Journal of Physics: Conference Series

Parameter optimization of support vector machine based on improved grid algorithm

To cite this article: Shenghua Wang *et al* 2020 *J. Phys.: Conf. Ser.* **1693** 012108

Parameter optimization of support vector machine based on improved grid algorithm

Shenghua Wang, Li Dong, Huichun Hua

Department of Mathematics and Physics, North China Electric Power University, Baoding 071000, China

DongLi@ncepu.edu.cn

Abstract. Support vector machines SVM) is a very popular algorithm used widely in the classification problem. The basic idea of SVM is to construct two parallel hyperplanes to separate two classes of instances and maximize the distance between the hyperplanes. In this paper, we propose a new algorithm called SVM-ICC-DBTCPRP one to solve the parameter optimization problem by improving the grid algorithm used by Fayed and Atiya. In the new algorithm there are two main sub-algorithms that are used to preselect the support vectors for reducing the time of training and preselect the parameter range to reduce the number of training respectively. Six typical data sets are selected to verify the effectiveness of our algorithm. The computed results show that our algorithm has the obvious advantage on the aspect of elapsed time than the one of Fayed and Atiya.

1. Introduction

Support vector machine (SVM) was proposed by Vapnik [1] in the 1990s and now is one of the most successful classification algorithms [2]. Many works for SVM have been developed to improve its performance and application in recent years [3-5] based on its advantages such as few tuning parameters, fast classification and high generalization capability.

The basic idea of standard SVM is to construct two hyperplanes to separate two classes of instances and maximize the distance between the hyperplanes, which results in a quadratic programming problem on the parameters involved in SVM. For obtaining the best-performing parallel hyperplanes of separating two classes, the parameters have to be carefully chosen. In fact, it is a difficulty although only a few tuning parameters exist in SVM. Exhaustive search method, the grid-search method, is the most commonly used approach, but it is time-consuming. Many approaches for avoiding an exhaustive grid-search have been obtained in the literature, for example, see [6-12]. However, in these methods, there is a disadvantage of the local convergence. In addition, the numerical methods also are used to find the optimal parameters in the literature. For example, in [13] Adankon and Cheriet proposed a method based on an approximation of the gradient of the empirical error, along with incremental learning, which reduced the resources required both in terms of processing time and of storage space. Zhang et al. [14] proposed a hybrid method in which the inter-cluster distance in the feature space (ICDF) is used to determine a small search interval from a larger kernel parameter search space, while a hybrid of the barebones particle swarm optimization and differential evolution (BBDE) is used to search the optimal parameter combination in the new search space.

However, in general, these methods can not guarantee to find the global optimal solution due to the non-convexity of the generalization bounds. Recently, for avoiding falling into the set of local optimal solutions, some authors used the evolutionary methods to find the optimal parameters. For example, Huang and Wang [15] proposed a method for feature selection and parameter optimization based on genetic algorithm. Lin et al. [16] used a simulated annealing algorithm to find the optimal parameters. Guo et al. [17] modified a method based on particle swarm optimization for the selection of the parameters. For the more the evolutionary methods, the readers may refer to [18-22]. Although the evolutionary methods can find the global optimal parameters from the theory, an exhaustive search for the parameters involved in the methods has to be done, which may need a large time cost.

The most direct and reliable method is the grid algorithm [23], which will traverse all parameter combinations. However, since the time complexity of the quadratic programming problem makes it impractical to be applied to large data sets, the complexity is further increased when an exhaustive grid search is used to find its optimal parameters (the kernel parameters and the penalty parameter, C). Very recently, to reduce the complexity, Fayed and Atiya [23] proposed an accelerated grid algorithm based on the C-condensation algorithm (in short, CC algoirthm), which greatly improved the speed of parameter optimization almost without the loss of accuracy. In [23], the authors prune the data points by removing the ones that have an extremely small chance of becoming support vectors. This is accomplished by using the support vectors obtained from the training of an SVM with a smaller value of C as the training patterns for an SVM with a slightly larger value. This can serve in reducing the grid-search time for the standard SVM and for the approximate methods. However, it observes that the accelerated grid algorithm introduced by Fayed and Atiya [23] sensitively depends on the value of C. Especially, the efficiency of the CC algorithm is not up to expectation when the value of C is smaller, whereas it may shrink excessively when the value of C is larger. In this paper, we improve the method of Fayed and Atiya [23] by combining the CC algorithm with other algorithms to find the global optimal parameters. Experiments showed that the training time and accuracy of our method are faster and better than the one of Fayed and Atiya.

