Improving Trained LS–SVM Performance with New Available Data

Yangguang Liu[1] **Bin Xu**[1] **Feng Liu**[1]

[1]Ningbo Institute of Technology, Zhejiang University, Ningbo, 315100, P. R. China

Abstract

Learning is obtaining an underlying rule by using training data sampled from the environment. In many practical situations in inductive learning algorithms, it is often expected to further improve the generalization capability after the learning process has been completed if new data are available. One of the common approaches is to add training data to the learning algorithm and retrain it, but retraining for each new data point or data set can be very expensive. In view of the learning methods of human beings, it seems natural to build posterior learning results upon prior results. Firstly, in this paper, we proposed an updating procedure for least square support vector machine(LS–SVM). If initial concept would be built up by LS–SVM inductive algorithm, then concept updated is the normal solution corresponding to the initial concept learned. Secondly, we discuss a general framework for updating learned concept. Finally, we illustrate the updating method and evaluate it on toy data and real data, their results show that the performance after updating is improved and almost equal to the performance of LS–SVM retrained on whole data.
Keywords: LS-SVM, concept updating, learning

1. Introduction

Learning is obtaining an underlying rule by using training data sampled from the environment. On the one hand in many practical situations in inductive learning algorithms, it is often expected to further improve the generalization capability after the learning process has been completed if new data are available. One of the common approaches is to add training data to the learning algorithm and retrain it, but retraining for each new data point or data set can be very expensive. In view of the learning methods of human beings, it seems natural to build posterior learning results upon prior results. On the other hand, many applications that involve massive data sets are emerging. In order to apply SVM to large scale data problem, many researchers proposed faster implementation of SVM for classification and regression[1, 5, 8].

In this paper, we consider both sides mentioned above. We propose an concept updating method for trained Least Square Support Vector Machine(LS–SVM)[2]. This updating method can be viewed as an incremental updating step, similar to standard LS–SVM in mathematical formula, to update the trained LS–SVM parameters. In the next section, we formulate the standard Least Square Support Vector Machine for Classification. Then we give the updating method and explain in more detail each of its main steps and on some implementation issues. In the experiment section, we present experiments using LS–SVM for classification and concept updating with our method. This results show that prediction accuracy after concept updating increases with updating step.

2. Least Squares SVM

Here we review basic idea of the least squares support vector machines. For detailed information about LS–SVM we refer to [2]

Let us consider the binary classification in the set of linear functions

$$f(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + b \tag{1}$$

Given a training set $\{\mathbf{x}_i, y_i\}_{i=1}^n$ with input patterns $\mathbf{x}_i \in \mathrm{R}^N$ and output values $y_i \in \{-1, +1\}$ indicating the class, SVM formulations[1] start from the assumption that

$$\begin{cases} \mathbf{w}^T\varphi(\mathbf{x}_i) + b \geq +1, & \text{if } y_i = +1; \\ \mathbf{w}^T\varphi(\mathbf{x}_i) + b \leq -1, & \text{if } y_i = -1. \end{cases} \tag{2}$$

which is equivalent to $y_i[\mathbf{w}^T\varphi(\mathbf{x}_i) + b] \geq 1 (i = 1, \cdots, n)$. Here the nonlinear mapping $\varphi(\cdot)$ maps the input data into a so-called higher dimensional feature space. In LS–SVM's[2] an equality constraint based formulation is made within the context of ridge regression as follows

$$\min_{\mathbf{w},\mathbf{e},} \mathcal{J}(\mathbf{w},\mathbf{e}) = \frac{1}{2}\parallel \mathbf{w} \parallel^2 + \gamma\frac{1}{2}\sum_{i=1}^{n} e_i^2 \tag{3}$$

subject to

$$y_i[\mathbf{w}^T\varphi(\mathbf{x}_i)+b] = 1 - e_i, i = 1,\cdots,n \tag{4}$$

with Lagrangian

$$\mathcal{L}(\mathbf{w},b,\mathbf{e};\alpha) = \mathcal{J}(\mathbf{w},\mathbf{e}) - \sum_{i=1}^{n}\alpha_i\{y_i[\mathbf{w}^T\varphi(\mathbf{x}_i)+b]-1+e_i\} \tag{5}$$

