



An Improved Particle Swarm Optimization for Feature Selection

Yuanning Liu^{1,2}, Gang Wang^{1,2}, Huiling Chen^{1,2}, Hao Dong^{1,2}, Xiaodong Zhu^{1,2}, Sujing Wang^{1,2}

1. College of Computer Science and Technology, Jilin University, Changchun 130012, P. R. China

2. Key Laboratory of Symbolic Computation and Knowledge Engineering of Ministry of Education, Jilin University, Changchun 130012, P. R. China

Abstract

Particle Swarm Optimization (PSO) is a popular and bionic algorithm based on the social behavior associated with bird flocking for optimization problems. To maintain the diversity of swarms, a few studies of multi-swarm strategy have been reported. However, the competition among swarms, reservation or destruction of a swarm, has not been considered further. In this paper, we formulate four rules by introducing the mechanism for survival of the fittest, which simulates the competition among the swarms. Based on the mechanism, we design a modified Multi-Swarm PSO (MSPSO) to solve discrete problems, which consists of a number of sub-swarms and a multi-swarm scheduler that can monitor and control each sub-swarm using the rules. To further settle the feature selection problems, we propose an Improved Feature Selection (IFS) method by integrating MSPSO, Support Vector Machines (SVM) with F-score method. The IFS method aims to achieve higher generalization capability through performing kernel parameter optimization and feature selection simultaneously. The performance of the proposed method is compared with that of the standard PSO based, Genetic Algorithm (GA) based and the grid search based methods on 10 benchmark datasets, taken from UCI machine learning and StatLog databases. The numerical results and statistical analysis show that the proposed IFS method performs significantly better than the other three methods in terms of prediction accuracy with smaller subset of features.

Keywords: particle swarm optimization, feature selection, data mining, support vector machines

Copyright © 2011, Jilin University. Published by Elsevier Limited and Science Press. All rights reserved.

doi: 10.1016/S1672-6529(11)60020-6

1 Introduction

Feature selection is one of the most important factors which can influence the classification accuracy rate. If the dataset contains a number of features, the dimension of the space will be large and non-clean, degrading the classification accuracy rate. An efficient and robust feature selection method can eliminate noisy, irrelevant and redundant data^[1].

Feature subset selection algorithms can be categorized into two types: filter algorithms and wrapper algorithms. Filter algorithms select the feature subset before the application of any classification algorithm, and remove the less important features from the subset. Wrapper methods define the learning algorithm, the performance criteria and the search strategy. The learning algorithm searches for the subset using the training data and the performance of the current subset.

Particle Swarm Optimization (PSO) was motivated

from the simulation of simplified social behavior of bird flocking, firstly developed by Kennedy and Eberhart^[2–3]. It is easy to implement with few parameters, and it is widely used to solve the optimization problems, as well as feature selection problem^[4–5]. Various attempts have been made to improve the performance of standard PSO in recent years. However, few studies have put emphasis on researching into multi-swarm strategy. Usually, the PSO-based algorithms only have one swarm that contains a number of particles. The PSO-based algorithms using multi-swarm strategy have more exploration and exploitation abilities due to the fact that different swarms have the possibility to explore different parts of the solution space^[6]. On the other hand, standard PSO converges over time, thereby losing diversity, and thus their ability to quickly react to a peak's move. The multi-swarm PSO can sustain the diversity of swarms, and ensure its adaptability, thereby improving the performance of PSO.

Corresponding author: Xiaodong Zhu

E-mail: zhuxiaodong.jlu@gmail.com

Blackwell and Branke^[7] split the population of particles into a set of interacting swarms. They used a simple competition mechanism among swarms that are close to each other. The winner is the swarm with the best function value at its swarm attractor. The loser is expelled and reinitialized in the search space, otherwise the winner remains. Parrott and Li^[8] divided the swarm population into species subpopulations based on their similarity. Additional duplicated particles are removed when particles are identified as having the same fitness with the species seed within the same species. After destroying the duplicated ones, the new particles are added randomly until its size is resumed to its initial size. Niu *et al.*^[9] proposed Multi-swarm Cooperative Particle Swarm Optimizer (MCPSO) based on a master-slave model, in which a population consists of one master swarm and several slave swarms. MCPSO is based on an antagonistic scenario, where the master swarm enhances its particles by a series of competitions with the slave swarms. The master swarm enhances its particles based on direct competition with the slave swarms, and the most fitted particles in all the swarms possess the opportunity to guide the flight direction of the particles in the master swarm.

However, the studies mentioned above have only solved the traditional optimization problems, namely continuous parameter optimization. Our proposed Multi-Swarm Particle Swarm Optimization (MSPSO) can not only solve the continuous parameter problems but also the discrete problems. Moreover, to maintain the diversity of swarms, they do not change the number of particles, as well as the number of swarms, thereby ignoring the competition among the swarms. In this paper, we propose MSPSO algorithm based on a modified multi-swarm PSO through introducing the mechanism for survival of the fittest to describe the competition among the swarms. Four rules are designed according to the mechanism, in which the number of sub-swarms is allowed to reduce during the iterations, namely, that some of the sub-swarms are destroyed during the iterations, and the destroyed sub-swarms can not be reconstructed any more.

