Research on SVM Algorithm with Particle Swarm Optimization

Yong-jie Zhai [1] Hai-li Li [1] Qian Zhou [2]

[1] Control of Science and Engineering College,
North China ElectricPower University, Baoding 071003
[2] Energy and Power Engineering Institute,
North China Electric Power University,Beijing 102206

Abstract

Support Vector Machines (SVM) is a practical algorithm that has been widely used in many areas. To guarantee its satisfying performance, it is important to set appropriate parameters of SVM algorithm. Sequential Minimal Optimization (SMO) is an effective training algorithm belonging to SVM, i.e.LS_SVM. Therefore, on the basis of the SMO algorithm and LS_SVM, which integrates SMO algorithm and LS_SVM, we introduced Particle Swarm Optimization (PSO) algorithm, and utilized an example to certify its validity. PSO is proposed to deal with the large amount of data, and the simulation results showed the effectiveness of this method.

Keywords: SVM; SMO; LS-SVM; PSO; selection of parameters

1. Introduction

Based on the statistical learning property and theoretical principles, Support Vector Machine has proved to be a practical algorithm[1]which has been widely used in pattern recognition, function fitting, forecasting and other areas. Its mathematical model can be deemed as a blending quadratic programming problem.

Sequential Minimal Optimization algorithm is first proposed by Platt [2] and it is a fast and effective solution to such problem. The results of the SMO algorithm vary greatly due to different selection of parameters. As a result, the choice of parameters in SMO becomes a crucial issue [3], which is the same case as it is in LS-SVM [4] [5].

As a simple and intelligent optimization algorithm, Particle Swarm Optimization (PSO) has been developed rapidly in recent years. In this paper, PSO algorithm is combined with SMO and LS-SVM to optimize the parameters of SMO and LS-SVM. As can be seen from the simulation results, the method works effectively.

2. Parameters Selection in SVM

2.1. Selection of the Kernel Function

When solving regressive problems by support vector machines, it is necessary to replace the inner product by an appropriate kernel function based on the characteristics of the problem, so as to transform the calculation of high-dimension inner product into that of a low-dimension kernel function implicitly. And it aims to solve the problem of "curse of dimensionality" while maintaining the accuracy of the problems. The kernel function should not only meet Mercer conditions theoretically, but also

reflect the distribution of sample data in practical application training. Therefore, while the support vector machine regression is to be used in a specific problem, the choice of an appropriate kernel function is the key factor[5]. However, there is still no specific method to find a proper kernel function for the corresponding problem. Usually there exist three types of kernel functions: polynomial kernel function, RBF kernel function and Sigmoid function.

The choice of kernel function determines the characteristics of the space structure. As to the polynomial kernel function, when the spatial dimension is very large, the value of d becomes great. It gives rise to the increase of calculation complexity that in some cases correct results cannot be achieved. Sigmoid kernel function has some limitations, because the parameters β and C in the function can only satisfy Mercer conditions when they are given of certain values. RBF kernel function is a universal kernel function; after the selection of the relevant parameters, it can be applied to arbitrary distributive samples. Polynomial kernel function embraces two controllable parameters d and θ while RBF kernel function has only one. Since the choice of parameters reflects the complexity of the model, RBF kernel function is the relatively wiser choice.

In conclusion, RBF kernel function is generally applied in the Support Vector Machine [6].

2.2. Impacts of the Parameters

The generalization ability of SVM algorithm depends on a set of parameters, including the penalty actor$C$, the estimated accuracy σ and the RBF kernel parameter δ