and Lagrange multipliers(support values) α_k. The conditions for optimality $\partial\mathcal{L}/\partial\mathbf{w} = 0, \partial\mathcal{L}/\partial b = 0, \partial\mathcal{L}/\partial e_i = 0, \partial\mathcal{L}/\partial\alpha_i = 0$ give $\mathbf{w} = \sum_{i=1}^n \alpha_i y_i \varphi(\mathbf{x}_i), \sum_{i=1}^n \alpha_i y_i = 0, \alpha_i = \gamma e_i, y_i[\mathbf{w}^T\varphi(\mathbf{x}_i) + b] = 1 - e_i (i = 1,\cdots,n)$, respectively. By eliminating $\mathbf{e},\mathbf{w}$ one obtains the KKT system

$$\left[\begin{array}{c|c} 0 & Y^T \\ \hline Y & \Omega+\gamma^{-1}I \end{array}\right]\left[\begin{array}{c} b \\ \hline \alpha \end{array}\right] = \left[\begin{array}{c} 0 \\ \hline \vec{1} \end{array}\right] \tag{6}$$

where $Y = [y_1;\cdots;y_n], \vec{1} = [1;\cdots;1], \alpha = [\alpha_1;\cdots;\alpha_n]$ and

$$\begin{aligned} \Omega_{ij} &= y_i y_j \varphi(\mathbf{x}_i)^T\varphi(\mathbf{x}_j), i,j = 1,\cdots,n \quad (7)\\ &= y_i y_j K(\mathbf{x}_i,\mathbf{x}_j) \end{aligned}$$

after application of the Mercer condition. This finally results into the following LS-SVM classifier

$$y(\mathbf{x}) = \mathtt{sign}\left(\sum_{i=1}^{n}\alpha_i y_i K(\mathbf{x},\mathbf{x}_i) + b\right) \tag{8}$$

where α, b are the solution to (6). For the choice of the kernel function $K(\cdot,\cdot)$ one has several possibilities including the RBF kernel $K(\mathbf{x},\mathbf{x}_i) = \exp\{-||\mathbf{x}-\mathbf{x}_i||_2^2/\sigma^2\}$. Note that σ,γ are to be considered as model parameters for the LS-SVM which must be determined using model selecting method before training LS-SVM.

3. Updating Trained LS-SVM Classifier

In this section, we extend our method in paper[6, 7] to LS-SVM. Recently, the same idea used for personalized handwriting recognition also appeared in paper [9]. First, we propose an updating method for trained LS–SVM, then discuss about the detail updating procedure and define a unified updating risk functional for updating procedure.

3.1. Formulation of Updating Method

To motivate our approach, note that the regularization term $||\mathbf{w}||^2$ in (3) encodes the prior knowledge. In a way, $\mathbf{w} = 0$ is our safest bet in case of unreliable or insufficient data. The idea of this work is to exploit a regularization mechanism by using trained LS-SVM as prior knowledge. To this end, we propose the following updating procedure.

We denote trained LS-SVM with parameters $(\mathbf{w}_0, b_0)$, and at the k-th incremental step with parameters $(\mathbf{w}_k, b_k)$. We propose solving the following programming problem as the incremental step

$$\min_{\mathbf{w}_k,\mathbf{e}} \mathcal{U}_k(\mathbf{w}_k,\mathbf{e}) = \frac{1}{2}\parallel \mathbf{w}_k - \mathbf{w}_0 \parallel^2 + \gamma\frac{1}{2}\sum_{i=1}^{\ell_k} e_i^2 \tag{9}$$

subject to

$$y_i[\mathbf{w}_k^T\varphi(\mathbf{x}_i)+b_k] = 1 - e_i, i = 1,\cdots,\ell_k \tag{10}$$

One defines the Lagrangian

$$\begin{aligned} \mathcal{L}_2(\mathbf{w}_k,b_k,\mathbf{e},\alpha) &= \mathcal{U}_k(\mathbf{w}_k,\mathbf{e}) \quad (11)\\ &\quad -\sum_{i=1}^{\ell_k}\alpha_i\{y_i[\mathbf{w}_k^T\varphi(\mathbf{x}_k)+b_k] \\ &\quad -1+e_i\} \end{aligned}$$

where α_i are Lagrange multipliers.

