding learning and sparse regression (JELSR) to perform feature selection [Hou, Nie, Li et al. (2014)].

Dadaneh proposed unsupervised probabilistic feature selection using ant colony optimization (UPFS) [Dadaneh, Markid and Zakerolhosseini (2016)]. They utilized the inter-feature information that showed the similarity between the features that led the algorithm to decreased redundancy in the final set. As ant colony optimization is heuristic algorithm as FOA, we choose this method as reference to verify our proposed algorithm.

Luo et al. [Luo, Nie, Chang et al. (2018)] poposed an adaptive unsupervised feature selection with structurer regularization, which characterized the intrinsic local structure by an adaptive reconstruction graph and simultaneously consider its multiconnected-components (multicluster) structure.

3 Methods

In order to solve part of the problems in related works, a feature selection algorithm with local search strategy based on forest optimization algorithm is proposed, namely FSLSFOA. There are quite lot of improvements and modifications while comparing with FSFOACD. FSLSFOA involves five main stages: (1) Initialize trees, (2) Group features, (3) Local seeding, (4) Size limitation, (5) Global seeding.

3.1 Initialization based on feature subset size determination mechanism

Since the main purpose of feature selection is to find the minimum feature subset necessary and sufficient to identify the target, the optimal feature subset generally has fewer features selected. In the initialization phase, the starting location of the tree in the forest is hoped to be closer to the optimal solution. In other word, the difference between the optimal feature subset size and the initialized feature subset size is not very large. Probabilistic random function is utilized to determine the initial size of the feature subset. The probability formula is defined as follows:

$$P(sf) = \frac{f - sf}{\sum_{l=1}^{L}(f - l)} \tag{1}$$

where f represents the number of original features of the dataset, sf represents the number of selected features and L indicates the distance between f and sf, that is $L = f - sf$. $P(sf)$ determines the initial probability of the number of selected features. According to Eq. (1), when sf is minimized, $P(sf)$ reaches the peak. Next, the roulette gamble algorithm is utilized to determine the number of sf. That is to say sf is randomly selected, its range of values is $[m, M]$. The value of m is set to 3, since it is an efficient and comprehensive lower bound of sf for most of the dataset. It is also allowed to make necessary adjustments to some extremely simple datasets in practice. Besides, $M = \varepsilon * f$. Where f depends totally on the given dataset and ε is used to adjust the size of M. If ε is set to be close to 1, the value of M will certainly be close to f. This will make the search space very large, which leads to the increasing of the computation cost and lots of invalid feature subsets. Our goal is to select a significant subset of features with a small number of

features. The discussion on ε will be explained in our experiment parameter setting section. The roulette gambling algorithm is designed as follows:

Algorithm 1 Roulette Gambling Algorithm

```
Randomly generated U between [0, 1]
sum = 0
for each sf ∈ [m, M] do
    sum = sum + P(sf)
    if U< sum then
        break;
    end if
end for
return sf
```

3.2 Grouping strategy based on feature importance

After the initialization phase, all features are divided into high-quality feature groups and low-quality feature groups according to their importance. The purpose of this grouping is to provide a reference for the forest optimization algorithm when selecting features. Those high-quality features are distributed into the forest optimization algorithm, thereby improving the quality of the trees (character subsets) in the forest. Here the Pearson correlation coefficient is applied to measure the correlation between different features [Mu, Liu and Wang (2017)]. Here the Pearson correlation coefficient is applied because of its simplicity and because it is simpler to manipulate as a univariate method [Coelho, Braga and Verleysen (2010)]. Extending to the subsequent importance, this part aims at the first round of coarse-grained partitioning of features. The definition is as follows:

$$C(i,j) = \frac{\sum_{k=1}^{f}(x_i(k)-\overline{x}_i)(x_j(k)-\overline{x}_j)}{\sqrt{\sum_{k=1}^{f}(x_i(k)-\overline{x}_i)^2}\sqrt{\sum_{k=1}^{f}(x_j(k)-\overline{x}_j)^2}} \tag{2}$$

where $C(i,j)$ is the correlation coefficient of feature i and feature j. f is the number of samples. $x_i(k)$ and $x_j(k)$ represents the value of i and j in the kth sample. $\overline{x}_i$ and $\overline{x}_j$ represents the mean value of x_i and x_j in the total f samples, respectively. According to Eq. (2), the correlation coefficient between two features weighs the similarity between features. The higher the correlation coefficient is, the higher the similarity between two features is. On the contrary, a lower correlation coefficient means that the similarity between two features is also comparatively low. After calculating the correlation coefficients for all possible combinations of features, the importance of feature i is calculated as follows:

$$contribute(i) = |C(i,c)| - \frac{1}{f-1}\sum_{j=1}^{f}|C(i,j)| \qquad if\ i \neq j \tag{3}$$

where $C(i, c)$ represents the correlation coefficient between feature i and class attribute c. The larger the value of $C(i, c)$ is, the greater the influence of feature i is on the classification effect. Conversely, the smaller the value is, the less the influence of the feature on the classification effect will be. If the importance value of a feature is high, it means this feature has high correlation with the class attribute and its redundancy is low. On the contrary, if the importance value of a feature is low, it means this feature has low correlation with the class attribute and its redundancy is high. For the forest optimization algorithm to select higher-quality feature subsets in the process of exploring the optimal solution, all the features are sorted according to their importance values in descending order and divided into two groups. The first half of features are placed in a high quality feature group ($Group\_high$) and the second half feature in a low quality feature ($Group\_low$). The importance of features in $Group\_high$ is greater than those in $Group\_low$, and the features of both groups are sorted in descending order according to their importance value. Fig. 1 is a schematic diagram of the feature grouping strategy.

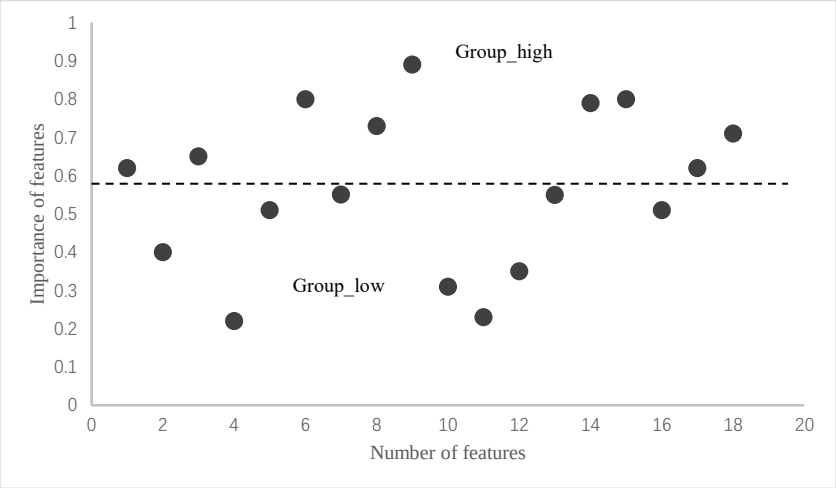


Figure 1: Feature grouping based on the importance of features

3.3 Local search strategy

After $Group\_high$ and $Group\_low$ produced by the feature grouping strategy, local search strategy is utilized to select suitable features from the two groups. This strategy aims to provide better locations for forest optimization algorithms to further explore optimal solutions. The local search strategy mainly improves the optimal solution searching of forest optimization algorithm through "add" and "delete" operations. For short, the "add" operation is defined as to select the desired number of features and use the "delete" operation is to remove a certain number of features from the current position of the tree. The "add" operation is to add high-class-correlation and low-redundancy features into the tree. The "delete" operation removes low-class-correlation and high-redundancy features from the tree. In the local seeding stage, a novel local search strategy is proposed to dynamically

change the position of the neighbor tree by adding and deleting as many features as possible from $Group\_high$ and $Group\_low$ in each time of adding neighbor tree.