To the best of our knowledge, this is the first paper to apply multi-swarm PSO to feature selection problem. The main innovations in this paper are described as follows:

(1) A MSPSO algorithm was proposed, which con-

sists of a number of sub-swarms and a scheduling module. The survival of the fittest is introduced to decide whether a sub-swarm should be destroyed or reserved. To achieve that goal, 4 rules are designed. The scheduling module monitors and controls each sub-swarm according to the rule during the iterations.

(2) The F-score^[10], which can calculate the score of each feature, was introduced to evaluate the results of the feature selection. The objective function is designed according to classification accuracy rate and the feature scores.

(3) An Improved Feature Selection (IFS) method was proposed, which consists of two stages. In the first stage, both the Support Vector Machines (SVM) parameter optimization and the feature selection are dynamically executed by MSPSO. In the second stage, SVM model performs the classification tasks using these optimal values and selected features via 10-fold cross validation.

The remainder of this paper is organized as follows. Section 2 reviews basic principles of PSO and SVM. Section 3 describes the objective function, multi-swarm scheduling module and IFS approach in detail. Section 4 presents the experimental results on 10 benchmark data sets. Finally, section 5 summarizes the conclusion.

2 Basic principles

2.1 Particle swarm optimization

PSO originated from the simulation of social behavior of birds in a flock^[2-3]. In PSO, each particle flies in the search space with a velocity adjusted by its own flying memory and its companion's flying experience. Each particle has its objective function value which is decided by a fitness function:

$$v_{id}^t = w \times v_{id}^{t-1} + c_1 \times r_1 (p_{id}^t - x_{id}^t) + c_2 \times r_2 (p_{gd}^t - x_{id}^t), \quad (1)$$

where i represents the i th particle and d is the dimension of the solution space, c_1 denotes the cognition learning factor, and c_2 indicates the social learning factor, r_1 and r_2 are random numbers uniformly distributed in $(0,1)$, p_{id}^t and p_{gd}^t stand for the position with the best fitness found so far for the i th particle and the best position in the neighborhood, v_{id}^t and v_{id}^{t-1} are the velocities at time t and time $t-1$, respectively, and x_{id}^t is the position of i th particle at time t . Each particle then moves to a new potential solution based on the following equation:

$$x_{id}^{t+1} = x_{id}^t + v_{id}^t, \quad d = 1, 2, \dots, D, \quad (2)$$

Kennedy and Eberhart^[11] proposed a binary PSO in which a particle moves in a state space restricted to 0 and 1 on each dimension, in terms of the changes in probabilities that a bit will be in one state or the other:

$$x_{id} = \begin{cases} 1, & rand() < S(v_{i,d}) \\ 0 \end{cases}, \quad (3)$$

$$S(v) = \frac{1}{1 + e^{-v}}. \quad (4)$$

The function $S(v)$ is a sigmoid limiting transformation and $rand()$ is a random number selected from a uniform distribution in $[0.0, 1.0]$.

2.2 Support vector machines

SVM is specifically designed for two-class problems^[12–13]. Given a training set of instance-label pairs (x_i, y_i) , $i = 1, 2, \dots, m$, where x_i belongs to $\mathbf{R}^n$ and y_i belongs to $\{+1, -1\}$, the generalized linear SVM finds an optimal separating value $f(x) = (w \times x) + b$. The classifier is:

$$f(x) = \text{sgn} \left\{ \sum_{i=1}^n a_i y_i (x_i \times x) + b \right\}. \quad (5)$$

For the non-linear case, SVM will map the data in a lower dimensional space into a higher-dimensional space through kernel trick. The classifier is:

$$f(x) = \text{sgn} \left\{ \sum_{i=1}^n a_i y_i K(x_i \times x) + b \right\}, \quad (6)$$

where $\text{sgn}\{\}$ is the sign function, a_i is Lagrange multiplier, x_i is a training sample, x is a sample to be classified, $K(x_i \times x)$ is the kernel function. Example kernel function includes polynomial function, linear function, and Radial Basis Function (RBF). In this work, we investigated the RBF kernel function.

3 IFS approach

We have proposed the IFS approach, which combines the parameter optimization and the feature selection, in order to obtain the higher classification accuracy rate. A modified PSO algorithm named MSPSO is proposed, which holds a number of sub-swarms scheduled by the multi-swarm scheduling module. The multi-swarm scheduling module monitors all the sub-swarms, and gathers the results from the sub-swarms.

The storage of MSPSO is shown in Fig. 1. The

SVM parameters, feature values and system parameters are described in detail. We modify the PSO to solve discrete problem according to Ref. [11].

The proposed method consists of two stages. In the first stage, both the SVM parameter optimization and the feature selection are dynamically executed by MSPSO. In the second stage, SVM model performs the classification tasks using these optimal values and selected feature subsets via 10-fold cross validation.

An efficient objective function is designed according to classification accuracy rate and F-score. The objective function consists of two parts: one is classification accuracy rate and the other is the feature score. Both of them are summed into one single objective function by linear weighting. The two weights are θ_a and θ_b , and each controls the weight of the specific part.