- Impact of the penalty factor C: The aim of the penalty factor C is to modulate the ratio between the space credibility and the experience risk in a certain digital space, so as to attain the best generalization ability for the machine model. Different digital space requires different optimal parameter C. In certain digital space, a small value of C could lead to weak punishment for the experience error, little complexity of learning machine yet large experience risk, or vice versa. The former is called "less-trained", and the latter is called "over-trained". When C exceeds a certain value, the complexity of SVM achieves the maximum tolerated by the data space, and the experience risks and generalization ability would not change any more. In each digital space there exists at least one suitable C to achieve the best generalization ability.
- Impacts of the RBF kernel parameter: RBF kernel parameter reflects the distribution or scope characteristics of training sample data, which defines the width of local neighborhood. A large mea10ns relatively little variance.
- Impacts of the estimated accuracy: The relaxation factor determines the width of the non-sensitive zone, and affects on the number of support vectors. By selecting a small value, the regression estimation becomes greatly accurate. However, in that case, the number of the support vectors and the complexity of SVM algorithm would both increase. By selecting a great value, regression estimation becomes less accurate, but the number of the support vectors could decrease and the complexity of SVM algorithm would be weakened. Similar to the relaxation factor , the estimated accuracy

has the same impacts on the system. Therefore, in the standard support vector machines, parameters C and determine the complexity of the model through different ways [6].

2.3. Methods in Choosing Model Parameters

The choice of SVM parameters has brought wide concern and many approaches have been proposed during recent years, such as enumeration, the three-step search strategy, optimization nuclear parameters algorithm (OMSA), max-min nuclear parameter selection, intelligent optimization and so forth. Some of these methods have certain limitations. For example, enumeration is time-consuming; if the number of parameters is over two, this method is hardly practicable. Moreover, people should be experienced to practise this method which makes it difficult for a novice of SVM.

Intelligent optimization has become more prevalent in recent years, involving genetic algorithm, PSO algorithm, and colony algorithm etc.. During the optimization process, the key part is the selection of the fitness function. In the SMO algorithm, it is common to choose the value representing the estimated generalization ability as the fitness function. The generalization ability can be estimated by several methods as follows:The choice of SVM parameters has brought wide concern and many approaches have been proposed during recent years, such as enumeration, the three-step search strategy, optimization nuclear parameters algorithm (OMSA), max-min nuclear parameter selection, intelligent optimization and so forth. Some of these methods have certain limitations. For example, enumeration is time-consuming; if the number of parameters is over two, this method is hardly practicable. Moreover, people should be experienced to practise this method which makes it difficult for a novice of SVM.

Intelligent optimization has become more prevalent in recent years, involving genetic algorithm, PSO algorithm, and colony algorithm etc.. During the optimization process, the key part is the selection of the fitness function. In the SMO algorithm, it is common to choose the value representing the estimated generalization ability as the fitness function. The generalization ability can be estimated by several methods as follows:

- K-fold Method:This method chooses k-fold cross-validation error to estimate the generalization ability of SVM model. Specifically, the data samples should be divided into K parts completely independently from each other. One of them is removed as a test sample, and the remaining K-1 parts are taken as training samples. This cycle repeats until all the K error records are taken. The average value of the K error records is the k-fold cross-validation error. The smaller the value is the stronger its generalization ability is and the more appropriate the parameters are.
- Leave-one-out Method:The method is a special case of k-fold method. In this method, one data sample is removed, and the remaining samples are taken for training and the sentencing guideline acquisition. The sentencing guidelines are utilized to classify the removed sample. If the classification is wrong, a left error would be generated [7]. Each sample is tested by the sentencing guidelines determined by the remaining training samples and the

recorded results. If the result is right, mark 0; otherwise mark 1. The average value of all the test records is used to estimate the generalization ability of the model. The smaller the value is, the stronger the generalization ability is. Although this method has good estimation performance and is applicable to all kinds of classifications, the efficiency of the algorithm is relatively low. As to l samples, there would be l times of studying and classification decisions. With the increase of samples, the required computation aggrandizes dramatically.

- Support Vectors Counting Method: Support Vectors Counting method is relatively simple; the method is to count the number of support vectors in SVM. Suppose N signifies the number of all the support vectors in SVM, l is the number of samples for the training, then the generalization ability estimated by Support Vectors method Counting is $\frac{N}{l}$.

