Feature Selection using PSO-SVM

Chung-Jui Tu, Li-Yeh Chuang, *Jun-Yang Chang,* and Cheng-Hong Yang, *Member, IAENG*

***Abstract*—The feature selection process can be considered a problem of global combinatorial optimization in machine learning, which reduces the number of features, removes irrelevant, noisy and redundant data, and results in an acceptable classification accuracy. Feature selection is of great importance in pattern classification, medical data processing, machine learning, and data mining applications. Therefore, a good feature selection method based on the number of features investigated for sample classification is needed in order to speed up the processing rate, predictive accuracy, and to avoid incomprehensibility. In this paper, particle swarm optimization (PSO) is used to implement a feature selection, and support vector machines (SVMs) with the one-versus-rest method serve as a fitness function of PSO for the classification problem. The proposed method is applied to five classification problems from the literature. Experimental results show that our method simplifies features effectively and obtains a higher classification accuracy compared to the other feature selection methods.**

***Index Terms*—Feature Selection, Machine Learning, Particle Swarm Optimization, Support Vector Machines.**

I. Introduction

For many pattern classification problems, a higher number of features used do not necessarily translate into a higher classification accuracy. In some cases the performance of algorithms devoted to speed and predictive accuracy of the data characterization can even decrease. Therefore, feature selection can serve as a pre-processing tool of great importance before solving the classification problems. The purpose of the feature selection is to reduce the maximum number of irrelevant features while maintaining an acceptable classification accuracy. A good feature selection method can reduce the cost of feature measurement, and increase classifier efficiency and classification accuracy. Feature selection is of considerable importance in pattern classification, data analysis, multimedia information retrieval, medical data processing, machine learning, and data mining applications.

Manuscript received March 31, 2006. This research was supported by the National Science Council, R.O.C., under grant NSC 94-2614-E-151-001.

C. -J. Tu is with the Department of Electronic Engineering, National Kaohsiung Univeristy of Applied Sciences, Kaohsiung, Taiwan 80708. (e-mail: 1093320134@cc.kuas.edu.tw).

L. -Y. Chuang is with the Department of Chemical Engineering, I-Shou Univeristy, Kaohsiung, Taiwan 80708. (e-mail: chuang@isu.edu.tw).

C. -H. Yang, is with the Department of Electronic Engineering, National Kaohsiung Univeristy of Applied Sciences, Kaohsiung, Taiwan 80708. (e-mail: chyang@cc.kuas.edu.tw).

J. -Y. Chang is with the Department of Information Management, National Kaohsiung Univeristy of Applied Sciences, Kaohsiung, Taiwan 80708. (e-mail: cyc@cc.kuas.edu.tw).

Several methods have been previously used to perform feature selection on training and testing data, for example genetic algorithms (Raymer *et al.*, 2000), branch and bound algorithms (Narendra *et al*., 1977), sequential search algorithms (Pudil *et al*., 1994), mutual information (Roberto, 1994), and tabu search (Zhang *et al.*, 2002). In order to obtain better classification accuracy for classification problems, an improved feature selection process is needed.

Many SVMs have been successfully applied to gene expression data classification problems (Furey *et al.*, 2000; Guyon *et al.*, 2002; Lee *et al.*, 2003) since they are not negatively affected by high dimensionality; hence they can obtain a higher accuracy than a general classification methods, SVMs obtain a maximum margin of a hyper-plane in order to optimize the obtained support vector machine. This avoids a common disadvantage of general classification methods, namely the long operation time, and can reduce training errors of the SVMs.

In this paper, PSO is used to implement a feature selection, and SVMs with the one-versus-rest method were used as evaluators for the PSO fitness function for five multiclass problems taken from the literature. The results reveal that our method elucidated a better accuracy than the classification methods they were compared to.

This paper is organized as follows: in the next section, the methods used are introduced. They include particle swarm optimization, support vector machines, and the one-versus-rest method. Section 3 details the experimental results and contains a discussion. Results obtained by the proposed method are compared with results obtained by using other methods. Finally, concluding remarks are made in Section 4.