This article is organized as follows. Section 2 briefly reviews the SVM algorithm and grid algorithm. In section 3, an improved CC algorithm based on Adaboost [24] is proposed to preselect support vector after analyzing the shortcomings of the CC algorithm. The basic principle of the parameter range preselection algorithm is analyzed in section 4, based on which the DBTCPRP algorithm has been constructed. Section 5 combines the algorithm of section 3 and section 4 to form the SVM-ICC-DBTCPRP algorithm. In section 6, six typical data sets are selected to verify the effectiveness of our algorithm. The conclusions are presented in Section 7.

2. SVM and grid algorithm

Consider the binary classification problem with the training set $\{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \cdots, (\boldsymbol{x}_n, y_n)\}$, where $\boldsymbol{x_i} \in R^n$, $y_i \in \{1, -1\}$. Suppose the optimal classification interface is $w^T x + b = 0$, then the SVM is constructed by solving the following quadratic optimization problem [1]:

$$\begin{aligned} &\min_{\boldsymbol{w}} f(\boldsymbol{w}, \boldsymbol{\xi}) = \frac{1}{2} \boldsymbol{w}^T \boldsymbol{w} + C \sum_{i=1}^{n} \xi_i \\ &\text{subject to:} \\ &y_i \left(\boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}_i) + b \right) \geq 1 - \xi_i, \\ &\xi_i \geq 0, \end{aligned} \tag{1}$$

where $\phi: R^n \to R^N$ maps $\boldsymbol{x_i}$ into a higher dimensional space to transform the problem from linearly indivisible to linearly separable, ξ_i is slack variable measuring the classification loss of examples, C is the penalty parameter. The Lagrangian dual problem is given by

$$\begin{aligned}&\max_{\lambda}\left(\sum_{i=1}^{n}\lambda_i-\frac{1}{2}\sum_{j=1}^{n}\sum_{i=1}^{n}\lambda_j\lambda_i y_i y_j \boldsymbol{K}(\boldsymbol{x}_i,\boldsymbol{x}_j)\right)\\&\text{subject to:}\\&\sum_{i=1}^{n}\lambda_i y_i=0,\\&C\ge\lambda_i\ge 0,\end{aligned}\tag{2}$$

where $K\left(\boldsymbol{x}_i,\boldsymbol{x}_j\right)=\langle\phi(\boldsymbol{x}_i),\phi(\boldsymbol{x}_j)\rangle=\phi^T(\boldsymbol{x}_i)\cdot\phi(\boldsymbol{x}_j)$ is the kernel function.

Suppose the solution of (2) is λ^*. The optimal solution of (1) is

$$\begin{aligned}&\boldsymbol{w}^*=\sum\nolimits_{i=1}^{n}\lambda_i^* y_i\boldsymbol{x}_i,\\&b^*=-\frac{\max_{i:y_i=-1}w^*x_i+\min_{i:y_i=1}w^*x_i}{2}.\end{aligned}\tag{3}$$

from which we can derive the decision function

$$f\left(\boldsymbol{x}\right)=\operatorname{sgn}\left[\sum\lambda_i^* y_i\boldsymbol{K}(\boldsymbol{x}_i,\boldsymbol{x})+b^*\right]\tag{4}$$

The KKT condition of the above process is

$$\begin{cases}\lambda_i\ge 0,\\ y_i f(x_i)-1\ge 0,\\ \lambda_i(y_i f(x_i)-1)=0.\end{cases}\tag{5}$$

The vector $\boldsymbol{x}_i$ that satisfies $w^{*T}\boldsymbol{x}_i+b^*=\pm 1$ is called the support vector. By (4) it is easy to see that the support vector $\boldsymbol{x}_i$ does not work in the decision function when λ_i=0. If $\lambda_i\ne 0$, it can be derived from (5) that $y_i f(\boldsymbol{x}_i)-1=0$. Under this case, one has $f(\boldsymbol{x}_i)=\pm 1$ for $y_i=\pm 1$. This fact means that only support vectors with $\lambda_i\ne 0$ play a role in the final model. So selecting the support vectors with $\lambda_i\ne 0$ in advance can greatly reduce the training time.