Statistical Learning Theory indicates that if the training samples can be completely divided by the optimal hyperplanes, the supremum of the expectation risk of SVM can be defined as:

$$E[R_e] \le \frac{N}{l-1} \quad (1)$$

Where R_e is the forecast error rate for unknown samples [7].

Therefore, the smaller *N* becomes, the less expectation risk there exists. And its prediction ability guided by the samples could be well-built.

3. The Implementation of SMO Algorithm Combining with Particle Swarm Optimization

3.1. Principles of PSO Algorithm

PSO algorithm embraces the excellent advantages of simplicity and fast speed.PSO algorithm was first proposed by the American social psychologist James Kennedy and the electrical engineer Russell Eberhart in 1995 as an optimization algorithm of colony intelligence. The idea was inspired by their early modeling and simulation research of the behaviors of the bird colony, and the model had been mainly used in biologist Frank Heppner's model. It is similar to other evolutionary algorithms, in terms of "colony" and "evolution" characteristics, and the operation is based on the values of the individual adaptation. generally applied.

3.2. Principles of Two-layer Optimization Structure Based on PSO Algorithms

The process of parameter optimization based on PSO algorithm is as follows: Firstly a set of parameter values is generated randomly, and SMO is trained; then another set of parameters is chosen according to the target value; SMO training repeats until a satisfactory training model is attained. This method is satisfactorily applicable by using the two-layer optimization structure. The optimization process is implemented with two levels. The goal of the upper level is to find a set of optimal parameters for the SMO algorithm; the lower level's task is to get the training model by SMO using the parameters acquired from the upper level.

3.3. Specific Steps

Take RBF kernel function as the kernel function. The parameters to be optimized are: the penalty factor C, RBF kernel parameter and the estimated accuracy . Use the k-fold method to estimate the generalization ability. The basic steps are stated as follows:

① Input the sample data, and set a group of parameters { C, δ , σ }randomly as the initial position of particles;

② Divide all the samples into k independent parts: $S_1, S_2, \cdots S_k$;

③ Train SMO based on the parameters { C, δ , σ }, then calculate the k-fold cross-validation error.

i Initialize i = 1;

ii S_i is left to test the model, and a training set is composed of the remaining samples. Implement SMO training;

iii Calculate the generalization error of the subset S_i , then let set i = i +1, repeat step ii until i = k +1;

iv Calculate the average value of the generalization errors and get the k-fold cross-validation error;

④Take k-fold cross-validation error as the fitness value and record the best positions of the individual particles and group samples corresponding to the best fitness value: p_{best} and g_{best} , and search for the better parameters {C, δ , σ } based on the PSO optimization algorithm.

⑤ Repeat step ② until the largest iteration number is reached or the ending conditions are met.

4. The Implementation of LS-SVM Algorithm with Particle Swarm Optimization

Although LS-SVM lab has the function to optimize the parameters, we can use PSO algorithm in addition which is similar to the SMO algorithm. RBF kernel function is taken as the kernel function. The parameters to be optimized are the penalty factor C and RBF kernel parameter . The steps can be summarized as follows:

① Input the sample data, set a group of parameters {C, δ } randomly as the initial position of particles;

② Implement LS-SVM training based on the parameters {C, δ }, then calculate the mean-squared error of all the training samples.

③ Take the training mean-squared error as the fitness value and record the best positions of the individual particles and group samples corresponding to the best fitness value: p_{best} and g_{best} , and search for better parameters {C, δ } based on the PSO optimization equation.

④ Repeat step ② until the largest iteration number is reached or the ending conditions are met.

5. Simulation

In this section, two application examples are introduced to verify the algorithms proposed above. PSO optimization proved effective in the simulation results.

5.1. Example1:Nonlinear Dynamic System Identification

The plant is described as

$$y(t+1) = \frac{y(t)y(t-1)[y(t)+2.5]}{1+y^2(t)+y^2(t-1)} + u(t) \tag{2}$$

$$y(0) = 0, y(1) = 0$$

$$u(t) = \sin(\frac{2\pi t}{25})$$

The model can be described as

$$\hat{y}(t+1) = f(y(t), y(t-1), u(t)) \tag{3}$$

There are three inputs and one output in the model. In our simulation, 600 pairs of input/output data were generated. The first 400 pairs were chosen as training data, the last 200 data pairs were used as testing data.