3.1 Classification accuracy

The classification accuracy for the dataset was measured according to following equation:

$$\begin{cases} accuracy(N) = \frac{\sum_{i=1}^{|N|} assess(n_i)}{|N|}, n_i \in N \\ assess(n) = \begin{cases} 1 & \text{if classify}(n) = nc \\ 0 & \text{otherwise} \end{cases} \end{cases}, \quad (7)$$

where N is the set of data items to be classified (the test set), $n \in N$, nc is the class of the item n , and $\text{classify}(n)$ returns the classification accuracy rates of n by IFS.

3.2 F-score

F-score is a simple technique which measures the discrimination of two sets of real numbers. Given training vectors X_k , $k = 1, 2, \dots, m$, if the number of positive and negative instances are n_+ and n_- , respectively, then the F-score of the i th feature is defined as follows^[10]:

$$F(i) = \frac{(\bar{x}_i^{(+)} - \bar{x}_i)^2 + (\bar{x}_i^{(-)} - \bar{x}_i)^2}{\frac{1}{n_+ - 1} \sum_{k=1}^{n_+} (\bar{x}_{k,i}^{(+)} - \bar{x}_i^{(+)})^2 + \frac{1}{n_- - 1} \sum_{k=1}^{n_-} (x_{k,i}^{(-)} - \bar{x}_i^{(-)})^2}, \quad (8)$$

where $\bar{x}_i$, $\bar{x}_i^{(+)}$, $\bar{x}_i^{(-)}$ are the averages of the i th feature of the whole, positive, and negative datasets, respectively. $\bar{x}_{k,i}^{(+)}$ is the i th feature of the k th positive instance, and $x_{k,i}^{(-)}$ is the i th feature of the k th negative instance. The numerator shows the discrimination between the positive and negative sets, and the denominator defines the

one within each of the two sets. The larger the F-score is, the more this feature is discriminative.

Both features of this data have low F-scores as in Eq. (8) denominator (the sum of variances of the positive and negative sets) is much larger than numerator.

Xie and Wang^[20] proposed the improved F-score to measure the discrimination between them. Given training vectors $x_k, k = 1, 2, \dots, m$, and the number of datasets, if the number of the j th dataset is $n_j, j = 1, 2, \dots, l$, then the F-score of the i th feature is defined as:

$$F_i = \frac{\sum_{j=1}^l (\bar{x}_i^{(j)} - \bar{x}_i)^2}{\sum_{j=1}^l \frac{1}{n_j - 1} \sum_{k=1}^{n_j} (x_{k,i}^{(j)} - x_i^{(j)})^2},$$

where $\bar{x}_i, \bar{x}_i^{(j)}$ are the average of the i th feature of the whole dataset and the j th dataset respectively, $x_{k,i}^{(j)}$ is the i th feature of the k th instance in the j th dataset. The numerator indicates the discrimination between each dataset, and denominator indicates the one within each of dataset. The larger the F-score is, the more this feature is discriminative.

In this study, we utilize F-score to calculate the score of each attribute in order to get the weights of the features according to $F(FS(i))$. Eq. (9) is responsible for calculating the scores of the feature masks. If the i th feature is selected ("1" represents that feature i is selected and "0" represents that feature i is not selected), $FS(i)$ equals the instance of feature i , otherwise $FS(i)$ equals 0.

$$FS(i) = \begin{cases} \text{instance } i, & \text{if } i \text{ is selected} \\ 0, & \text{if } i \text{ is not selected} \end{cases} \quad (9)$$

3.3 Objective function definition

We design an objective function which combines classification accuracy rate and F-score. Objective function is the evaluation criteria for the selected features. To get accuracy rate, we need to train and test the dataset according to the selected features.

$$fitness_i = \theta_a \times accuracy_i + \theta_b \times \left[\frac{\sum_{j=1}^{N_b} F(FS(i))}{\sum_{k=1}^{N_b} F(k)} \right]. \quad (10)$$

In Eq. (10), θ_a is the weight for SVM classification ac-

curacy rate, $accuracy_i$ the classification accuracy rate for the selected features, θ_b the weight for the score of selected features, $F(FS(i))$ the function for calculating the score of the current features, and the total score of the selected features and all features respectively are

$$\sum_{k=1}^{N_b} F(k) \quad \text{and} \quad \sum_{j=1}^{N_b} F(FS(i))$$

3.4 Multi-swarm scheduling module

MSPSO is proposed, which holds a number of swarms scheduled by the multi-swarm scheduling module. Each swarm controls its iteration procedure, position updates, velocity updates, and other parameters respectively. Each swarm selects different occasions from current computing environment, then, sends the current results to the multi-swarm scheduling module to decide whether it affects other swarms. The scheduling module monitors all the sub-swarms, and gathers the results from the sub-swarms.

Fig. 1 shows the structure of multi-swarm scheduling model, which consists of a multi-swarm scheduler and some sub-swarms. Each sub-swarm contains a number of particles. The multi-swarm scheduler can send commands or data to sub-swarms, and vice versa.

(1) The swarm request rule

If the current sub-swarm meets the condition according to Eq. (11), it sends the results which correspond $pbest$ (local best fitness) and $gbest$ (global best fitness) values to the multi-swarm scheduler. If $S_i = 1$, the current swarm sends records which contain the $pbest$ and $gbest$ values, otherwise the current swarm does not send the results.

$$S_i = \begin{cases} 1, & \text{if } d_i < \frac{tit_i - it_i}{tit_i} \times rand() \times Fitness \\ 0, & \text{if } d_i \geq \frac{tit_i - it_i}{tit_i} \times rand() \times Fitness \end{cases}, \quad (11)$$

In Eq. (11), d represents a threshold, tit the maximal iteration number, it the current iteration number. $rand()$ is a random number uniformly distributed in $U(0, 1)$.