The specific steps of the local search strategy are as follows:

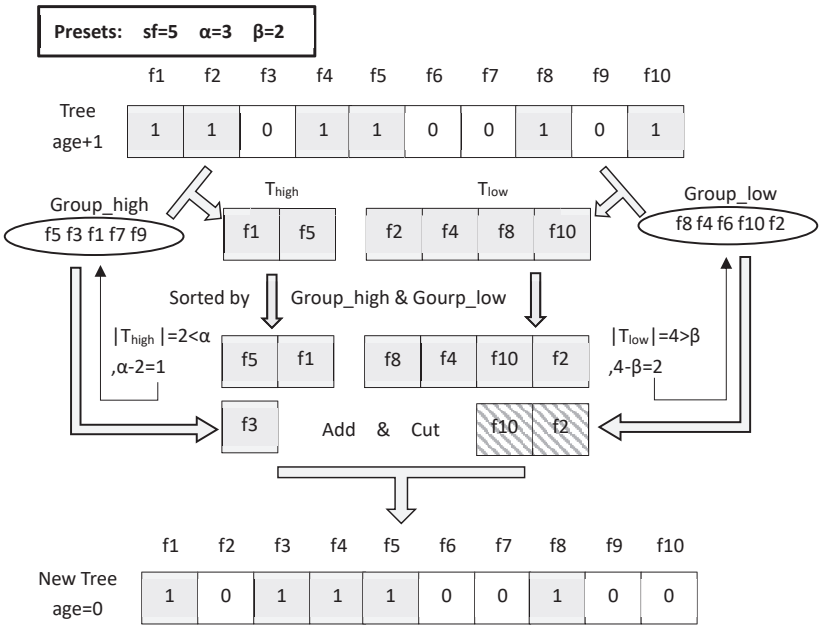


Figure 2: Feature grouping based on the importance of features

Firstly, for each tree of age 0 in the forest, its features which equals 1 are put into a subset T. This subset T includes all the randomly selected features at the beginning of local search strategy. As is shown in Fig. 2, if the current tree is like $Tree = [1, 1, 0, 1, 1, 0, 0, 1, 0, 1, age = 0]$, a subset T with six features is obviously obtained, $T = f_1, f_2, f_4, f_5, f_8, f_{10}$. In this example, it is assumed that $Group_high = f_5, f_3, f_1, f_7, f_9$and $Group\_low = f_8,]f_4, f_6, f_{10}, f_2$,which should be calculated through grouping strategy in part 3.2.

Secondly, T is grouped to two subsets, T_{high} and T_{low}, by $Group\_high$ and $Group\_low$. T_{high} contains high-quality features in T, which also included in $Group\_high$. On contrast, T_{low} includes low-quality features in T, which contains in $Group\_low$. T_{high} and T_{low} are sorted in descending order according to their importance value respectively. $T_{high} = f_5, f_1$ and $T_{low} = f_8, f_4, f_{10}, f_2$ can be easily inferred from Fig. 2 ,because $f_5, f_1 \in Group\_high$ and $f_8, f_4, f_{10}, f_2 \in Group\_low$.

Thirdly, the "add" operation is used to select the desired features, and the "delete" operation is to remove a certain number of features from the current tree. How many features empirically should being in T_{high} and T_{low} depends on parameter α and β, where $\alpha = \lambda * sf$, $\beta = (1 - \lambda) * sf$. Where λ is specified by the user and sf is initialized by the feature subset size determination mechanism in part 3.1. The parameter λ is used to control

the proportion of features selected from $Group\_high$ and $Group\_low$. The discussion on λ will be explained in our experiment parameter setting section.

If the number of high quality features in the tree, $|T_{high}|$, is smaller than α, there will be $\alpha - |T_{high}|$ of high-quality features added into the current tree selected from the set $\{Group\_high - T_{high}\}$, which means included in $Group\_high$ and not included in T_{high}. Otherwise, if $|T_{high}|$ is larger than α, add_num of high-quality features will be added into the current tree from the set $\{Group\_high - T_{high}\}$. The parameter add_num is defined as follow:

$$add\_num = argmin(Sign(Random(1) - \frac{\sum_{y=1}^{|Group\_high - T_{high}| - add\_num} y^2}{\sum_{y=1}^{|Group\_high - T_{high}|} y^2})) \tag{4}$$

$$Sign(v) = \begin{cases} v & v \geq 0 \\ 1 & v < 0 \end{cases} \tag{5}$$

where add_num and y are natural number ranged in $[1, |Group\_high - T_{high}|]$. $|Group\_high - T_{high}|$ indicates the feature number of the set $\{Group\_high - T_{high}\}$. $Random(1)$ is an uniformly distributed random variable ranged in $(0, 1]$. The function $Sign(v)$ means if $v < 0$, ved will be set to 1, which announces that v will be withdrawn from the competition of $argmin(\cdot)$. Here v represents the value inside function $Sign(\cdot)$. The $argmin(\cdot)$ help us obtain the add_num that minimize the result of $Sign(\cdot)$.

In addition, if the number of low-quality features in the tree, $|T_{low}|$, is bigger than β, there will be $|T_{low}| - \beta$ of features with low quality deleted from T_{low}. Otherwise, if $|T_{low}|$ is smaller than β, del_num of features will be deleted from T_{low}. The parameter del_num is calculated as follow:

$$del\_num = argmin(Sign(Random(1) - \frac{\sum_{z=1}^{\beta - |T_{low}| - del\_num} z^2}{\sum_{z=1}^{\beta - |T_{low}|} z^2})) \tag{6}$$

where del_num and z are natural number ranged in $[1, \beta - |T_{low}|]$, and the definition of $Random(1)$ and $Sign(x)$ are same with them in Eq. (4) and Eq. (5). According to this step, the sample in Fig. 2 finally comes to the $NewTree = [1, 0, 1, 1, 1, 0, 0, 1, 0, 0, age = 0]$. Besides, the old tree's age should grown up to 1 in this sample. Note that the "add" operation always selects the highest-quality feature, and the "delete" operation always drops the lowest-quality feature. Our strategy is different from the strategy proposed by Moradi et al. [Moradi and Gholampour (2016)]. Especially when dealing with two cases, $|T_{high}| > \alpha$ and $|T_{low}| < \beta$. In these two cases, neither the number of high-quality features is pull back to α, nor the number of low-quality features is compelled to β. The calculation

of add_num and del_num can be recognized as the probability in considering of feedback theory.