The conditions for optimality

$$\begin{aligned} \frac{\partial\mathcal{L}_2}{\partial\mathbf{w}_k} = 0 &\rightarrow \mathbf{w}_k = \mathbf{w}_0 + \sum_{i=1}^{\ell_k}\alpha_i y_i\varphi(\mathbf{x}_i) && (12)\\ \frac{\partial\mathcal{L}_2}{\partial b_k} = 0 &\rightarrow \sum_{i=1}^{\ell_k}\alpha_i y_i = 0 && (13)\\ \frac{\partial\mathcal{L}_2}{\partial e_i} = 0 &\rightarrow \alpha_i = \gamma e_i, i = 1,\cdots,\ell_k && (14)\\ \frac{\partial\mathcal{L}_2}{\partial\alpha_i} = 0 &\rightarrow y_i[\mathbf{w}_k^T\varphi(\mathbf{x}_i)+b_k] - 1 + e_i = 0, \\ &\quad i = 1,\cdots,\ell_k && (15) \end{aligned}$$

can be written immediately as the solution to the following set of linear equations

$$\left[\begin{array}{ccc|c} I & 0 & 0 & Y^T \\ 0 & 0 & 0 & -Y^T \\ 0 & 0 & \gamma I & -I \\ \hline Z & Y & I & 0 \end{array}\right]\left[\begin{array}{c} \Delta\mathbf{w} \\ \Delta b \\ \mathbf{e} \\ \hline \alpha \end{array}\right] = \left[\begin{array}{c} 0 \\ 0 \\ 0 \\ \hline \vec{1}-\vec{f} \end{array}\right] \tag{16}$$

where $Z = [y_1\varphi(\mathbf{x}_1)^T,\cdots,y_{\ell_k}\varphi(\mathbf{x}_{\ell_k})^T]^T, Y = [y_1,\cdots,y_{\ell_k}]^T, \Delta\mathbf{w} = \mathbf{w}_k - \mathbf{w}_0, \Delta b = b_k -$

$b_0, \vec{1} = [1, \cdots, 1]^T, \mathbf{e} = [e_1, \cdots, e_{\ell_k}]^T, \vec{f} = [y_1 f(\mathbf{x}_1), \cdots, y_k f(\mathbf{x}_{\ell_k})]^T$. The solution is also given by

$$\left[\begin{array}{c|c} 0 & -Y^T \\ \hline Y & ZZ^T + \gamma^{-1} I \end{array}\right] \left[\begin{array}{c} \Delta b \\ \hline \alpha \end{array}\right] = \left[\begin{array}{c} 0 \\ \hline \vec{1} - \vec{f} \end{array}\right]. \quad (17)$$

The kernel trick can be applied again to the matrix $\Omega = ZZ^T$ where

$$\Omega_{ij} = y_i y_j \varphi(\mathbf{x}_i)^T \varphi(\mathbf{x}_j) = y_i y_j K(\mathbf{x}_i, \mathbf{x}_j). \quad (18)$$

after application of the kernel trick. This finally results into the following updated LS-SVM classifier

$$\begin{aligned} y_k(\mathbf{x}) &= \mathtt{sign}\Big((\mathbf{w}_0 + \Delta\mathbf{w})^T \varphi(\mathbf{x}) + b_0 + \Delta b \Big) \quad (19) \\ &= \mathtt{sign}\Big(\big(\sum_{i=1}^{\ell_0} \alpha_i^{(0)} y_i K(\mathbf{x}, \mathbf{x}_i^{(0)}) + b_0 \big) \\ &\quad + \big(\sum_{j=1}^{\ell_k} \alpha_j^{(k)} y_j K(\mathbf{x}, \mathbf{x}_i^{(k)}) + \Delta b \big) \Big) \quad (20) \end{aligned}$$

Note that the equation (6) and (17) are only different on the righthand, so we can solve the updating problem with LS–SVM solvers.

3.2. Updating Procedure

In fact, we, proposed a method to update the trained support vector machine from above discussing. The support vector updating procedure can be described as following:

1. Learning initial concept: training LS-SVM with initial data, and then attain the initial regression function determined by $\mathbf{w}_0, b_0$
2. Organizing new data: if our method is used to solve large scale data regression, large data set can be partitioned into small data set. There exist many techniques to partition the large data set, such as random sampling, stratified sampling and sequential multi-sample learning etc(see[3] for a review). And if new coming data are available, we also have many strategies to organize the new data, such as Error-driven, exceeding-margin, fixed-partition[4] etc.
3. updating last learned concept: solve updating problem with organized data, then attain the updated classification function determined by $(\mathbf{w}_k, b_k)$. Return last step if new data available.