(2) The multi-swarm scheduler request rule

The multi-swarm scheduler monitors each sub-swarm, and sends a request in order to obtain a result from current sub-swarm when the current sub-swarm is valuable. If sub-swarm has sent the swarm request rules more than $k \times n$ times, where $k = 3, n = 1, 2, 3, \dots, 100$,

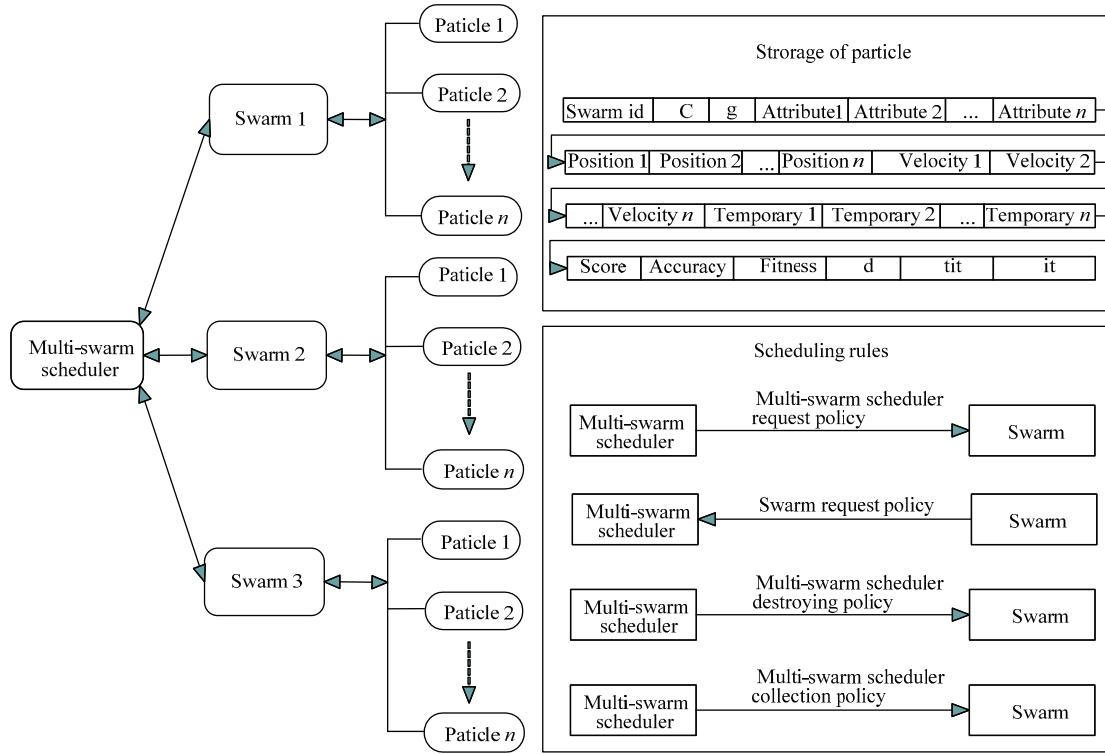


Fig. 1 The structure of multi-swarm scheduling.

the multi-swarm scheduler will send the rule. The multi-swarm scheduler request rule is touched off according to evaluating the activity level of the current sub-swarm. The more active the sub-swarm is, the more valuable it is, since the best result may be in it.

(3) The multi-swarm collection rule

The multi-swarm scheduler collects results from the alive sub-swarm and updates $pbest$ and $gbest$ from storage table.

(4) The multi-swarm destroying rule

a. If the swarm sends the swarm request rule k times and $k < f_i$ according to Eq. (12), then the multi-swarm scheduler destroys the current sub-swarm.

b. If the swarm does not change the $gbest$ in pn iterations, then the multi-swarm scheduler destroys the current sub-swarm. We set pn in the initialization of PSO.

$$f_i = \frac{\sum_{l=1}^n ite(l) \times m}{pl} \quad (12)$$

In Eq. (12), $ite()$ is the function for calculating how many times the sub-swarm sends swarm request rule, m a threshold, pl the alive sub-swarm size.

3.5 MSPSO algorithm

Step 1: Load the dataset from the text file and convert the dataset from stream format to object format. Store the formatted memory data to temporary table for the initialization of PSO. Initialize the size of swarms randomly, and assign different memory to each swarm. Initialize all particle positions x_{ij} and velocities v_{ij} of each swarm with random values, then calculate objective function. Update $pbest$ (local best) and $gbest$ (global best) of each swarm from the table. Go to Step 2.

Step 2: Specify the parameters of each swarm including the lower and upper bounds of the velocity, the size of particles, the number of iterations, c_1 (the cognition learning factor), c_2 (social learning factor), d_i (in Eq. (11)), m (in the multi-swarm destroying rule) and pn (in Eq. (12)). Set iteration number = 0, current particle number = 1, tit_i = size of particles, and it_i = current particle number. Go to Step 3.