3.4 Fitness function

The definition of the novel fitness function is as follows:

$$Fittness = \mu * Accuracy(Tree) + (1 - \mu) * \frac{|f| - |C|}{|f|} \tag{7}$$

where $Accuracy(Tree)$ means the classification accuracy of the current tree (feature subset) on the classifier. $|C|$ represents the number of features selected in the current tree. $|f|$ is the total number of features of the dataset. The parameter μ is defined to describe the importance of the classification accuracy and the dimensionality reduction ratio of the current feature subset. Eq. (7) aim to find a balance point between the classification accuracy and the dimensionality reduction ratio, since the core purpose of feature selection is to increase the rate of dimensionality reduction ratio on the premise of ensuring the classification accuracy. Generally, accuracy is more important than dimensionality reduction ratio. In the experiments of parameter selection, the best fitness value achieved when μ is set to 0.8. Note that all trees in the forest are assessed by this novel fitness function instead of considering the accuracy only.

3.5 Size limitation

In order to speed up the local seeding process, "life time" and "area limit" is given to control the number of trees in the forest. Those trees who does not match the restrictions will be moved from the forest to a candidate set. In our algorithm, it works like a recycle bin for that it collects those trees with older age or lower fitness value. However, candidate trees are useful in the next step called global seeding. Note that if the tree with highest fitness value reaches "life time", it is age will be reset to 0.

3.6 Global seeding

This step gives the candidate trees a chance of renascence. We select "flood rate" of trees with the lowest fitness from the bottom of the candidate set. After a big rainfall, these weakest trees are carried by the flood to far far away. Their features and ages are completely reversed. For example, if a tree is like $[1, 0, 1, 1, 1, 0, 0, 1, 0, 0, age = 10]$, it will be reversed into $[0, 1, 0, 0, 0, 1, 1, 0, 1, 1, age = 0]$. The flood motivates such a global seeding process, which brings us many strange new trees that seems quite different from those we got from local seeding. These strange new trees will participate in the next iteration and effectively prevent the algorithm falling into local optimal solution.

The whole pseudo code of our FSLSFOA is as follows:

Algorithm 2 FSLSFOA

Input: life time, area limit, flood rate, ε, λ, μ

Output: the best feature set with the highest fitness value

STEP 1 Initialization

use feature subset size determination mechanism to calculate sf which can be affected by ε

sf of features are initialized to 0 and 1 randomly for the "area limit" of trees created

each tree is represented to $(D+1)$ dimension vector (D is features and age is set to 0)

STEP 2 Feature Grouping

compute the importance of each feature and divide the feature set into $Group\_high$ and $Group\_low$

while iterations < maximum iterations **do**

STEP 3 Local Seeding (use λ to empirically control the proportion of selected features)

use local search strategy to create new trees from trees aged 0 (new trees for last iteration)

the old trees' age add 1

STEP 4 Fitness computation

compute fitness value which can be affected by μ

STEP 5 Size Limitation

sort all trees according to fitness value and move low fitness trees into candidate set

set the best tree's age to 0

STEP 6 Global Seeding

select "flood rate" of trees with the lowest fitness from the bottom of the candidate set

reverse the front D features of the tree vectors and set their age to 0

end while

return the best feature set with the highest fitness value

4 Experiment

10 standard datasets along with 3 classical classifiers are selected to conduct experiments. At the same time, our algorithm is compared with other feature selection methods. It includes feature selection using forest optimization algorithm based on contribution degree(FSFOACD) [Ma, Jia, Zhou et al. (2018)], feature selection based on ant colony optimization unsupervised probability algorithm (UPFS) [Dadaneh, Markid and Zakerolhosseini (2016)], feature selection based on supervised rough sets (ANFS) [Luo, Nie, Chang et al. (2018)] and the feature selection of a novel initialization and update mechanism of particle swarm optimization (PSO(4-2)) [Tran, Xue and Zhang (2014b)].

4.1 Data sets

In the experiment, 10 standard datasets from UCI machine learning database are selected to conduct experiments. These include Glass, Wine, Vehicle, Hepatitis, Parkinsons,

Ionosphere, Dermatology, SpamBase, Sonar and Arrhythmia datasets. These datasets cover small, medium and large datasets, and they have been widely used in many research areas of machine learning. Tab. 1 shows the details of the dataset mentioned, such as the number of features, the number of categories, and the number of samples.

In experiments, the average of corresponding features is used to fill missing values. In addition, in real cases, the numerical ranges between eigenvalues vary greatly. Therefore, the characteristics associated with large range values dominate those associated with small range values. In order to overcome this problem, all of the features are normalized.

Table 1: Data sets

Name	Number of features	Number of classes	Number of samples
Glass	9	7	214
Wine	13	3	178
Vehicle	18	4	946
Hepatitis	19	2	155
Parkinsons	23	2	197
Ionosphere	34	2	351
Dermatology	34	6	366
SpamBase	57	2	4601
Sonar	60	2	208
Arrhythmia	279	16	452

4.2 Classifiers

To verify the generality of our proposed algorithm, several classical classifiers are utilized to test the classification effect of the selected feature subset. It includes support vector machine (SVM), decision tree (DT) and k-nearest neighbor (KNN). These classifiers are widely used in the field of data mining. We use the SMO, IBK and J48 algorithms in WEKA to implement the functions of SVM, KNN and DT classifier.

4.3 Evaluation index

In the experiment, classifiers above are able to classify datasets according to the features selected by feature selection algorithm. The accuracy of a classifier on one dataset is the percentage of samples that can be correctly labeled by the classifier. In order to measure the stability, the feature selection algorithm is repeatedly run 10 times in each condition. Immediately behind it, standard deviation is calculated by 10 accuracies of classifier. Moreover, the fitness value who leads in the feature number penalty is utilized inside the model to find a balance between the goal of feature selection and classification. In practice, $Accuracy(Tree)$ in Eq. (7) can be calculated by feeding the classifier with the selected features and the label of the dataset. The fitness value takes the place of accuracy as the internal evaluation index of the model, while external experimental results still use more common accuracy for the help to other researchers.

4.4 Parameter discussion

In this section, a discussion on the parameters of FSLSFOA is conducted. The number of trees in the forest is initialized to 50, which is satisfactory for most datasets we used. The values of "life time", "flood rate" and "area limit" are set to 15, 5 and 50 respectively. "life time" is of great use to control the vitality of the forest. "flood rate" gives a chance to those trees at the bottom of candidate set, but may also bring confusion to the forest. So, it is recommended to set "flood rate" within 10% of "area limit", which directly affects the time cost of our algorithm. There is no doubt that parameters above are greatly affected by the size of actual datasets, but performance discussion on a specific dataset means little in comparison with the average accuracy on all datasets we used.