The grid algorithm is an exhaustive search method that sorts and combines all possible values of parameter to generate a grid. After trying all the nodes of the grid, a suitable classifier will be returned, which is reliable but time-consuming. So for saving the training time, it is a good method that constructs in advance a set of possible parameter values where the optimal parameter value can be found.

Through the analysis of SVM and grid algorithm, it is easy to see that it can improve the grid algorithm of SVM by preselecting the support vectors and reducing the range of parameter values. In this paper, we design a new algorithm to improve the efficiency of parameter optimization of SVM. In this new algorithm, the support vectors are preselected and the range of parameter values is reduced before performing a standard grid algorithm.

3. Support vector preselection algorithm based on improved C-condensation algorithm

It is known that the number of support vectors is larger when the value of C is smaller and decreases rapidly with the increasing of the value of C. In fact, the CC algorithm is highly sensitive to the value of C. For correcting this shortcoming, we propose an improved CC algorithm (in short, ICC algorithm) by combining a support vector preselecting algorithm based on Adaboost [24] with the CC algorithm. We describe the ICC algorithm as follows.

Algorithm 1: ICC
Input: Training set T, kernel function parameter set Γ, Set of penalty coefficients $C:\{C_1,C_2,\cdots C_n\}$
Output: Optimal parameter γ^*,C^* and the final model M
Initialization: The number of sharded data sets: *SplitNum*, the number of support vectors to stop ICC: *EndCCNum*, *C* of the primary preselection algorithm
Split T to $\{T_1,T_2,\cdots,T_{SplitNum}\}$
for $i=1,2,\cdots,SplitNum$ **do**

Train an SVM model, M_i, using γ and C with T_i
SV_i = Support vectors of M_i
end for
T= $[SV_1, SV_2, \cdots, SV_{SplitNum}]$
for each ($\gamma \in \Gamma$) **do**
T_i=T
for $i = 1, 2, \cdots, k$ **do**
Train an SVM model, M_i, using γ and C_i with T_i
SV_i = Support vectors of M_i
CV_i = cross-validation precision rate for M_i
if CV_i > CV **then**
CV = CV_i
$\gamma^* = \gamma$;
$C^* = C_i$;
end if
if N_{SVi} > *EndCCNum* **then**
T_{i+1} = SV_i
end if
end for
end for
Train an SVM model, M, using γ^* and C^* with T

4. Parameter range preselection algorithm based on DBTC

The basic idea of constructing the parameter range preselection algorithm is to select an index that can reflect the classification effect, traverse the kernel parameters in a large range and calculate the index value accordingly, and finally select the parameter range with the optimal solution of the index value as the center. So the index should meet the two points: it can reflect the classification effect and has a global optimal solution.

In [25], it is proved that the distance between the means of two classes (in short, DBTC) can measure the classification effect and has a global maximum. Hence DBTC satisfies the two points mentioned above. However, since DBTC is taken as the only indicator, the more important index, i.e., precision rate, is neglected in [25]. In this paper, we first preselect the range of kernel parameters by a parameter range preselection algorithm constructed based on DBTC (in short, DBTCPRP algorithm), then use the grid algorithm to select the model with the precision rate.

The DBTCPRP algorithm works by the manner that for the binary-classification data set, use the fixed C value, traverse all kernel function parameters, and calculate DBTC accordingly. In the DBTCPRP algorithm, if the kernel parameter corresponding to the maximum value of DBTC is 2^p, then the preselected parameter range is $\{2^{p-1}, 2^p, 2^{p+1}\}$. For the multi-classification data set, calculate the DBTC value of each two classes, and the kernel parameter corresponding to the DBTC maximum value between each two classes is put as $2^m, \cdots, 2^n$. If the minimum is 2^p, the maximum is 2^q, then the preselected range is $\{2^p, 2^{p+1} \cdots, 2^q\}$. We describe the DBTCPRP algorithm as follows.