II. Methods

A. Feature Selection Method

Particle swarm optimization (PSO) is a population-based stochastic optimization technique, and was developed by Kennedy and Eberhart in 1995. PSO simulates the social behavior of organisms, such as bird flocking and fish schooling, to describe an automatically evolving system. In PSO, each single candidate solution is "an individual bird of the flock", that is, a particle in the search space. Each particle makes use of its individual memory and knowledge gained by the swarm as a whole to find the best solution (Venter 2002). All of the particles have fitness values, which are evaluated by a fitness function to be optimized, and have velocities which direct the movement of the particles. During movement, each particle adjusts its position according to its own experience, as well as according to the experience of a neighboring particle,

and makes use of the best position encountered by itself and its neighbor. The particles move through the problem space by following a current of optimum particles.

The initial swarm is generally created in such a way that the population of the particles is distributed randomly over the search space. At every iteration, each particle is updated by following two "best" values, called *pbest* and *gbest*. Each particle keeps track of its coordinates in the problem space, which are associated with the best solution (fitness) the particle has achieved so far. This fitness value is stored, and called *pbest*. When a particle takes the whole population as its topological neighbor, the best value is a global "best" value and is called *gbest*. The pseudo code of the PSO procedure is given below.

Initialize population
While (number of generations, or the stopping criterion is not met)
  For p = 1 to number of particles
    If the fitness of X_p is greater than the fitness of $pbest_p$
    then Update $pbest_p = X_p$
    For $k \in$ Neighborhood of X_p
      If the fitness of X_k is greater than that of $gbest$ then
        Update $gbest = X_k$
    Next k
    For each dimension d
      $v_{pd}^{new} = w \times v_{pd}^{old} + c_1 \times rand_1 \times \left(pbest_{pd} - x_{pd}^{old}\right)$
      $+ c_2 \times rand_2 \times \left(gbest_d - x_{pd}^{old}\right)$
      if $v_{pd} \notin \left(V_{\min},\ V_{\max}\right)$ then
      $v_{pd} = \max\left(\min\left(V_{\max},\ v_{pd}\right),\ V_{\min}\right)$
      $x_{pd} = x_{pd} + v_{pd}$
    Next d
  Next p
Next generation until stopping criterion

v_{pd}^{new} and v_{pd}^{old} are the particle velocities, x_{pd}^{old} is the current particle position (solution), and x_{pd}^{new} is the updated particle position (solution). The values $pbest_{pd}$ and $gbest_d$ are defined as stated above. The two factors $rand_1$ and $rand_2$ are random numbers between (0, 1), whereas c_1 and c_2 are acceleration factors, usually $c_1 = c_2 = 2$. Particle velocities of each dimension are tried to a maximum velocity $V_{\max}$. If the sum of velocities causes the total velocity of that dimension to exceed $V_{\max}$, then the velocity of that dimension is limited to $V_{\max}$. $V_{\max}$ is a user-specified parameter.

Based on the rules of particle swarm optimization, we set the required particle number first, and then the initial coding alphabetic string for each particle is randomly produced. In our case we coded each particle to imitate a chromosome in a genetic algorithm; each particle was coded to a binary alphabetic string $S = F_1 F_2 \ldots F_n$, $n = 1,\ 2,\ \ldots,\ m$; the bit value {1} represents a selected feature, whereas the bit value {0} represents a non-selected feature.

The adaptive functional values were data based on the particle features representing the feature dimension; this data was classified by a support vector machine (SVM) to obtain classification accuracy; the SVM serves as an evaluator of the PSO fitness function. For example, when a 10-dimensional data set (n=10) $S_n = (F_1 F_2 F_3 F_4 F_5 F_6 F_7 F_8 F_9 F_{10})$ is analyzed using particle swarm optimization to select features, we can select any number of features smaller than n, i.e. we can chose a random 6 features, here $S_n = (F_1 F_3 F_5 F_7 F_9 F_{10})$. When the adaptive value is calculated, these 6 features in each data set represent the data dimension and are evaluated by the SVM. The fitness value for the SVM evolves according to the K-fold Cross-Validation Method (Stone, 1974) for small sample sizes, and according to the Holdout Method (Stone, 1974) for big sample sizes. Using the K-Fold Cross-Validation Method, we separated the data into 10 parts $\{D_1,\ D_2,\ \ldots,\ D_{10}\}$, and carried out training and testing a total of 10 times. If every part D_n, $n = 1,\ 2,\ \ldots,\ 10$ is processed as a test set, the other 9 parts will be training sets. Following 10 times of training and testing, 10 classification accuracies are produced, and the averages of these 10 accuracies are used as the classification accuracy for the data set. When the Holdout Method is used, the data can be divided into two parts, a training set part, which contains a larger amount of data, and a test set part, which contains relatively fewer data. We assumed that the obtained classification accuracy is an adaptive functional value.