Step 3: In each swarm, if current iteration number < iteration number or $gbest$ keeps no changes less than 45 iterations, go to Step 4, otherwise destroy the swarm, and go to Step 10. The main scheduling module updates the $pbest$, and compares the $gbest$ of current swarm with the previous one in the module, then judge whether to

update *gbest* using multi-swarm scheduler request rule or not. If *gbest* or *pbest* is changed, execute multi-swarm collection rule.

Step 4: In each swarm, if current particle number < particle size, go to Step 5, otherwise, go to Step 9.

Step 5: In each swarm, get *gbest* and *pbest* from the table and each particle updates its position and velocity. Go to Step 6.

Step 6: Restrict position and velocity of each individual. Go to Step 7.

Step 7: Each particle calculates its fitness and updates *pbest* and *gbest*. Execute swarm request rule, and go to Step 8. If the current swarm needs to be destroyed according to multi-swarm destroying rule, dispose the current swarm, and exit.

Step 8: current particle number = current particle number + 1. Go to Step 4.

Step 9: current iteration number = current iteration number + 1. Go to Step 3.

Step 10: Execute multi-swarm collection rule, and exit.

3.6 Convergence and complexity analysis

Convergence analysis and stability studies have been reported by Clerc and Kennedy^[14], Trelea^[15], Kadirkamanathan *et al.*^[16], and Jiang *et al.*^[17]. The above studies proved conditions which could lead PSO to converge in limited iterations. In order to guarantee the convergence of the proposed method, we set the parameters of PSO as $\omega = 0.9$, $c_1 = 2$, $c_2 = 2$ (according to Refs. [18] and [19]).

The time complexity of the proposed method is $O(M \times N \times K)$, where M , N , K are the number of iterations, the number of sub-swarms, the number of particles respectively. In the worst case, if the number of sub-swarms remains unchanged and the number of iteration reaches the maximum iteration number, the time complexity is $O(M \times N \times K)$. In general, the number of sub-swarms is reduced after some iterations, and thus the time complexity is

$$O\left(\sum_{i=1}^M L \times K\right), \text{ where } 1 \leq L \leq N.$$

4 Experiments and results

4.1 Experimental setting

The numbers of iterations and particles are set to

400 and 50 respectively. The searching ranges for c and γ are as follow: $c \in [2^{-15}, 2^{15}]$, $\lambda \in [2^{-15}, 2^{15}]$, $[-v_{\max}, v_{\max}]$ is predefined as $[-1000, 1000]$ for parameter c , as $[-1000, 1000]$ for parameter γ , and as $[-6, 6]$ for feature mask. For objective function, we set w_a and w_b to 0.8 and 0.2 according to our experience. The following datasets taken from the UCI machine learning and StatLog databases are used to evaluate the performance of the proposed IFS approach: Australian, German, Cleveland heart, breast cancer, heart disease, vehicle silhouettes, hill-valley, landsat satellite, sonar, and Wisconsin Diagnostic Breast Cancer (WDBC).

The 10-fold cross validation was used to evaluate the classification accuracy. Then the average error across all 10 trials was computed. Because hill-valley and landsat satellite datasets have pre-defined training/test splits. Thus, except these datasets, all of the experimental results are averaged over the 10 runs of 10-fold Cross-Validation (CV).

Table 1 Dataset description

No.	Dataset	Classes	Instances	Features	Missing value
1	Australian (Statlog project)	2	690	14	Yes
2	German (Statlog project)	2	1000	24	No
3	Cleveland heart	2	303	13	Yes
4	Breast cancer (Wisconsin)	2	699	9	Yes
5	Heart disease (Statlog project)	2	270	13	No
6	Vehicle silhouettes (Vehicle)	4	846	17	No
7	Hill-valley	2	1212	100	No
8	Landsat satellite (Landsat)	6	6435	36	No
9	Sonar	2	208	60	No
10	WDBC	2	569	30	No

4.2 Results

Table 2 shows the classification accuracy rates of IFS with and without feature selection. As shown in Table 2, the IFS with feature selection performs significantly better than IFS without feature selection in almost all cases examined at the significance level of 0.05, except the Australian dataset. The average classification accuracy rate for each dataset improved significantly after feature selection.

The results show that the classification accuracy rates of the IFS approach with and without feature selection were better than those of grid search in all cases

as shown in Table 3. Grid search is a local search method which is vulnerable to local optimum. Grid search can supply local optimal parameters to SVM, but the search region is small, and it can not lead SVM to higher classification accuracy rate. The empirical analysis indicates that the developed IFS approach can obtain the optimal parameter values, and find a subset of discriminative features without decreasing the SVM classification accuracy.