3.3. The General Formulation

It can be observed that proposed method has in risk functional (21). A unified updating risk functional is defined as following

$$H[\Delta f] = \lambda \parallel \Delta f \parallel_K^2 + \frac{1}{\ell} \sum_{i=1}^{\ell} V(y_i, f_k(\mathbf{x}_i)) \quad (21)$$

where $V(\cdot, \cdot)$ is *lost function.* The updating method of Support Vector Machine for classification and regression[6, 7] and LS-SVM correspond to the minimization of H in equation(21) for different choices for V:

- Least Square Support Vector Machine(LS–SVM)

$$V(y_i, f_k(\mathbf{x}_i)) = (y_i - f_k(\mathbf{x}_i))^2 \quad (22)$$

- Support Vector Machines Regression(SVMR)

$$V(y_i, f_k(\mathbf{x}_i)) = |y_i - f_k(\mathbf{x}_i)|_\epsilon \quad (23)$$

- Support Vector Machines Classification(SVMC)

$$V(y_i, f_k(\mathbf{x}_i)) = |y_i - f_k(\mathbf{x}_i)|_+ \quad (24)$$

where $|\cdot|_\epsilon$ is Vapnik's epsilon–insensitive norm(see [1]), $|x|_+ = x$ if x is positive and zero otherwise, and y_i is a real number in SVMR, whereas it takes values -1, 1 in LS–SVM and SVMC.

4. Experimental results

In this section, we describe and discuss experiments we have performed with synthetic toy data and natural data from the UCI repository(available at http://www.ics.uci.edu/∼mlearn/ MLRepository.html): banana, breast cancer, diabetes, german, heart, flare-solar(see Table 1 for detail). Some of the problems are originally not binary classification problems, hence a random partition into two classes is used.

The initial concept was learned by LS–SVM matlab tools, and the updating problem is solved base on the modified LS–SVM matlab tool. All our experiments are done on a Celoren 1.7Ghz machine with 256MB memory running on WindowsXP.

We generate 10 partitions into training and test set. On each partition we train a classifier and then compute its test set error. In our experiments, we chose the RBF kernel function with parameter σ(see Table 4). We used one-third of train data to learn initial concept and the other for updating.

Data set	#Trn	#Tst	#Attr.	Para. γ	Para. σ
Banana	400	4900	2	316.2	1.0
Cancer	200	77	9	15.2	50.0
Diabetes	468	300	8	10.0	20.0
Flare	666	400	9	10.2	3.0
German	700	300	20	3.2	55.0
Heart	170	100	13	3.2	120.0

Table 1: Dataset and Experimental Parameters

Data	Init. LS–SVM			uLS–SVM		
TS%	1/3	1/2	2/3	2/3	1/2	1/3
Banana	85.89	87.07	88.05	87.49	88.47	**88.61**
Cancer	69.48	69.74	72.43	70.13	70.74	**72.68**
Diabetes	72.90	73.83	74.08	73.10	74.83	**74.93**
Flare	64.03	64.13	66.30	64.28	**65.23**	64.28
German	73.97	75.27	75.56	75.32	**75.76**	75.62
Heart	79.80	80.90	82.84	80.30	82.90	**83.55**

Table 2: Average Performance over 10 partitions of the data set, the second row indicate the percentage of data for initial training and updating

4.1. Experimental Results

The test performance over 10 partitions of the banana data sets is given in Fig.1. The dotted line describes the initial performance on independent test set. The solid line line describes the performance after updating. The dash-dotted line describes the LS–SVM trained on all data at once. The initial concept learned by LS–SVM on banana data set is shown in Fig. 2, updated concept and full concept trained on all data by LS–SVM are shown in Fig. 3. The average generalization performance over 10 partitions of the data sets is given in the Table 4.