Each particle renewal is based on its adaptive value. The best adaptive value for each particle renewal is *pbest*, and the best adaptive value within a group of *pbest* is *gbest*. Once *pbest* and *gbest* are obtained, we can keep track of the features of *pbest* and *gbest* particles with regard to their position and speed. In this study, a binary version of a PSO algorithm is used for particle swarm optimization (Kennedy *et al.*, 1997). The position of each particle is given in a binary string form that represents the feature selection situation. Each particle is updated according to the following equations.

$$v_{pd}^{new} = w \times v_{pd}^{old} + c_1 \times rand_1 \times \left(pbest_{pd} - x_{pd}^{old}\right) + c_2 \times rand_2 \times \left(gbest_d - x_{pd}^{old}\right) \quad (1)$$

$$S\left(v_{pd}^{new}\right) = \frac{1}{1 + e^{-v_{pd}^{new}}} \quad (2)$$

$$\text{if } \left(rand < S\left(v_{pd}^{new}\right)\right) \text{ then } x_{pd}^{new} = 1 \text{ ; else } x_{pd}^{new} = 0 \quad (3)$$

The feature after renewal is calculated by the function $S(v_{id}^{new})$ (Eq. 2), in which the speed value is v_{pd}^{new}. If $S(v_{pd}^{new})$ is larger than a randomly produced disorder number that is within (0, 1), then its position value F_n, $n = 1,\ 2,\ \ldots,\ m$ is represented as {1} (meaning this feature is selected as a required feature for the next renewal). If $S(v_{id}^{new})$ is smaller than a randomly produced disorder number that is within {0~1}, then its position value F_n, $n = 1,\ 2,\ \ldots,\ m$ is

represented as {0} (meaning this feature is not selected as a required feature for the next renewal).

B. Multiclass Classification Method

Support Vector Machines (SVMs) were originally introduced by Vapnik and co-workers (Frieß *et al.*, 1998) for classification tasks, and were subsequently extended to regression problems (Drucker *et al.*, 1997). The idea behind SVMs is the following: input points are mapped to a high dimensional feature space, where a separating hyper-plane can be found. The algorithm is chosen in such a way as to maximize the distance from the closest patterns, a quantity which is called the margin. SVMs are learning systems designed to automatically trade-off accuracy and complexity by minimizing an upper bound on the generalization error provided by the Vapnik-Chervonenkis (VC) theory (Vapnik, 1995). In a variety of classification problems, SVMs have shown a performance which can reduce training and testing errors, thereby obtaining a higher recognition accuracy. SVMs can be applied to very high dimensional data without changing their formulation.

The hyper-plane of SVMs is usually found by using a quadratic programming routine, which is then solved with optimization routines form numerical libraries. These steps are non-trivial to implement and computationally intensive (Frieß *et al.*, 1998). In this study, Kernel-Adatron (KA) algorithms (Frieß *et al.*, 1998), are used to emulate SVM training procedures, which combine the implementation simplicity of the Adatron with the capability of working in nonlinear feature spaces. The Adatron comes with the theoretical guarantee of converging exponentially fast in a given number of iterations, provided that a solution exists (Anlauf *et al.*, 1989; Opper, 1988). By introducing Kernels into the algorithm it is possible to find a maximal margin hyper-plane in a high feature space, which is equivalent to nonlinear decision boundaries in the input space. The algorithm comes with all the theoretical guarantees given by the VC (Vapnik and co-workers) theory for large margin classifiers (Boser *et al.*, 1992; Cortes *et al.*, 1995), as well as the convergence properties detailed in the statistical mechanics literature.

The Kernel-Adatron algorithm theoretically converges in a finite number of steps to the maximal margin, provided that the linearly independent data points are linearly separable in the feature space with a margin $\lambda > 0$. This result can be obtained for the following two reasons: all the fixed points of KA are Kuhn-Tucker points and, vice versa, KA always converges to a unique fixed point (Colin 1998). The KA procedure is described below.