Table 2 Results of the proposed IFS with and without feature selection

Dataset	With feature selection			Without feature selection	Pair <i>t</i> test <i>P</i> -value
	Number of original features	Number of selected features	Accuracy rate (%)	Accuracy rate (%)	
Australian	14	8.4 ± 2.318	90.9	86.4	0.06
German	23	12.7 ± 1.025	80.2	75.9	< 0.001
Cleveland heart	13	6.1 ± 1.103	91.1	85.7	< 0.001
Breast cancer	9	4.9 ± 0.734	99.1	96.9	< 0.001
Heart disease	13	7.8 ± 0.949	91.5	84.4	< 0.001
Vehicle	17	7.1 ± 0.432	89.6	85.8	< 0.001
Hill-valley	100	40.1 ± 1.264	74.1	71.2	< 0.001
Landsat	36	13 ± 0.668	95.4	91.9	< 0.001
Sonar	60	25.1 ± 0.977	93.7	90.1	< 0.001
WDBC	30	13 ± 1.331	99.4	97.8	0.011

Table 3 Experimental results summary of IFS with feature selection, IFS without feature selection and grid search algorithm

Dataset	(1) IFS with feature selection	(2) IFS without feature selection	(3) Grid search	Pair <i>t</i> test (1)vs(3)	Pair <i>t</i> test (2)vs(3)
Australian	90.9	86.4	84.7	< 0.001	< 0.001
German	80.2	75.9	75.7	< 0.001	< 0.001
Cleveland heart	91.1	85.7	82.3	< 0.001	< 0.001
Breast cancer	99.1	96.9	95.2	< 0.001	< 0.001
Heart disease	91.5	84.4	83.6	< 0.001	< 0.001
Vehicle	89.7	85.8	84.2	< 0.001	0.21
Hill-valley	74.1	71.2	69.8	0.01	< 0.001
Landsat	95.4	91.9	91.1	< 0.001	0.012
Sonar	93.7	90.1	88.9	0.028	< 0.001
WDBC	99.4	97.8	97.4	< 0.001	0.531

The comparison between IFS and GA + SVM by using feature selection is shown in Table 4. The detail parameter settings for GA+SVM were as follows: population size = 500, crossover rate = 0.7, mutation rate = 0.02. The classification accuracy rates of IFS with feature selection were higher than GA + SVM for all datasets, whereas the classification accuracy rates of GA + SVM were higher than IFS without feature selection as shown in Table 4. Therefore, it is important to eliminate noisy, irrelevant features for increasing the classification accuracy rates.

Table 4 Comparison between the IFS and GA + SVM approach

Dataset	Number of original features	IFS		GA + SVM	
		Number of selected features	Accuracy rate(%)	Number of selected features	Accuracy rate(%)
Australian	14	8.4 ± 2.318	90.9	7.9 ± 0.432	88.1
German	23	12.7 ± 1.025	80.2	10.1 ± 0.986	77.4
Cleveland heart	13	6.1 ± 1.103	91.1	6.9 ± 2.011	86.8
Breast cancer	9	4.9 ± 0.734	99.1	5.5 ± 0.988	98.2
Heart disease	13	7.8 ± 0.949	91.5	8.1 ± 0.445	86.7
Vehicle	17	7.1 ± 0.432	89.6	11.5 ± 0.664	88.1
Hill-valley	100	40.1 ± 1.264	74.1	55.9 ± 1.981	73.5
Landsat	36	13 ± 0.668	95.4	18.3 ± 1.498	93.4
Sonar	60	25.1 ± 0.977	93.7	31.0 ± 1.212	91.6
WDBC	30	13 ± 1.331	99.4	17.3 ± 0.991	98.9

Fig. 2a and Fig. 2b show the global best classification accuracies with different iterations on Australian and German datasets using IFS, PSO+SVM, GA+SVM respectively. Fig. 2e and Fig. 2f show the local best classification accuracies with different iterations on Australian and German datasets using IFS, PSO+SVM and GA+SVM respectively. The convergence speeds of PSO+SVM and GA +SVM were faster than IFS, whereas the resultant classification accuracies of PSO+SVM and GA+SVM were lower than IFS. Moreover, PSO+SVM and GA+SVM prematurely converged to local optimum, and thus it convinces that IFS has more exploration capability. The numbers of selected features with evolution on German and Australian datasets using three methods are shown in Fig. 3 and Fig. 4 respectively. Fig. 2c and Fig. 2d show the number

of sub-swarms with different iterations on Australian and German datasets using IFS. With different numbers of initial sub-swarms, a great number of sub-swarms were reduced, and only a small number of sub-swarms were remained at the final iteration. Most of the weak sub-swarms are eliminated during the evolution, and thus it can be seen that excellent sub-swarms are preserved after competition, as enhance the exploration ability of the whole swarm to obtain more important features.

The comparison between IFS and PSO+SVM using

feature selection in terms of number of selected features and average classification accuracy rates is shown in Table 5. For comparison purpose, we implemented the PSO+SVM approach using the standard PSO algorithm, and the parameter settings were described as follows: iteration size was set as 500, number of particles as 100. The classification accuracy rate was adopted as the objective function. The analytical results reveal that IFS with feature selection performs significantly superior to the standard PSO with feature selection in all datasets in terms of the classification accuracy rates.