Next, the baseline settings of ε, λ and μ are 0.5, 0.5 and 1 respectively. Then, discussion on them are given respectively according to the average accuracy of all classifiers on all datasets. Each time, only one parameter is changed from the baseline, namely variable-controlling approach, which ensures the reflection of parameter's influence on average accuracy.

Table 2: Performance evaluation on ε according to average accuracy on all datasets using all classifiers

ε	**Accuracy(%)**	ε	**Accuracy(%)**
0.2	81.32	0.6	83.94
0.3	85.35	0.7	79.86
0.4	**87.86**	0.8	75.42
0.5	86.83		

In this part, the evaluation indicator is the total average of 30 accuracy rates of three classifiers on ten datasets. Firstly, in Tab. 2, ε shows best performance when it takes 0.4. This result shows that only about 40% of features are worth being selected. In another word, feature selection is universally necessary for common datasets. Secondly, the average accuracy peaks at 87.58% when λ is set to 0.6 in Tab. 3. The truth is that the contribution of feature grouping process is very limited. 60% of $Group\_high$ along with 40% of $Group\_low$ rather than 100% selecting from $Group\_high$ achieves the best result. Thirdly, μ is used to adjust composition of fitness value. As is shown in Tab. 4, the best value of μ is 0.8, which indicates that even better average accuracy can be achieved than using accuracy as evaluation index by optimizing the fitness value containing 20% of feature number penalty. What surprises us is that feature number penalty, which aims to reduce the number of features, brings better performance for our algorithm. We speculate that this is probably because those feature sets with the highest training accuracy has been over fitted.

Table 3: Performance evaluation on λ according to average accuracy on all datasets using all classifiers

λ	**Accuracy(%)**	λ	**Accuracy(%)**
0.2	77.38	0.6	**87.58**
0.3	81.44	0.7	87.02
0.4	84.87	0.8	85.83
0.5	86.83		

Table 4: Performance evaluation on μ according to average accuracy on all datasets using all classifiers

μ	**Accuracy(%)**	μ	**Accuracy(%)**
0.1	67.58	0.6	84.41
0.2	72.91	0.7	86.22
0.3	76.48	0.8	**87.51**
0.4	80.18	0.9	87.18
0.5	83.12	1.0	86.83

4.5 Results and comparisons

To verify the generality of the proposed algorithm, different classifiers are utilized to evaluate the performance of the proposed algorithm, including KNN, DT (decision tree) and SVM. In order to obtain steady results, the classification accuracy of feature selection algorithm is measured by 10-fold cross validation. Tabs. 5 to 7 show the mean value and standard deviation of the classification accuracy of different classifiers SMO, IBK and J48 with different feature selection algorithms UPFS, ANFS, PSO(4-2), FSFOACD and FSLSFOA over the 10 datasets. Note that the best results for our FSLSFOA in the parameter discussion above have been used in this part from beginning to the end.

From the results shown in Tab. 5 and Fig. 3, along with KNN classifier, the proposed FSLSFOA algorithm performs best on most datasets compared with other feature selection algorithms. Only on the dataset Vehicle, FSLSFOA, whose classification accuracy is 81.21%, is second to PSO(4-2) algorithm, whose classification accuracy is 84.31%. On the other hand, the standard deviation of our algorithm is the lowest in all datasets, which improves that our algorithm is very stable. Especially on the datasets Glass, Hepatitis, Sonar and Arrhythmia, our FSLSFOA has a significant improvement in classification accuracy compared with other feature selection algorithms.

Table 5: Performance of FSLSFOA and other feature selection algorithm using KNN classifier

Dataset	UPFS	ANFS	PSO(4-2)	FSFOACD	FSLSFOA
Glass	68.76 ± 2.37	72.87 ± 2.53	73.21 ± 2.94	76.30 ± 0.07	**78.16** ± 0.24
Wine	96.12 ± 0.82	96.41 ± 1.17	93.93 ± 2.87	98.91 ± 0.003	**98.97** ± 0.04
Vehicle	65.02 ± 2.42	76.33 ± 1.61	**84.31** ± 3.63	78.59 ± 0.09	81.21 ± 0.31
Hepatitis	78.71 ± 0.12	82.12 ± 0.94	88.34 ± 2.85	89.20 ± 0.05	**91.48** ± 1.51
Parkinsons	91.26 ± 1.07	88.94 ± 1.28	89.02 ± 1.98	88.92 ± 0.19	**92.72** ± 0.47
Ionosphere	89.48 ± 1.67	90.85 ± 1.64	87.21 ± 2.74	93.43 ± 0.07	**94.43** ± 0.39
Dermatology	97.56 ± 0.98	96.49 ± 1.07	98.01 ± 1.85	99.19 ± 0.18	**99.25** ± 0.22
SpamBase	89.66 ± 1.23	92.76 ± 0.92	91.57 ± 1.66	93.02 ± 0.06	**93.96** ± 0.55
Sonar	85.12 ± 2.58	86.87 ± 1.14	83.24 ± 3.98	93.44 ± 0.12	**94.22** ± 0.82
Arrhythmia	54.12 ± 1.43	55.83 ± 2.62	48.14 ± 5.82	61.41 ± 0.47	**63.66** ± 1.58
Average	81.58 ± 1.47	83.95 ± 1.49	83.63 ± 3.03	87.24 ± 0.13	**88.81** ± 0.61

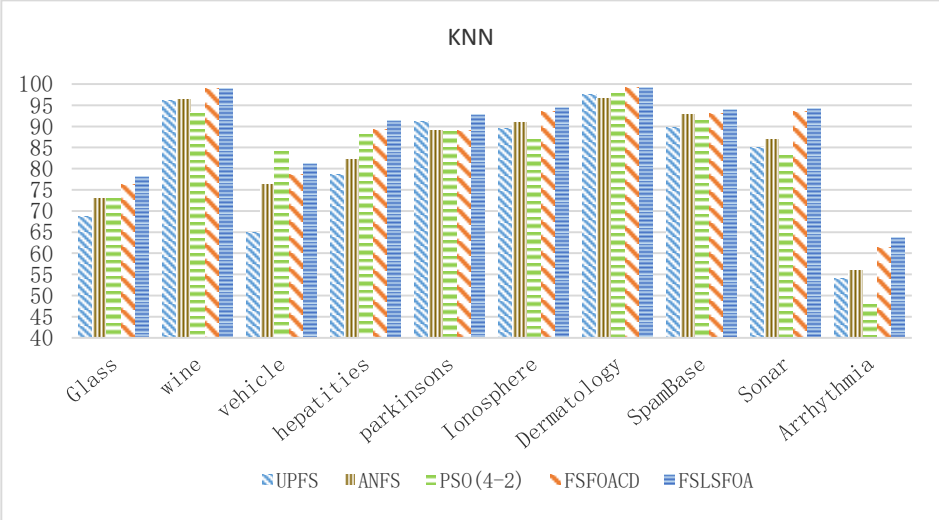


Figure 3: Performance of FSLSFOA and other feature selection algorithm using KNN classifier

From the results shown in Tab. 6 and Fig. 4, with the SVM classifier, the proposed FSLSFOA algorithm has the highest average classification accuracy compared with other feature selection algorithms on all datasets. Particularly on the datasets Wine, when the accuracy of FSFOACD is 99.17% , the accuracy of FSLSFOA achieves 99.09%. We infer that this is because of the comparatively high standard variance of FSLSFOA and the high accuracy of the dataset. For the rest of the datasets, our local search strategy and global seeding process have played a great role in improving the feature selection.