1) Initialize $\alpha_i^t = 0$.
2) For $i = 1,\ 2,\ 3,\ \ldots,\ n$ execute step 3, 4 below.
3) For a labeled point (x_i, y_j) calculate:

$$z_i = \sum_{j=1}^{n} \alpha_j y_j k(x_i, y_j) \tag{4}$$

4) Calculate $\delta\alpha_i^t = \eta(1 - z_i y_i)$:

4.1) If $(\alpha_i^t + \delta\alpha_i^t) \le 0$ then $\alpha_i^t = 0$.

4.2) If $(\alpha_i^t + \delta\alpha_i^t) > C$ then $\alpha_i^t = C$.

4.3) If $(\alpha_i^t + \delta\alpha_i^t) > 0$ then $\alpha_i^t = (\alpha_i^t + \delta\alpha_i^t)$.

5) If a maximum number of iterations is exceeded or the margin λ is approximately 1 then stop, otherwise return to step 2.

$$\lambda = \frac{1}{2}[\min_{y=+1}(z_i) - \max_{y=-1}(z_i)] \tag{5}$$

The maximum number of iterations is 100, and the kernel function is the Radial Basis Function (RBF):

$$k(x_i, y_j) = \exp^{-r\left\|x_i - y_j\right\|},\ i = j = 1,\ 2,\ \ldots,\ n \tag{6}$$

This algorithm is a gradient ascent routine that maximizes the margin in the feature space similar to a perceptron-like algorithm, the Adatron, and was dubbed by Campbell and Christianini the Kernel-Adatron algorithm (Frieß *et al.*, 1998). C and r are used to control the trade-off between training error and generalization ability. The decomposition techniques used for KA are one-versus-rest.

C. One-Versus-Rest

The one-versus-rest method assembles classifiers that distinguish one from all the other classes. For each i, $1 \le i \le k$, a binary classifier separating class i from the rest is built. To predict a class label of a given data point, the output of each of the k classifiers is obtained. If there is a unique class label, say j, which is consistent with all the k predictions, the data point is assigned to class j. Otherwise, one of the k classes is selected randomly. Very often though, a situation arises in which consistent class assignment does not exist, which could potentially lead to problems (Scholkopf and Smola, 2002)

The pseudo code of the proposed method for classification problems is given below.

Initialize population
While (number of generations, or the stopping criterion is not met)
  For p = 1 to number of particles
    Segment training data and testing data
    Initialize super parameter α
    $k(x_i, y_j) = \exp^{-r\left\|x_i - y_j\right\|} X_p$
    While (number of iterations, or the stopping criterion is not met)
      For i = 1 to number of training data
        $z_i = \sum_{j=1}^{n} \alpha_j y_j k(x_i, x_j)$
        $\delta\alpha_i = \eta(1 - z_i y_i)$
        If $(\alpha_i + \delta\alpha_i) \le 0$ then $\alpha_i = 0$
        If $(\alpha_i + \delta\alpha_i) > C$ then $\alpha_i = C$
        If $(\alpha_i + \delta\alpha_i) > 0$ then $\alpha_i = (\alpha_i + \delta\alpha_i)$
      Next i
    Next iteration until criterion
    For i = 1 to number of testing data

$$z_i = \sum_{j=1}^{n} \alpha_j y_j k(x_i, x_j)$$

If $z_i > 0$ then $class_i = +1$ else $class_i = -1$

If $class_i = real\ class\ of\ testing\ data$ then

$right = right + 1$

Next i

$fitness_p = right / number\ of\ testing\ data$

If the fitness of X_p is greater than the fitness of $pbest_p$ then Update $pbest_p = X_p$

For $k \in$ Neighborhood of X_p

If the fitness of X_k is greater than that of $gbest$ then

Update $gbest = X_k$

Next k

For each dimension d

$$v_{pd}^{new} = w \times v_{pd}^{old} + c_1 \times rand_1 \times \left(pbest_{pd} - x_{pd}^{old}\right) + c_2 \times rand_2 \times \left(gbest_d - x_{pd}^{old}\right)$$

$$S\left(v_{pd}^{new}\right) = \frac{1}{1 + e^{-v_{pd}^{new}}}$$

if $\left(rand < S\left(v_{pd}^{new}\right)\right)$ then $x_{pd}^{new} = 1$; else $x_{pd}^{new} = 0$

Next d

Next p

Next generation until stopping criterion.