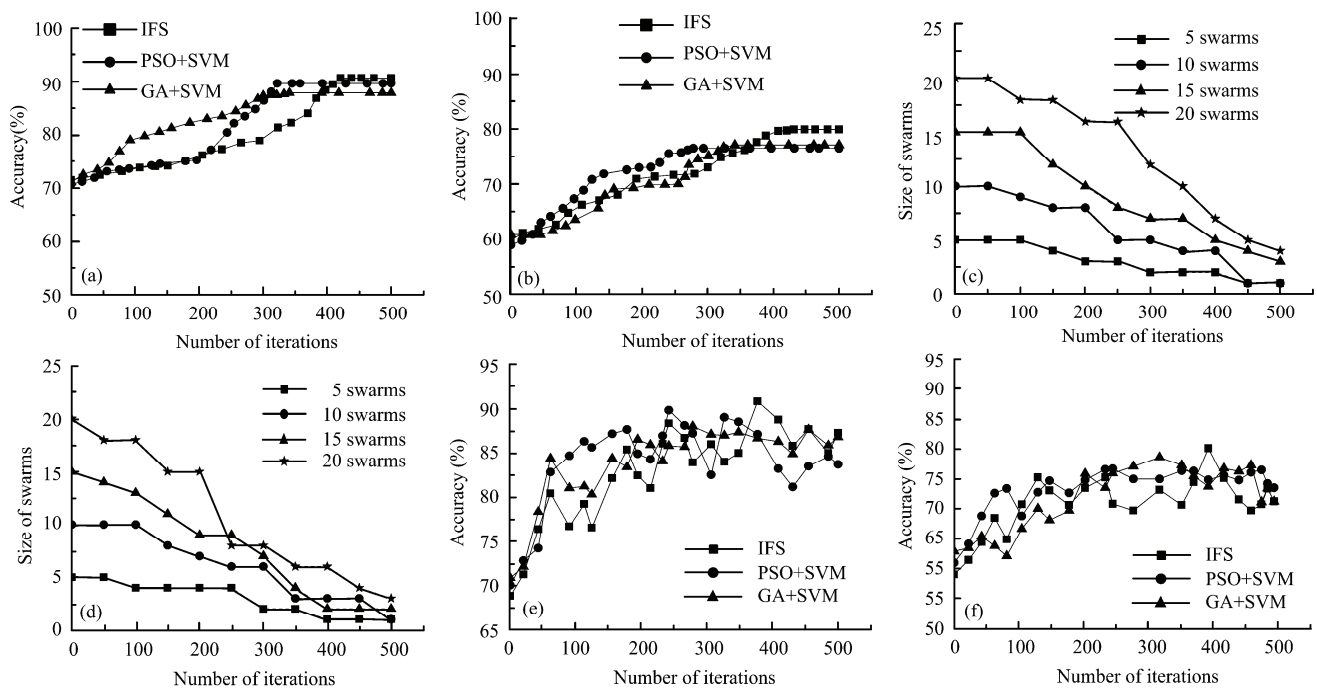


Fig. 2 Prediction accuracies and number of sub-swarm with different iterations. (a) Global best accuracies with different iterations on Australian dataset using IFS, PSO+SVM and GA+SVM. (b) Global best accuracies with different iterations on German dataset using IFS, PSO+SVM and GA+SVM. (c) Each curve corresponding to a number of initial sub-swarms on Australian dataset using IFS. (d) Each curve corresponding to a number of initial sub-swarms on German dataset using IFS. (e) Local best accuracies with different iterations on Australian dataset using IFS, PSO+SVM and GA+SVM. (f) Local best accuracies with different iterations on German dataset using IFS, PSO+SVM and GA+SVM.

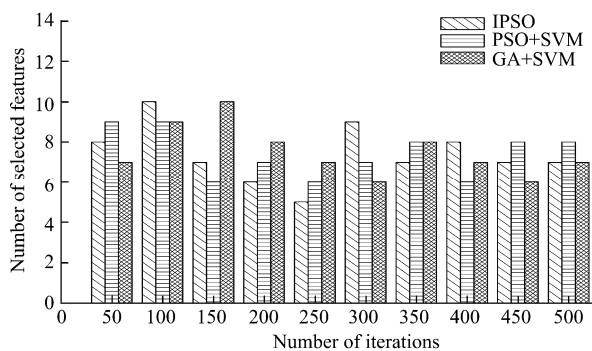


Fig. 3 Number of selected features with different iterations on Australian dataset using IFS, PSO+SVM and GA+SVM.

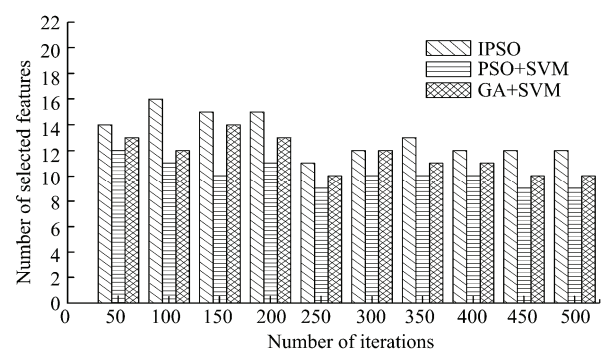


Fig. 4 Number of selected features with different iterations on German dataset using IFS, PSO+SVM and GA+SVM.