Table 6: Performance of FSLSFOA and other feature selection algorithm using SVM classifier

Dataset	UPFS	ANFS	PSO(4-2)	FSFOACD	FSLSFOA
Glass	61.70 ± 2.22	68.43 ± 1.85	70.28 ± 1.98	70.56 ± 0.01	**72.54** ± 0.22
Wine	93.29 ± 1.22	96.32 ± 1.16	97.03 ± 1.37	**99.17** ± 0.02	99.09 ± 0.12
Vehicle	63.01 ± 2.67	77.57 ± 1.88	79.63 ± 2.45	80.16 ± 1.02	**82.72** ± 1.67
Hepatitis	84.46 ± 1.34	83.11 ± 2.23	82.97 ± 2.01	85.26 ± 0.14	**85.36** ± 0.91
Parkinsons	87.21 ± 0.01	90.69 ± 0.73	88.76 ± 1.34	89.56 ± 0.01	**91.77** ± 0.53
Ionosphere	86.91 ± 1.01	89.97 ± 0.68	93.28 ± 0.78	95.87 ± 0.07	**96.33** ± 0.23
Dermatology	97.70 ± 0.46	97.25 ± 1.54	96.91 ± 1.92	98.63 ± 0.04	**98.95** ± 0.34
SpamBase	88.21 ± 0.24	89.84 ± 0.43	90.49 ± 0.94	92.69 ± 0.02	**93.64** ± 0.62
Sonar	80.29 ± 0.32	82.58 ± 0.82	85.47 ± 2.31	88.46 ± 0.03	**89.55** ± 1.59
Arrhythmia	59.93 ± 1.02	62.44 ± 1.56	65.56 ± 1.08	65.21 ± 0.28	**67.58** ± 1.43
Average	80.27 ± 1.05	83.82 ± 1.29	85.03 ± 1.62	86.56 ± 0.17	**87.75** ± 0.77

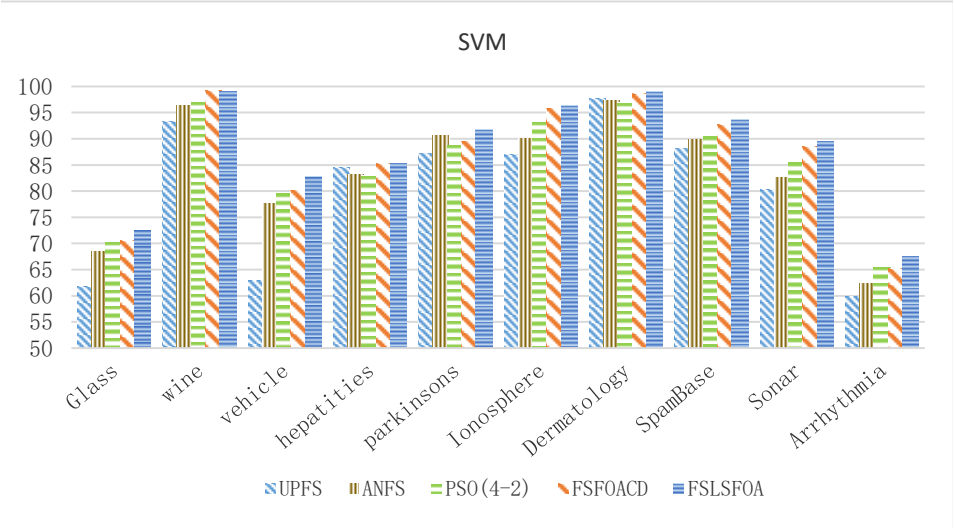


Figure 4: Performance of FSLSFOA and other feature selection algorithm using SVM classifier

As can be seen from Tab. 7 and Fig. 5, cooperating with DT classifier, FSLSFOA also has the highest classification accuracy on most datasets compared with other algorithms. This is benefit from the local search strategy. In the stage of local seeding and global seeding, the local search strategy guides the forest optimization algorithm to select more features with high class correlation and low redundancy, while eliminating low class correlation and high redundancy features. Thus it guarantees the quality of the feature subset in the forest. The state-of-art probability model in our local search strategy allows more variance for each tree, which allows more possibility for the limited trees to find the optimal solution within a limited number of iterations.

Table 7: Performance of FSLSFOA and other feature selection algorithm using DT classifier

Dataset	UPFS	ANFS	PSO(4-2)	FSFOACD	FSLSFOA
Glass	71.15 ± 2.34	77.26 ± 1.73	80.56 ± 3.01	79.07 ± 0.07	**81.62** ± 0.17
Wine	94.09 ± 1.75	96.34 ± 1.54	95.87 ± 2.61	97.77 ± 0.03	**98.53** ± 0.35
Vehicle	69.02 ± 2.57	77.61 ± 2.09	**80.14** ± 2.14	76.01 ± 0.04	78.82 ± 0.44
Hepatitis	86.51 ± 0.86	86.73 ± 1.4	84.93 ± 1.45	89.11 ± 0.01	**91.91** ± 1.21
Parkinsons	86.95 ± 2.55	87.85 ± 2.28	86.88 ± 2.67	**90.89** ± 0.18	89.86 ± 0.88
Ionosphere	90.16 ± 2.90	92.51 ± 1.62	94.26 ± 2.01	95.45 ± 0.09	**97.47** ± 1.30
Dermatology	95.13 ± 1.67	97.12 ± 1.33	95.98 ± 1.88	98.09 ± 0.12	**98.75** ± 0.73
SpamBase	88.95 ± 2.48	90.96 ± 1.81	91.79 ± 1.43	93.58 ± 0.17	**94.93** ± 1.54
Sonar	84.09 ± 1.94	82.42 ± 0.87	83.87 ± 1.92	84.20 ± 0.02	**86.54** ± 0.42
Arrhythmia	60.40 ± 1.28	62.71 ± 1.56	59.45 ± 0.98	61.56 ± 0.68	**62.76** ± 0.84
Average	82.64 ± 2.03	85.15 ± 1.62	85.37 ± 2.01	86.27 ± 0.14	**88.12** ± 0.79