The two factors $rand_1$ and $rand_2$ are random numbers between (0, 1), whereas c_1 and c_2 are learning factors, usually $c_1 = c_2 = 2$. In this paper, we used the SVM parameters shown in Table 1 for all classification problems.

III. Results and Discussion

The dataset we used in this study was obtained from the UCI Repository, with the number of features (feature dimensionality) being greater than 10 [Murphy *et al.*, 1994]. The data format was arranged as shown in Table 1. Three types of classification test problems were tested. If the feature dimensionality is between 10 and 19, the sample groups can be considered small, and can be used for Vowel and Wine problems. If the feature dimensionality is between 20 and 49, the sample group test problems are of middle size, and include WDBC and Ionosphere problems. If the feature dimensionality is over 50, the test problems are large sample group problems, which include Sonar test problems. Two evaluation methods were used: the 10-fold Cross-Validation and Holdout Method. The 10-fold Cross-Validation Method is used for small sample group test problems. The other two groups of test problems use the Holdout Method. For the test problem Wine, we used a standard normalization form to arrange the value between 0 and 1.

In this paper, binary PSO is used to serve as feature selection for classification problems. It helps to improve the performance owing to its smaller number of simple parameter settings. PSO is an evolution computing technology which simulates the social behavior of fish in a school. At each iteration, a particle will, according to its fitness value and swarm fitness value, be optimized. A KA-SVM is used to evaluate the fitness values of the PSO, which can be obtained by comparing the characteristics of the general test data. The classification problems have different sample sizes and dimensions. The SVM can be applied to different dimensional data by introducing a Kernel function to find a maximal margin hyper-plane in a high feature space that is well suited to the different classification problem structures. At the same time, it reduces the amount of training and testing, thereby increasing the classification accuracy for classification problems.

Table 1 shows the format of five classification problems [Oh *et al.*, 2004]. Table 2 compares experimental results obtained by other methods from the literature with the proposed method [Oh *et al.*, 2004]. The proposed method obtained the highest classification accuracy for the Wine, WDBC and Ionosphere classification problems. The classification accuracy of the Wine and Ionosphere classification problems obtained by the proposed method are 100% and 97.33%, respectively, an increase of 4% and 2% classification accuracy compared to the other methods shown in Table 2. For the Wine classification problem, the proposed method obtained 100% classification accuracy. However, the number of features selected is less in the proposed method. This means that not all features are needed to achieve total classification accuracy. Even though the classification accuracy for the classification problems of Vowel and Sonar, is worse than the classification accuracy of the other feature selection methods, it is still comparable. These results indicate that for different classification problems, the proposed method (binary particle swarm optimization) can serve as a pre-processing tool and help optimize the feature selection process, which leads to an increase in classification accuracy. A good feature selection process reduces feature dimensions and improves accuracy.

PSO is based on the idea of collaborative behavior and swarming in biological populations. Both PSO and genetic algorithms (GAs) are population-based search approaches that depend on information sharing among their population members to enhance their search processes using a combination of deterministic and probabilistic rules. However, PSO does not have genetic operators such as crossover and mutation. Particles update themselves with the internal velocity. Compared with GAs, the information sharing mechanism in PSO is considerably different. In GAs, chromosomes share information with each other, so the whole population moves like one group towards an optimal area. In PSO, only gbest gives out the information to others. It is a one-way information sharing mechanism. The evolution only looks for the best solution. Compared with GAs, all the particles tend to converge to the best solution quickly even in the local version in most cases.

The computation time used in PSO is less than in GAs. The parameters used in PSO are also fewer. However, if the proper parameter values are set, the results can easily be optimized. Proper adjustment of the inertia weight w and the acceleration factors c_1, c_2 is very important. If the parameter adjustment is too small, the particle movement is too small. This scenario will

also result in useful data, but is a lot more time-consuming. If the adjustment is excessive, particle movement will also be excessive, causing the algorithm to weaken early, so that a useful feature set can not be obtained. Hence, suitable parameter adjustment enables particle swarm optimization to increase the efficiency of feature selection. For SVMs, correct parameter adjustment is crucial, since many parameters are involved. This can have a profound influence on the results. For different classification problems, different parameters have to be set for SVMs. The two factors r and C are especially important. A suitable adjustment of these parameters results in a better classification hyper-plane found by the SVM, and thereby enhances the classification accuracy. Bad parameter settings affect the classification accuracy negatively. In this paper, we used the parameters in Table 1 for all classification problems. The parameters settings used in our study were optimized, and could be used as a reference for future studies.