Table 5 Comparison between the IFS and standard PSO

Dataset	Number of original features	IFS		PSO+SVM	
		Number of selected features	Accuracy rate (%)	Number of selected features	Accuracy rate (%)
Australian	14	8.4 ± 2.318	90.9	7.1 ± 0.798	89.9
German	23	12.7 ± 1.025	80.2	9.4 ± 1.233	76.8
Cleveland heart	13	6.1 ± 1.103	91.1	6.4 ± 0.558	87.4
Breast cancer	9	4.9 ± 0.734	99.1	5.8 ± 0.447	97.6
Heart disease	13	7.8 ± 0.949	91.5	6.2 ± 0.976	85.3
Vehicle	17	7.1 ± 0.432	89.66	10.2 ± 1.298	86.2
Hill-Valley	100	40.1 ± 1.264	74.12	61.3 ± 2.110	72.3
Landsat	36	13 ± 0.668	95.44	15.1 ± 0.975	93.4
Sonar	60	25.1 ± 0.977	93.71	35.2 ± 1.123	90.8
WDBC	30	13 ± 1.331	99.41	16.9 ± 1.652	98.2

5 Conclusion

In this study, a novel multi-swarm MSPSO algorithm is proposed to solve discrete problem, an efficient objective function of which is designed by taking into consideration classification accuracy rate and F-score. In order to describe the competition among the swarms, we introduced the mechanism for survival of the fittest. To further settle the feature selection problem, we put forward the IFS approach, in which both the SVM parameter optimization and the feature selection are dynamically executed by MSPSO algorithm, then, SVM model performs the classification tasks using the optimal parameter values and the subset of features. The evaluation on the 10 benchmark problems by comparing with the standard PSO based, genetic algorithm based, and grid search based methods indicates that the proposed approach performs significantly advantageously over others in terms of the classification accuracy rates.

Acknowledgments

This work was supported by National Natural Science Foundation of China (Grant no. 60971089), National Electronic Development Foundation of China (Grant no. 2009537), Jilin Province Science and Technology Department Project of China (Grant no. 20090502).

References

- [1] Guyon I, Elisseeff A. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 2003, **3**, 1157–1182.
- [2] Kennedy J, Eberhart R. Particle swarm optimization. *Proceedings of the IEEE International Conference on Neural Network*, Perth, Australia, 1995, 1942–1948.
- [3] Eberhart R, Kennedy J. A new optimizer using particle swarm theory. *Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, Nagoya, Japan, 1995, 39–43.
- [4] Lin S W, Ying K C, Chen S C, Lee Z J. Particle swarm optimization for parameter determination and feature selection of support vector machines. *Expert Systems with Applications*, 2008, **35**, 1817–1824.
- [5] Huang C L, Dun J F. A distributed PSO-SVM hybrid system with feature selection and parameter optimization. *Applied Soft Computing*, 2008, **8**, 1381–1391.
- [6] Blackwell T. Particle swarms and population diversity. *Soft Computing*, 2005, **9**, 793–802.
- [7] Blackwell T, Branke J. Multiswarms, exclusion, and anti-convergence in dynamic environments. *IEEE Transactions on Evolutionary Computation*, 2006, **10**, 459–472.
- [8] Parrott D, Li X D. Locating and tracking multiple dynamic optima by a particle swarm model using speciation. *IEEE Transactions on Evolutionary Computation*, 2006, **10**, 440–458.
- [9] Niu B, Zhu Y L, He X X, Wu H. MCPSPSO: A multi-swarm cooperative particle swarm optimizer. *Applied Mathematics and Computation*, 2007, **185**, 1050–1062.
- [10] Chen Y W, Lin C J. Combination of feature selection approaches with SVM in credit scoring. *Expert Systems with Applications*, 2006, **37**, 315–324.
- [11] Kennedy J, Eberhart R. A discrete binary version of the particle swarm algorithm. *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, Orlando, USA, 1997, 4104–4108.
- [12] Vapnik V N. *The Nature of Statistical Learning Theory*, 2nd ed, Springer, New York, 1999.
- [13] Boser B E, Guyon I M, Vapnik V N. A training algorithm for optimal margin classifiers. *Proceedings of the fifth Annual Workshop on Computational Learning Theory*, Pittsburgh, USA, 1992, 144–152.
- [14] Clerc M, Kennedy J. The particle swarm-explosion, stability, and convergence in a multidimensional complex space. *IEEE Transactions on Evolutionary Computation*, 2002, **6**, 58–73.
- [15] Trelea I C. The particle swarm optimization algorithm:

- convergence analysis and parameter selection. *Information Processing Letters*, 2003, **85**, 317–325.
- [16] Kadirkamanathan V, Selvarajah K, Fleming P J. Stability analysis of the particle dynamics in particle swarm optimizer. *IEEE Transactions on Evolutionary Computation*, 2006, **10**, 245–255.
- [17] Jiang M, Luo Y P, Yang S Y. Stochastic convergence analysis and parameter selection of the standard particle swarm optimization algorithm. *Information Processing Letters*, 2007, **102**, 8–16.
- [18] Shi Y, Eberhart R. Modified particle swarm optimizer. *Proceedings of IEEE International Conference on Evolutionary Computation*, Anchorage, USA, 1998, 69–73.
- [19] Zhan Z H, Zhang J, Li Y. Adaptive Particle Swarm Optimization. *IEEE Transactions on Systems Man and Cybernetics Part B-Cybernetics*, 2009, **39**, 1362–1381.
- [20] Xie J Y, Wang C X. Using support vector machines with a novel hybrid feature selection method for diagnosis of erythematous-squamous diseases. *Expert Systems with Applications*, 2011, **38**, 5809–5815.