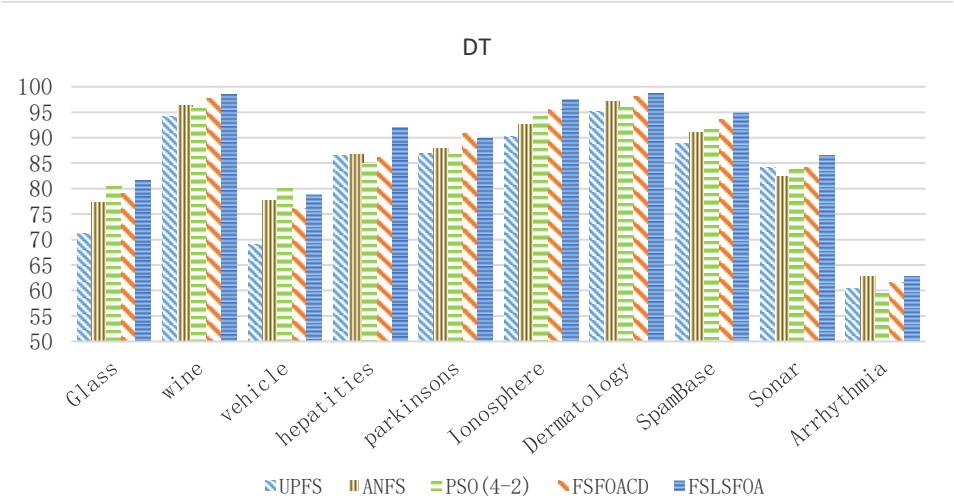
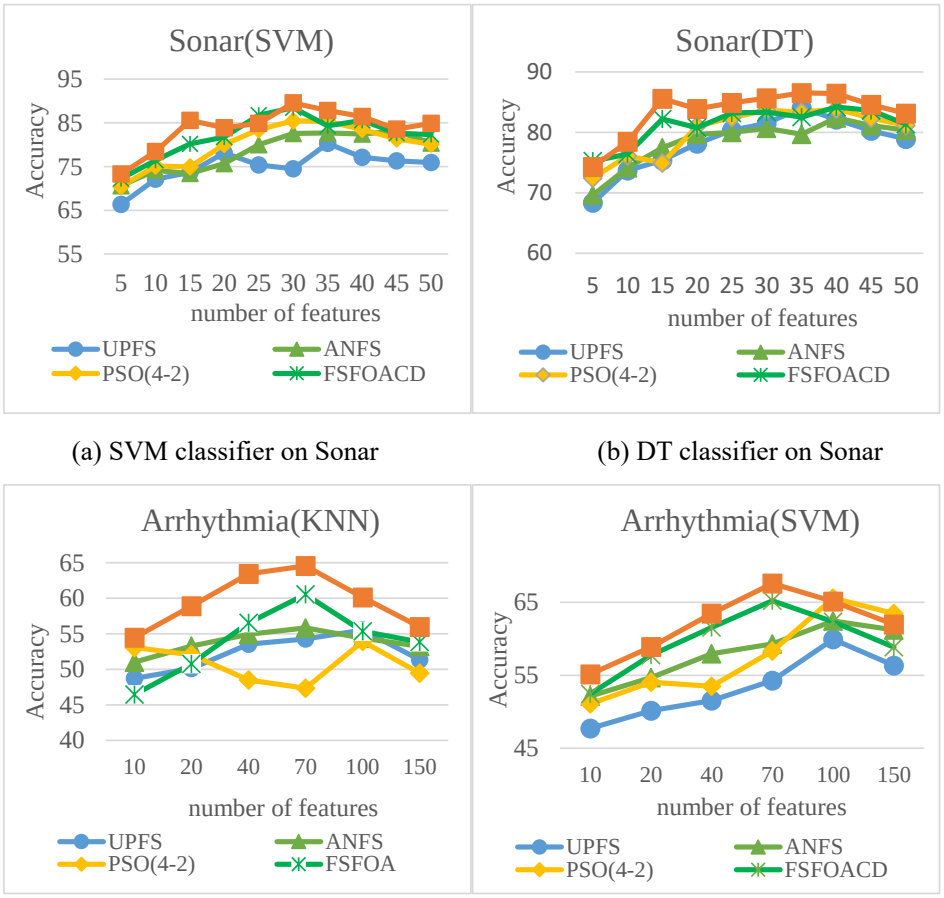


Figure 5: Performance of FSLSFOA and other feature selection algorithm using DT classifier

In addition, several experiments have been conducted to compare the classification accuracy of the proposed FSLSFOA with other feature selection methods based on the number of selected features. Fig. 6 and Tab. 8 show the results on datasets Sonar and Arrhythmia. In each graph, the X-axis represents the subset size of the selected feature, while the Y-axis represents the accuracy. In Figs. 6(a), 6(b) and Tabs. 8(a), 8(b), the accuracy of FSLSFOA based on SVM and DT classifier is obviously better than other feature selection algorithms.

Figs. 6(c), 6(d) and Tabs. 8(c), 8(d) show similar results. Tab. 8(c) shows that when the number of selected features in Arrhythmia dataset is set to 70, the accuracy of FSLSFOA is 63.66%. In this case, the accuracy of UPFS, ANFS, PSO (4-2) and FSFOACD is 54.12%, 55.83%, 48.14% and 61.41%, respectively. As shown in Fig. 6(d), FSLSFOA outperforms

over other feature selection algorithms in the range of 10-70 features, but is defeated by PSO (4-2) when the feature number go over 100. This deficiency is worthy of our further research.



(a) SVM classifier on Sonar (b) DT classifier on Sonar

(c) KNN classifier on Arrhythmia (d) SVM classifier on Arrhythmia

Figure 6: Comparison of classification accuracy of feature subsets of different sizes on Sonar and Arrhythmia datasets

Table 8: Comparison of classification accuracy of feature subsets of different sizes on Sonar and Arrhythmia datasets

(a) Sonar(SVM)

Feature number	UPFS	ANFS	PSO(4-2)	FSFOACD	FSLSFOA
5	66.34	63.61	70.44	60.23	73.28
10	72.12	65.14	75.09	63.44	78.45
15	73.67	67.45	74.94	65.21	85.59
20	78.13	75.67	80.03	67.78	83.89
25	75.34	80.01	83.45	69.78	84.91
30	74.48	81.68	84.57	68.33	91.67
35	80.29	82.31	85.47	67.33	87.81
40	77.14	82.58	83.48	65.54	86.45
45	76.29	81.94	81.34	66.64	83.56
50	75.93	80.38	80.12	64.34	84.89

(b) Sonar(DT)

Feature number	UPFS	ANFS	PSO(4-2)	FSFOACD	FSLSFOA
5	68.34	69.61	72.44	75.23	74.28
10	73.72	74.14	76.09	76.44	78.45
15	75.37	77.45	74.94	82.21	85.59
20	78.13	79.67	81.03	80.78	83.89
25	80.34	80.01	82.45	83.2	84.91
30	81.48	80.68	83.65	83.33	85.67
35	84.09	79.67	83.35	82.53	86.54
40	82.14	82.42	83.87	84.2	86.45
45	80.29	81.11	82.34	83.64	84.56
50	78.93	80.38	81.12	81.34	83.12

(c) Arrhythmia(KNN)