IV. Conclusions

Building an efficient classification model for classification problems with different dimensionality and different sample size is important. The main tasks are the selection of the features and the selection of the classification method. In this paper, we used PSO to perform feature selection and then evaluated fitness values with a SVM, which was combined with the one-versus-rest method, for five classification profiles. Experimental results show that our method simplified feature selection and the total number of parameters needed effectively, thereby obtaining a higher classification accuracy compared to other feature selection methods. The proposed method can serve as an ideal pre-processing tool to help optimize the feature selection process, since it increases the classification accuracy and, at the same time, keeps computational resources needed to a minimum. It could also be applied to problems in other areas in the future.

Acknowledgment

This work is partly supported by the National Science Council in Taiwan under grants NSC94-2622-E-151-025-CC3, NSC94-2311-B037-001, NSC93-2213-E-214-037, NSC92-2213-E-214-036, NSC92-2320-B-242-004, NSC92-2320-B-242-013 and by the CGMH fund CMRPG1006.

References

[1] Raymer, M.L., Punch, W.F., Goodman, E.D., Kuhn, L.A., and Jain, A. K., "Dimensionality Reduction Using Genetic Algorithms," IEEE Trans. Evolutionary Computation, vol. 4, no. 2, pp. 164-171, July 2000.

[2] Narendra, P.M. and Fukunage, K., "A Branch and Bound Algorithm for Feature Subset Selection," IEEE Trans. Computers, vol.6, no. 9, pp. 917-922, Sept. 1977.

[3] Pudil, P., Novovicova, J., and Kittler, J., "Floating Search Methods in Feature Selection," Pattern Recognition Letters, vol.15, pp. 1119-1125, 1994.

[4] Roberto B., "Using mutual information for selecting features in supervised neural net learning," IEEE Transactions on Neural Networks, 5(4):537-550, 1994.

[5] Zhang, H. and Sun, G.. Feature selection using tabu search method. *Pattern Recognition*, 35: 701-711, 2002.

[6] Furey, T.S., Cristianini, N., Duffy, N., Bednarski, D.W., Schummer, M. and Haussler, D. (2000) Support vector machine classification and validation of cancer tissue samples using microarray expression data. Bioinformatics, 16, 906-914.

[7] Guyon, I., Weston, J., Barnhill, S. and Vapnik, V. (2002) Gene selection for cancer classification using support vector machines. Machine Learning, 46, 389-422.

[8] Lee, Y. and Lee, C.-K. (2003) Classification of multiple cancer types by multicategory support vector machines using gene expression data. Bioinformatics, 18, 1132-1139.

[9] Venter, G. and Sobieszczanski-Sobieski, J. Particle swarm optimization. Proceedings of the 43rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, Denver, CO. 2002.

[10] Stone, M., "Cross-Validation choice and assessment of statistical predictions" Journal of the Royal Statistical Society B, 36,, 1974, pp.111-147.

[11] Kennedy, J.; Eberhart, R.C., "A discrete binary version of the particle swarm algorithm", Systems, Man, and Cybernetics, 1997. 'Computational Cybernetics and Simulation'., 1997 IEEE International Conference on Volume 5, 12-15 Oct. 1997 Page(s):4104 - 4108 vol.5.

[12] Kennedy, J. and Eberhart, R.C., Particle swarm optimization, IN proceedings of the 1995 IEEE International Conference on Neural Networkds, volume 4, pages 1942-1948, Perth, Australia, 1995.

[13] Frieß, T., N. Cristianini, and C. Campbell (1998), "The Kernel-Adatron: a Fast and Simple Learning Procedure for Support Vector Machines," *Proceedings of the Fifteenth International Conference on Machine Learning*, pp. 188-196, 1998.

[14] Drucker, H., Burges, C., Kaufman, L., Smola, A. and Vapnik, V. (1997): Support Vector Regression Machines, In: Mozer, M., Jordan, M. and Petsche, T. (ed.), Neural Information Processing Systems, Vol. 9. MIT Press, Cambridge, MA, 155-161.