Section A

The mean-square error of all the training samples are used as the fitness value, and it is defined as:

$$J = \text{RMSE\_train} = \sqrt{\frac{\sum_{i=1}^{n} e_i^2}{n}} \quad (4)$$

By using MATLAB as the platform, we got the SMO parameters:

C= 0.9169 δ = 0.4289 σ = 0.4449

The training error and the testing error are as follows:

RMSE_train = 0.0015

RMSE_test = 2.2711e-004

To prove the superiority of the algorithm, the results generated in this paper were compared with the results in paper [7]. The results can be found in Table.1.

It can be found in Table 1 that the effects of the SMO algorithm based on PSO optimization are much better than that of other algorithms mentioned in [8].

Section B

In this part, we used LS-SVM lab to identify the nonlinear dynamic system in Example 1. We identified the system using LS-SVM, which was optimized by its embedded function and by PSO algorithm, respectively.

- LS-SVM Identification Using its embedded optimization function:
  By using MATLAB as the platform, we got the LS-SVM parameters:
  C=455.0876 δ = 0.3965718
  RMSE_train=0.0016
  RMSE_test =6.7481e-004
  From the results above, we found the performance was a little less satisfactory than that of SMO optimized by PSO.
- LS-SVM Identification Using the PSO Algorithm
  Through the simulation in MATLAB, we got the parameters as follows:
  C=850.0000 δ = 0.0148
  The training error and the testing error are as follows:
  RMSE_train =2.7103e-004
  RMSE_test =2.2182e-004

All the results are listed in Table.1. As can be seen from Table 1, PSO algorithm offers a nice solution to the problem of the selection of the SVM parameters.

Table 1: The Results of Different Algorithms for Nonlinear Dynamic System Identification

Algorithm	RMSE_train	RMSE_test
DFNN	0.0283	_a
GDFNN	0.0241	_a
SOFNN	0.0157	0.0151
SMO with PSO	0.0015	2.2711e -004
LS-SVM(optimized by its own function)	0.0016	6.7481e-004
LS-SVM(optimize by PSO)	2.7103e-004	2.2182e-004

5.2. Example2: Mackey-Glass Series

Since Glass and Mackey found the chaos phenomenon in the time-delay system in 1977, the time-delay chaotic system has been commonly used as a standard test model in nonlinear systems. Mackey Glass-time series is defined as:

$$x(t+1) = (1-a)x(t) + \frac{bx(t-\tau)}{1+x^{10}(t-\tau)} \quad (5)$$

$a = 0.1, b = 0.2, \tau = 17, x(0) = 1.2.$

If $\tau \geq 17$, the chaotic property would be in display, and the greater the value is, the more chaotic the system would be. In this paper, we generated 2000 data according to the definition. The first 1000 data were taken as training samples, and the left 1000 data were taken as test samples. The purpose of forecasting Mackey-Glass series is to predict the latest output value based on the sample values generated before. Select the forecasting model as:

$$\hat{x}(t+6) = f(x(t), x(t-6), x(t-12), x(t-18)) \quad (6)$$

In this paper, we forecasted the sixth step output value. The model is built up of four inputs and one output.

5.2.1. The Method to Set a Large Amount of Data Based on PSO

Due to the large amount of data in this example, the training process would spend too much time during the optimization process. To solve this problem, a method is proposed in this paper. The main idea of this method can be summarized as follows:

- Optimize the SVM parameters using a small amount of given data, and get the optimal parameters.
- Train the system using the parameters in step 1 and construct support vectors.
- Forecast the output with the support vectors in step 2.

5.2.2. Simulations Using SMO Algorithm and LS-SVM

Section A

In this part, we forecasted the system output using SMO optimized by PSO with the method proposed above.

Select the first 100 data as training samples for parameters selection.