Algorithm 2: DBTCPRP

Input: Training set T, kernel function parameter set $\Gamma : \{\gamma_1, \gamma_2, \cdots, \gamma_k\}$, and the penalty coefficient C
Output: Kernel parameter range Γ^*
Initialization: A zero column vector of length N(the number of classes): ***DBTC***, a null column vector of length N: ***P***.
if T is a binary-classification data set **then**
for $i = 1, 2, \cdots, k$ **do**
Train an SVM model, M_i, using γ_i and C with T
Calculate $DBTC_i$

if $DBTC_i$>DBTC **then**
DBTC= $DBTC_i$, $p=i$
end if
end for
$\Gamma^*=\{\gamma^{p-1},\gamma^{p},\gamma^{p+1}\}$
else T is a multi-classification data set **then**
for $i=1,2,\cdots,k$ **do**
Train an SVM model, M_i using γ_i and C with T
for $m=1,2,\cdots,Number\ of\ class$ **do**
for $n=1,2,\cdots,Number\ of\ class$ **do**
Calculate $DBTC_{m,n}$
if $DBTC_{m,n}>DBTC[m]$ **then**
$DBTC[m]=DBTC_{m,n}$
$P[m]=i$
end if
end for
end for
end for
p=min(P),q=max(P)
$\Gamma^*=\{\gamma^{p},\gamma^{p+1},\cdots,\gamma^{q}\}$
end if

5. SVM-ICC-DBTCPR algorithm

The ICC algorithm and DBTCPRP algorithm are combined to form the SVM-ICC-DBTCPRP algorithm in which the DBTCPRP algorithm is first used to preselect the parameters, then train the preselected parameters by the ICC algorithm.

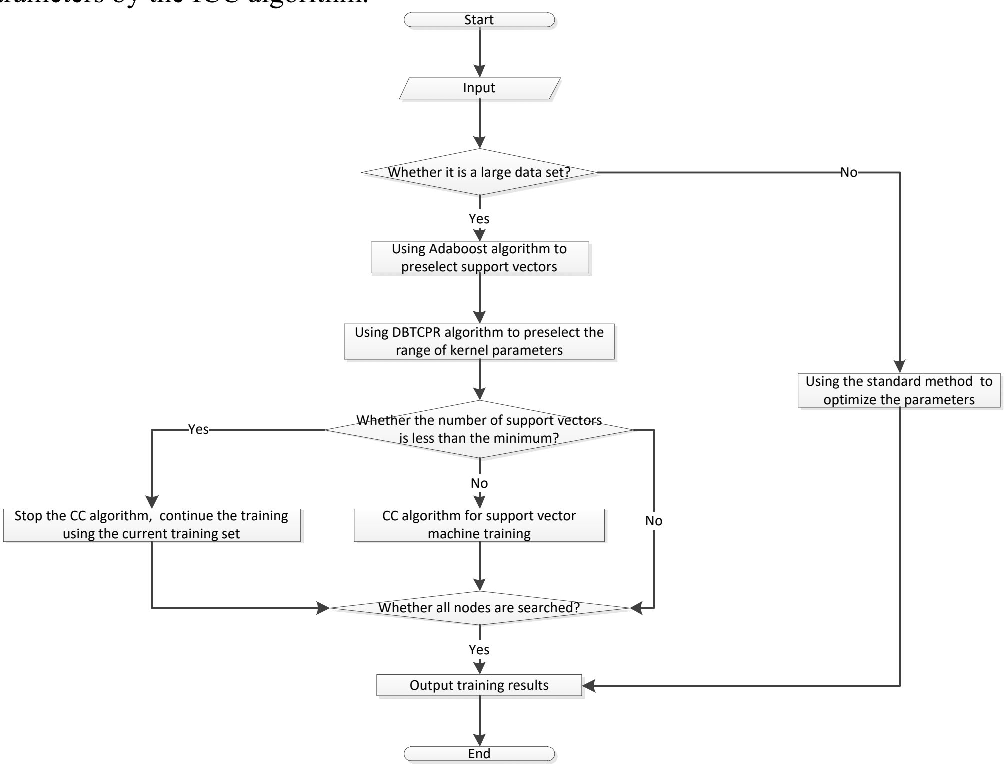

Figure 1. The flow of SVM-ICC-DBTCPR algorithm

6. Experiment

To verify the effectiveness of the algorithm, multiple data sets were selected from LIBSVM and UCI for experiments. They are Parabola Data Set(1000), Parabola Data Set(10000), Bank Marketing Data Set, Avila Data Set, Audit data set, Skin data set, and Letter Data Set. These data sets cover multiple dimensions such as binary-classification and multi-classification, few-attribute and multi-attribute, and data size. They can prove the validity of the thesis method from many angles. The detailed characteristics of the datasets are shown in table 1.