Feature number	UPFS	ANFS	PSO(4-2)	FSFOACD	FSLSFOA
10	48.72	51.01	53.09	46.44	54.45
20	50.13	53.29	52.03	50.78	58.89
40	53.54	54.94	48.48	56.54	63.45
70	54.29	55.83	47.34	60.54	64.56
100	55.53	54.38	53.92	55.34	60.12
150	51.33	53.23	49.45	53.84	55.97

(d) Arrhythmia(SVM)

Feature number	UPFS	ANFS	PSO(4-2)	FSFOACD	FSLSFOA
10	47.72	52.14	51.09	52.44	55.15
20	50.13	54.67	54.03	57.78	58.89
40	51.54	57.94	53.48	61.54	63.45
70	54.29	59.31	58.34	65.21	67.58
100	59.93	62.44	65.56	62.34	65.12
150	56.33	61.23	63.45	58.84	61.97

Based on the average accuracy in Tabs. 5 to 7, the average classification accuracy can be illustrated in Fig. 7. Obviously, our proposed FSLSFOA achieves the best results in combination with three classifiers. Among the three classifiers, DT achieves the best performance in combination with different feature selection methods. However, FSLSFOA+KNN get an accuracy of 88.81%, which is the better than SVM and DT.

To sum up the unsatisfactory results, FSLSFOA performs not very well on the Vehicle

and Parkinsons. Besides, its standard deviation on accuracy is worth that FSFOACD, which is caused by the randomness in local search strategy. Eqs. (4)-(6) not only bring these randomness, but also improve the accuracy. After weighing the pros and cons, this mechanism is preserved.

In our future work, FOA based feature selection methods is still the main task. More deep learning methods can be applied to further improve the performance of our work. The proposed methods should be compared with more recent study and methods on more datasets. In order to make the research results more convincing, it is also necessary to apply the model to some areas of urgent need. Abdel-Basset M will be our guide i the future work. The works of his team is worth study to improve our model.

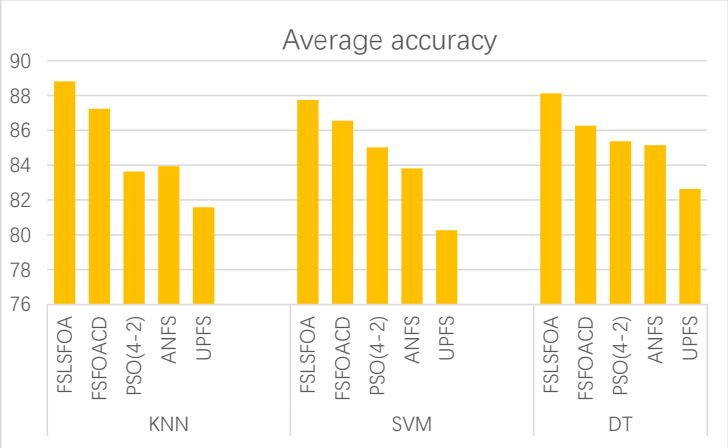


Figure 7: Average classification accuracy of feature selection methods on all datasets

5 Conclusion

This article proposes a feature selection algorithm with local search strategy based on the forest optimization algorithm, namely FSLSFOA. Our local search strategy guides the forest optimization algorithm to select features with higher class correlation and lower redundancy, and remove features with lower class correlation and higher redundancy. While searching for the optimal solution, this strategy guarantees the quality of the feature subset in the forest. Next, the fitness function is improved, which not only consider the classification accuracy, but also consider the size of the feature subset, thus to get an optimal solution with smaller feature subset. To avoid falling into local optimum, a new global seeding method is attempted, which picks up trees on the bottom of candidate set and gives the algorithm more possibilities. Finally, 10 datasets with different attribute benchmarks are selected to verify the effectiveness of the proposed algorithm. The experiment is implemented in terms of classification accuracy and the size of the selected feature subset. Most of the results are superior to previous feature selection algorithms we selected as the comparative method.

Acknowledgement: This work was supported in part by National Science Foundation of

China (Nos. U1736105, 61572259, 41942017). The authors extend their appreciation to the Deanship of Scientific Research at King Saud University for funding this work through research group no. RGP-VPP-264.

References

Aburomman, A. A.; Reaz, M. B. I. (2016): A novel svm-knn-pso ensemble method for intrusion detection system. *Applied Soft Computing*, vol. 38, no. C, pp. 360-372.

Ambusaidi, M.; He, X.; Nanda, P.; Tan, Z. (2016): Building an intrusion detection system using a filter-based feature selection algorithm. *IEEE Transactions on Computers*, vol. 65, no. 10, pp. 2986-2998.

Anthimopoulos, M.; Christodoulidis, S.; Ebner, L.; Christe, A.; Mougiakakou, S. (2016): Lung pattern classification for interstitial lung diseases using a deep convolutional neural network. *IEEE Transactions on Medical Imaging*, vol. 35, no. 5, pp. 1207-1216.

Benmahamed, Y.; Teguar, M.; Boubakeur, A. (2018): Application of SVM and KNN to duval pentagon 1 for transformer oil diagnosis. *IEEE Transactions on Dielectrics & Electrical Insulation*, vol. 24, no. 6, pp. 3443-3451.

Coelho, F.; Braga, A. P.; Verleysen, M. (2010): Multi-objective semi-supervised feature selection and model selection based on pearson correlation coefficient. *Lecture Notes in Computer Science*, vol. 6419, pp. 509-516.

Dadaneh, B. Z.; Markid, H. Y.; Zakerolhosseini, A. (2016): Unsupervised probabilistic feature selection using ant colony optimization. *Expert Systems with Applications*, vol. 53, no. C, pp. 27-42.

Fernández-García, A. J.; Iribarne, L.; Corral, A.; Criado, J. (2018): A comparison of feature selection methods to optimize predictive models based on decision forest algorithms for academic data analysis. In Rocha, Á.; Adeli, H.; Reis, L. P.; Costanzo, S.(Eds): *Trends and Advances in Information Systems and Technologies*, pp. 338–347, Cham. Springer International Publishing.

Ghaemi, M.; Feizi-Derakhshi, M. R. (2014): Forest optimization algorithm. *Expert Systems with Applications*, vol. 41, no. 15, pp. 6676-6687.

Ghaemi, M.; Feizi-Derakhshi, M. R. (2016): Feature selection using forest optimization algorithm. *Pattern Recognition*, vol. 60, pp. 121-129.

Gorelick, R.; Bertram, S. M. (2010): Multi-way multi-group segregation and diversity indices. *PLoS One*, vol. 5, no. 6. e10912.

Guyon, I. (2003): An introduction to variable and feature selection. *JMLR*, pp. 1157-1182.

Hancer, E.; Xue, B.; Zhang, M.; Karaboga, D.; Akay, B. (2017): Pareto front feature selection based on artificial bee colony optimization. *Information Sciences*, vol. 422.

Hou, C.; Nie, F.; Li, X.; Yi, D.; Wu, Y. (2014): Joint embedding learning and sparse regression: A framework for unsupervised feature selection. *IEEE Transactions on Cybernetics*, vol. 44, no. 6, pp. 793-804.