We can get the SMO parameters as follows through simulation:

C= 1.9105 δ = 0.1572 σ = 0.0010

Then forecast the output.,the training error and the testing error are:

RMSE_train =0.0026

RMSE_test =0.0032

In order to prove the algorithm's effectiveness, the simulation results are compared with the results of SOFNN presented in paper [9].

Table.2 The Results of Different Algorithms in Mackey-Glass Series

Algorithm	RMSE_train	RMSE_test
SOFNN	0.0077193	0.0081944
SMO(optimized by PSO)	0.0026	0.0032
LS-SVM (optimized by its ownfunction)	0.0022	0.0026
LS-SVM (optimized by PSO)	0.0013	0.0021

As can be seen in Table.2, SMO based on PSO optimization is more satisfactory than the algorithm proposed in paper [9] in terms of forecasting capability. By using the method proposed above, the training speed can be accelerated. As a result, the method is feasible in the case with large amount of data.

Section B

In this part, we use LS-SVM to forecast the Mackey-Glass *s*eries in Example 2.

- LS-SVM Identification Using its embedded function

  Select the first 100 data to train the system using LS-SVM optimized by its embedded function, and the parameters optimized by the algorithm are:

  C= 791.733 δ = 1.207752

  Forecast the output using LS-SVM with the first 1000 training

data and the parameters above. The training error and the testing error are:

RMSE_train = 0.0022

RMSE_test = 0.0026

- Identification Using LS-SVM Optimized by PSO Algorithm

  Select the first 100 data to train the system using LS-SVM optimized by PSO, and the parameters generated from the optimization process are:

  C=909.9985 δ =0.6089

  Forecast the output using LS-SVM with the first 1000 training data and the parameters. The training error and the testing error are:

  RMSE_train = 0.0013

  RMSE_test = 0.0021

It can be seen in Table.2 that the results gotten from LS-SVM using the proposed method are adequately fine. Besides, it costs much less time than training with all the 1000 data once off. As a result, the proposed method is feasible.

6. Conclusion

SMO algorithm and LS-SVM has been studied in this paper and a proper solution to the choice of the SVM parameters has been proposed. It is effective and simple to optimize the SVM parameters using the PSO algorithm. The method is verified through the simulation of two examples using both SMO and LS-SVM. In the last part of the paper, an optimization method with a large amount of data is proposed utilizing PSO algorithm and its feasibility has been verified by the simulation results.

7. Acknowledgements

The research is support by Science and Technology Foundation of North China Electric Power University, No.200814003.

8. References

[1] Zhang Xue-gong. Introduction to statistical learning theory and support vector machines[J]. Acta Automatic Sinica, 2000. 26(1):32-42.

[2] J.Platt Sequential minimal optimization:A fast algorithm for training support vector machines[J]. Advances in Kernel Methods-support Vector Learning, 1999.

[3] Yang Jin-fang. Research and applications of SVR in predictive control[D], 2007.North China Electic Power University.

[4] Pelckmans. LS-SVM lab: a MATLAB/C toolbox for Least Squares Support Vector Machines. http://www.esat.kuleuven.ac.be/sista/lssvmlab

[5] Chen Ru-qing, Yu Jin-shou. Soft sensor modeling based on Particle Swarm Optimization and Least Squares Support Vector Machines. Joural of System Simulation. 2007.19(22): 5307-5310

[6] SU Gao-li, Deng Fang-ping. Introduction to Model selection of SVM Regr ession[J]. Bulletin of Science and Technology, 2006.22(2):154-157.

[7] Dong Chun-xi. Method for selecting the parameters of support vector machines[J]. Systems Engineering and Electronics, 2004. 26(8): 1117-1120.

[8] Leng Gang, Prasad,T.M. McGinnity. An on-line algorithm for creating self-organizing fuzzy neural networks[J]. Neural Netwoks, 2004(17):1477-1493.

[9] Zhai Yong-jie. Prediction of chaotic time series using self-organizing fuzzy neural networks with entropy criterion. Journal of North China Electric Power University, 2008. 1(1).