In the experiment, the gauss kernel function is used for training, and the search range of penalty coefficient C is $\{2^{-15},2^{-14},\cdots,2^{15}\}$, but for a very large number of data sets only $\{2^{-5},2^{-4},\cdots,2^{15}\}$ is searched. The search range of kernel parameters γ is $\{2^{-15},2^{-14},\cdots,2^{15}\}$, or $\{2^{-7},2^{-14},\cdots,2^{7}\}$ for large data set. Four algorithms, Standard SVM, SVM-PSO, SVM-CC, and SVM-ICC-DBTCPRPC were used for analysis in the three dimensions, precision rate, recall rate, time of training and testing. The test method was five-folds cross-validation. The results are shown in Table 2 and Table 3.

Table 1. Characteristics of data sets

	Number of training samples	Number of attributes	Number of classes
Parabola Data Set(1000)	1000	2	2
Parabola Data Set(10000)	10000	2	2
Bank Marketing Data Set	4521	17	2
Avila Data Set	20867	10	12
Audit data set	777	18	2
Skin data set	122529	3	2
Letter Data Set	15000	16	26

Table 2 Experiment time, precision rate and recall rate (the best results are shown in bold)

	Standard SVM			SVM-PSO		
	Time(s)	precision rate	recall rate	Time(s)	precision rate	recall rate
Parabola Data Set(1000)	217.51	**99.87%**	**99.91%**	97.25	99.67%	99.81%
Parabola Data Set(10000)	19044.89	**99.91%**	**99.91%**	10002.72	99.81%	99.90%
Bank Marketing Data Set	4326.35	90.26%	89.79%	1657.23	90.26%	89.79%
Avila Data Set	57698.12	99.32%	99.12%	24113.20	99.22%	99.00%
Audit data set	193.94	93.10%	92.41%	100.72	92.72%	92.31%
Skin data set	235267.25	**99.98%**	**99.96%**	92156.12	99.95%	**99.96%**

	SVM-CC			SVM-ICC-DBTCPRP		
	Time(s)	precision rate	recall rate	Time(s)	precision rate	recall rate
Parabola Data Set(1000)	52.12	**99.87%**	**99.91%**	**9.45**	**99.87%**	**99.91%**
Parabola Data Set(10000)	8921.20	86.33%	82.36%	**184.08**	97.91%	97.80%
Bank Marketing Data Set	1926	90.24%	89.87%	**222.10**	**90.32%**	**90.02%**
Avila Data Set	13256.20	99.32%	99.22%	**5142.02**	**99.42%**	**99.32%**
Audit data set	87.92	92.12%	92.01%	**28.23**	**92.11%**	**92.59%**
Skin data set	58156.92	99.95%	**99.96%**	**10812.01**	**99.98%**	**99.96%**

Table 3. Time-saving rate of the experiment

	SVM-PSO	SVM-CC	SVM-ICC-DBTC
Parabola Data Set(1000)	55.29%	76.04%	95.66%
Parabola Data Set(10000)	47.48%	53.16%	99.03%
Bank Marketing Data Set	61.70%	55.48%	94.18%
Avila Data Set	58.21%	77.02%	91.09%
Audit data set	48.07%	54.67%	85.44%
Skin data set	60.83%	75.28%	95.40%

It can be seen from Table 2 and Table 3 that the SVM-ICC-DBTCPRP algorithm hardly causes a decrease in accuracy and recall rate for most data sets, and even has some improvement, while the training and test time have a great range of improvement, especially for large-scale data, the time-saving rate can reach more than 95%, indicating that the algorithm is very effective.

7. Conclusion

This paper presents an accelerated grid algorithm called the SVM-ICC-DBTCPRP algorithm to speed up the parameter optimization process of SVM. In this algorithm, we use the ICC algorithm and DBTCPRP algorithm to preselect support vector and to preselect the parameter range, respectively. The SVM-ICC-DBTCPRP algorithm improves the efficiency of support vector machine parameter optimization. By the experiments, we find that the SVM-ICC-DBTCPRP algorithm can reduce the parameter optimization time by more than 95% without affecting the classification effect for large data sets, and it has good adaptability to large data.