Keskar, N. S.; Mudigere, D.; Nocedal, J.; Smelyanskiy, M.; Tang, P. T. P. (2016): On large-batch training for deep learning: generalization gap and sharp minima. arxiv:1609.04836.

Luo, M.; Nie, F.; Chang, X.; Yang, Y.; Hauptmann, A. G. et al. (2018): Adaptive unsupervised feature selection with structure regularization. *IEEE Transactions on Neural Networks & Learning Systems*, vol. 29, no. 4, pp. 944-956.

Lv, Y.; Ma, T.; Tang, M.; Cao, J.; Tian, Y.; fAdaptive unsupervised feature selection with structure regularizationothers (2016): An efficient and scalable density-based clustering algorithm for datasets with complex structures. *Neurocomputing*, vol. 171, pp. 9-22.

Ma, T.; Jia, D.; Zhou, H.; Xue, Y.; Cao, J. (2018): Feature selection using forest optimization algorithm based on contribution degree. *Intelligent Data Analysis*, vol. 22, no. 6, pp. 1189-1207.

Ma, T.; Shao, W.; Hao, Y.; Cao, J. (2018): Graph classification based on graph set reconstruction and graph kernel feature reduction. *Neurocomputing*, vol. 296, pp. 33-45.

Melian, B.; Verdegay, J. (2011): *Using Fuzzy Numbers in Network Design Optimization Problems*. IEEE Press.

Migdady, H. M. (2013): A features extraction wrapper method for neural networks, with application to data mining and machine learning. *Dissertations & Theses - Gradworks*.

Mitra, P.; Murthy, C. A.; Pal, S. K. (2002): Unsupervised feature selection using feature similarity. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 3, pp. 301-312.

Moradi, P.; Gholampour, M. (2016): A hybrid particle swarm optimization for feature subset selection by integrating a novel local search strategy. *Applied Soft Computing*, vol. 43, no. C, pp. 117-130.

Mu, Y.; Liu, X.; Wang, L. (2017): A pearson correlation coefficient based decision tree and its parallel implementation. *Information Sciences*, vol. 435, pp. 40-58.

Paisitkriangkrai, S.; Shen, C.; Hengel, A. V. D. (2016): Pedestrian detection with spatially pooled features and structured ensemble learning. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, vol. 38, no. 6, pp. 1243.

Rodriguezgaliano, V. F.; Luqueespinar, J. A.; Chicaolmo, M.; Mendes, M. P. (2018): Feature selection approaches for predictive modelling of groundwater nitrate pollution: An evaluation of filters, embedded and wrapper methods. *Science of the Total Environment*, vol. 624, pp. 661-672.

Rong, H.; Ma, T.; Cao, J.; Tian, Y.; Al-Dhelaan, A.; Al-Rodhaan, M. (2019): Deep rolling: a novel emotion prediction model for a multi-participant communication context. *Information Sciences*, vol. 488, pp. 158-180.

Selvam, K.; Kumar, D. M. V.; Siripuram, R. (2017): Distributed generation planning using peer enhanced multi-objective teaching-learning based optimization in distribution networks. *Journal of the Institution of Engineers*, vol. 98, no. 2, pp. 1-9.

Shirani, H.; Habibi, M.; Besalatpour, A. A.; Esfandiarpour, I. (2015): Determining the features influencing physical quality of calcareous soils in a semiarid region of iran using a hybrid pso-dt algorithm. *Geoderma*, vol. 259-260, pp. 1-11.

Stuart, D. (2016): The data revolution: big data, open data, data infrastructures and their consequences. *Media Culture & Society*, vol. 37, no. 7, pp. 1110-1111.

Sun, Y.; Goodison, S.; Li, J.; Liu, L.; Farmerie, W. (2007): Improved breast cancer prognosis through the combination of clinical and genetic markers. *Bioinformatics*, vol. 23, no. 1, pp. 30.

Tran, B.; Xue, B.; Zhang, M. (2014): Improved pso for feature selection on high-dimensional datasets. In Dick, G.; Browne, W. N.; Whigham, P.; Zhang, M.; Bui, L. T.; Ishibuchi, H.; Jin, Y.; Li, X.; Shi, Y.; Singh, P.; Tan, K. C.; Tang, K.(Eds): *Simulated Evolution and Learning*, pp. 503–515, Cham. Springer International Publishing.

Tran, B.; Xue, B.; Zhang, M. (2014): Overview of particle swarm optimisation for feature selection in classification. *Asia-Pacific Conference on Simulated Evolution and Learning*, pp. 605-617.

Tran, B.; Zhang, M.; Xue, B. (2016): A pso based hybrid feature selection algorithm for high-dimensional classification. *IEEE Congress on Evolutionary Computation*, pp. 3801-3808.

Tzanis, G. (2017): Biological and medical big data mining. *International Journal of Knowledge Discovery in Bioinformatics*, vol. 4, no. 1, pp. 42-56.

Vandersypen, L.; Leeuwenhoek, A. V. (2017): 1.4 quantum computing - the next challenge in circuit and system design. In *Solid-State Circuits Conference*, pp. 24-29.

Wipf, D.; Nagarajan, S. (2016): Iterative reweighted and methods for finding sparse solutions. *IEEE Journal of Selected Topics in Signal Processing*, vol. 4, no. 2, pp. 317-329.

Xue, B.; Zhang, M.; Browne, W. N. (2013): Particle swarm optimization for feature selection in classification: A multi-objective approach. *IEEE Transactions on Cybernetics*, vol. 43, no. 6, pp. 1656-1671.

Xue, B.; Zhang, M.; Browne, W. N.; Yao, X. (2016): A survey on evolutionary computation approaches to feature selection. *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 4, pp. 606-626.

Yang, P.; Liu, W.; Zhou, B. B.; Chawla, S.; Zomaya, A. Y. (2013): Ensemble-based wrapper methods for feature selection and class imbalance learning. *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pp. 544-555.

Zawbaa, H. M.; Emary, E.; Hassanien, A. E.; Parv, B. (2016): A wrapper approach for feature selection based on swarm optimization algorithm inspired from the behavior of social-spiders. *Soft Computing and Pattern Recognition*, pp. 191-199.

Zhang, Y.; Ma, T.; Cao, J.; Shen, J.; Tang, M. et al. (2015): KDVEM : a k-degree anonymity with vertex and edge modification algorithm. *Computing*, vol. 97, no. 12, pp. 1165-1184.

Zhang, Y.; Wang, S.; Sui, Y.; Yang, M.; Liu, B. et al. (2017): Multivariate approach for alzheimer's disease detection using stationary wavelet entropy and predator-prey particle swarm optimization. *Journal of Alzheimers Disease JAD*, no. 13, pp. 1-15.

Zhu, C.; Xu, J.; Chen, C. H.; Lee, L. H.; Hu, J. Q. (2016): Balancing search and estimation in random search based stochastic simulation optimization. *IEEE Transactions on Automatic Control*, vol. 61, no. 11, pp. 3593-3598.