[15] Vapnik, V.N., The Nature of Statistical Learning Theory, Springer Verlag, 1995.

[16] Anlauf, J.K., and Biehl, M. (1989), The Adatron-an adaptive perceptron algorithm, Europhysics Letters, 10, 687-692.

[17] Opper, M. (1988). Learning Time of Neural Networks: Exact Solution for a Perceptron Algorithm. *Physical Review* A38:3824.

[18] Boser, B., Guyon, I., Vapnik, V. (1992), A training algorithm for optimal margin classifiers, Fifth Annual Workshop on Computational Learning Theory, ACM Press.

[19] Cortes, C., and Vapnik, V. (1995), Support Vector networks, Machine Learning 20:273-297.

[20] Colin, C. and Nello, C., "Simple Learning. Algorithms for Training Support Vector Machines," 1998.

[21] Scholkopf, B. and Smola, A. (2002) *Learning with Kernels: Support Vector Machines, Regularization, Optimization and Beyond*. MIT Press, Cambridge, MA.

[22] Murphy, P.M., and Aha, D.W. UCI Repository of Machine Learning Databases. technical report, Department of Information and Computer Science, University of California, Irvine, Calif., 1994. Available: http://www.ics.uci.edu/~mlearn/MLRepository.html.

[23] Oh, I.-S., Lee, J.-S., and Moon, B.-R. Hybrid Genetic Algorithms for Feature Selection. *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 26, no.11, Nov. 2004.

Table 1. Format of classification text problem

Datasets	Number of samples	Number of classes	Number of features	Value of r	Value of C
Vowel	990	11	10	2^4	2^{12}
Wine	178	3	13	2^4	2^{12}
WDBC	569	2	30	2^0	2^{12}
Ionosphere	*201/150*	2	34	2^{-1}	2^{12}
Sonar	*104/104*	2	60	2^0	2^{12}

Legends:
x/y: indicate that x and y represent the number of test and train samples, respectively.

Table 2. Accuracy of classification for tested data

Datasets	d*	SFS	PTA	SFFS	SGA	HGA (1)	HGA (2)	HGA (3)	HGA (4)	PSO-SVM	
										d*	%
Vowel (D=10)	2	62.02	62.02	62.02	62.02	62.02	62.02	NA	NA	7	99.49
	4	92.63	92.83	92.83	92.83	92.83	92.83	92.83	92.83		
	6	98.28	98.79	98.79	98.79	98.79	98.79	98.79	98.79		
	8	**99.70**	**99.70**	**99.70**	**99.70**	**99.70**	**99.70**	**99.70**	NA		
Wine (D=13)	3	93.82	93.82	93.82	93.82	93.82	93.82	93.82	NA	8	**100**
	5	94.38	94.38	94.94	95.51	95.51	95.51	95.51	95.51		
	8	95.51	95.51	95.51	95.51	95.51	95.51	95.51	95.51		
	10	92.13	92.13	92.70	92.70	92.70	92.70	92.70	92.70		
WDBC (D=30)	6	93.15	93.15	94.20	93.67	94.90	94.90	93.99	93.99	13	**95.61**
	12	92.62	92.97	94.20	94.38	94.38	94.38	94.38	94.38		
	18	94.02	94.20	94.20	93.85	94.20	94.20	94.20	94.20		
	24	92.44	93.50	93.85	93.85	93.85	93.85	93.85	93.85		
Ionosphere (D=34)	7	93.45	93.45	93.45	95.44	95.73	95.73	95.73	95.73	15	**97.33**
	14	90.88	92.59	93.79	94.87	95.73	95.73	95.73	95.73		
	20	90.03	92.02	92.88	94.30	94.30	94.30	94.02	94.30		
	27	89.17	91.17	90.88	91.45	91.45	91.45	91.45	91.45		
Sonar (D=60)	12	87.02	89.42	92.31	93.75	94.71	95.67	95.19	95.67	34	96.15
	24	89.90	90.87	93.75	95.67	96.63	96.63	**97.12**	**97.12**		
	36	88.46	91.83	93.27	95.67	96.15	96.15	96.15	96.15		
	48	91.82	92.31	91.35	92.79	92.79	93.27	93.27	93.27		

Legends: Highest values are in bold-type.