Acknowledgments

The paper is supported by Research and Practice Project on Education and Teaching Reform in Hebei Province（Grant No. 2017GJJG358）

References

[1] V. Vapnik. Statistical Learning Theory[M]. DBLP, 1998.
[2] I. Steinwart, A. Christmann. Support vector machines.[J]. Information science & statistics, 2008, 158(18):1-28.
[3] A.K. Rajendran. Parallel support vector machines for multi-category classification of large scale data[J]. Dissertations & Theses Gradworks, 2007.
[4] J. Zhang, Z. Li, J. Yang, A parallel SVM training algorithm on large-scale classification problems[C]// Proceedings of 2005 International Conference on Machine Learning and Cybernetics, 2005: 1637-1641.
[5] Y. Xu, Maximum margin of twin spheres support vector machine for imbalanced data classification[C]// IEEE Transactions on Cybernetics, 2017,47(6): 1540-1550.
[6] O. Chapelle, V. Vapnik, O. Bousquet, et al. Choosing multiple parameters for support vector machines, Machine Learning, 2002,46(1): 131-159.
[7] S. Keerthi, Efficient tuning of SVM hyperparameters using radius/margin bound and iterative algorithms[J]. IEEE Transactions on Neural Networks, 2002, 13(5)：1225-1229.
[8] G.H. Bakir, L. Bottou, J. Weston, Breaking SVM complexity with cross-training[C]// Advances in Neural Information Processing Systems, Vancouver, Canada, 2004, 81–88.
[9] T. Hastie, S. Rosset, R. Tibshirani, J. Zhu, The entire regularization path for the support vector machine[J]. journal of machine learning research, 2004, 5:1391-1415.
[10] S. Keerthi, V. Sindhwani, O. Chapelle, An efficient method for gradient based adaptation of hyperparameters in SVM models[C]// Advances in Neural Information Processing Systems 19, Max-Planck-Gesellschaft, MIT Press, Cambridge, MA, USA, 2007, 673–680.
[11] S. Li, M. Tan, Tuning SVM parameters by using a hybrid CLPSO-BFGS algorithm[J]. Neurocomputing, 2010, 73(10-12): 2089-2096.
[12] T. Krueger, D. Panknin, M. Braun, Fast cross-validation via sequential testing [J].Journal of Machine Learning Research, 2015, 16(1): 1103–1155.
[13] M.M. Adankon, M. Cheriet, Optimizing resources in model selection for support vector machine [J].Pattern Recognition.2007, 40(3)：953-963.
[14] X.Y. Zhang, D.Y. Qiu, F. Chen, Support vector machine with parameter optimization by a novel hybrid method and its application to fault diagnosis[J]. Neurocomputing, 2015, 641–651
[15] C.L. Huang, C.J. Wang, A GA-based feature selection and parameters optimization for support vector machines[J]. Expert Systems with Applications, 2006, 31(2):231-240.
[16] S.W. Lin, Z.J. Lee, S.C. Chen, et al. Parameter determination of support vector machine and feature selection using simulated annealing approach[J]. Applied soft computing, 2008, 8(4): 1505-1512.
[17] X.C. Guo, J.H. Yang, C.G. Wu, et al. A novel LS- SVMs hyper-parameter selection based on particle swarm optimization[J]. Neurocomputing, 2008,71(16–18):3211–3215.
[18] Y. Tian, M. Fu, F. Wu, Steel plates fault diagnosis on the basis of support vector machines[J].

Neurocomputing, 2015, 151: 296-303.
[19] M. Fu, Y. Tian, F. Wu, Step-wise support vector machines for classification of overlapping samples[J]. Neurocomputing, 2015, 155: 159-166.
[20] Z. Xue, P. Du, H. Su, Harmonic analysis for hyperspectral image classification integrated with PSO optimized SVM[J]. IEEE Journal of Selected Topics in Applied Earth Observations and Re-mote Sensing, 2014, 7(6): 2131-2146.
[21] F. Imbault, K. Lebart, A stochastic optimization approach for parameter tuning of support vector machines[C]// Proceedings of the 17th International Conference on IEEE, 2004, 4:597-600.
[22] L. Shen, H. Chen, Z. Yu, et al. Evolving support vector machines using fruit fly optimization for medical data classification[J]. Knowledge Based Systems, 2016, 96: 61-75.
[23] H.A. Fayedm, A.F. Atiya, Speed up grid-search for parameter selection of support vector machines[J]. Applied Soft Computing Journal, 2019, 80: 202–210.
[24] R.E. Schapire, The strength of weak learnability[J]. Machine Learning, 1990, 5(2):197-227.
[25] J. Sun, C. Zheng, X. Li, et al. Analysis of the Distance Between Two Classes for Tuning SVM Hyperparameters[J]. Neural Networks IEEE Transactions on, 2010, 21(2): 305-318.