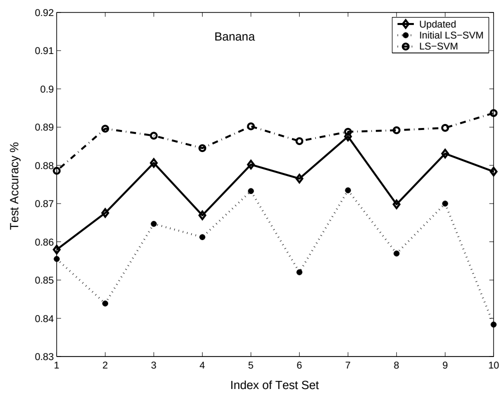

Fig. 1: Test Accuracy over 10 partitions of the banana data sets

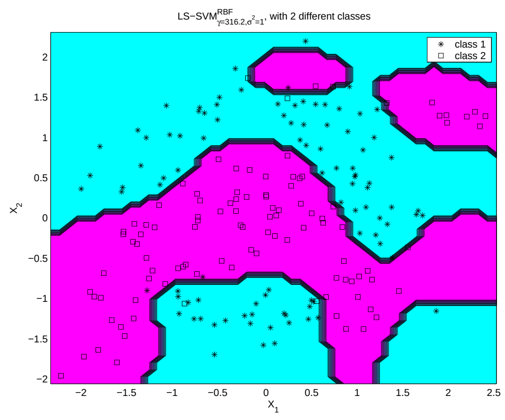

Fig. 2: Initial classification boundary trained on half data

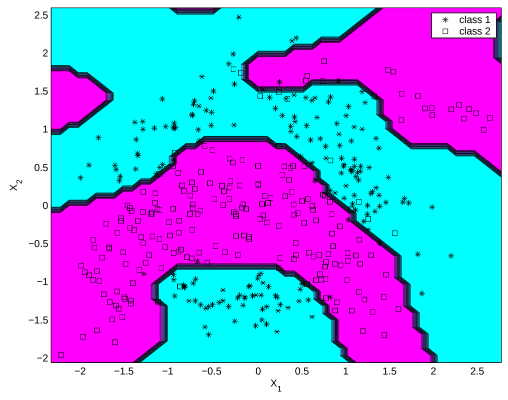

Fig. 3: Updated classification boundary using remained half data

The second column in Table 2 showing "Init. LS–SVM", gives the average initial performance, the third column give the average performance of updating method and performance of LS–SVM and standard SVM trained whole data at once in Table 3, respectively. Our experiments on 6 data(cf. Table 4) show that the results of updating procedure improve the performance of initial concept and almost equal to the performance of LS–SVM retrained on all data at once(cf. Table 3).

5. Conclusions

In this paper, we proposed an updating procedure for least square support vector machine(LS–SVM). If initial concept would be built up by LS–SVM inductive algorithm, then concept updated is the normal solution corresponding to the initial concept learned. Then, we discuss a general framework for updating learned concept. Finally, we illustrate the

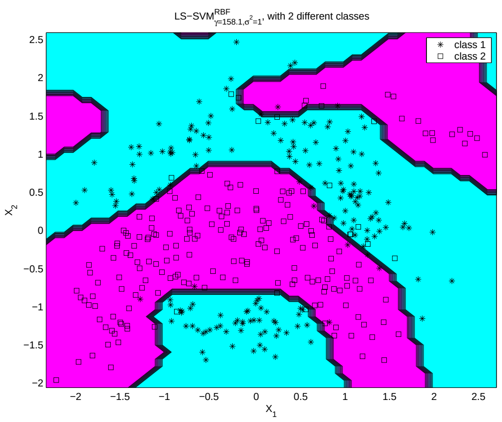

Fig. 4: LS-SVM classification boundary trained on whole data

Data Set	LS-SVM	SVM
Banana	**88.78**	88.47
Cancer	*70.39*	**73.96**
Diabetes	74.93	**76.43**
Flare	64.28	**67.57**
German	**77.20**	76.39
Heart	83.70	**84.05**

Table 3: Average Performance of LS-SVM and SVM trained on whole data

updating method and evaluate it on toy data and real data, their results show that the performance after updating is improved and almost equal to the performance of LS–SVM retrained on all data at once. Next further works concern tests on large scale learning tasks like text categorization.

ACKNOWLEDGMENTS

This work was supported in part by a grant(2005A610002) from the Ningbo Doctor Science Fund and the grand 1141157G601 from Ningbo Institute of Technology, Zhejiang University in China.

References

[1] V. Vapnik, *The Nature of Statistical Learning Theory*, Springer-Verlag, 1995.

[2] J.A.K. Suykens and J. Vandewlle, Least squares support vector machine classifiers, *Neural Processing Letters*, 9(3):293–300, Springer, 1999.

[3] F. Provost and V. Kolluri, A survey of methods for scaling up inductive algorithms, *Data Mining and Knowledge Discovery*, 3(2):131–169, Kluwer Academic Publishers, 1999.

[4] C. Domeniconi and D. Gunopulos, Incremental support vector machine construction. In N. Cercone, T.Y. Lin, X. Wu, editors, proceedings of the 2001 *IEEE International Conference on Data Mining*, IEEE Computer Society, November 29 – December 2, pages 589–592, California(USA), 2001.

[5] M. Martin, On-line support vector machines for function approximation. Technical Report, Department of Software, Universitat Politecnica de Catalunya, LSI-02-11-R, Catalunya, 2002.

[6] Y.G.Liu, Q. Chen, Y.C Tang and Q.M. He, An incremental updating method for support vector machines. In J. X. Yu, X.M Lin, H.J. Lu and Y.C Zhang, editors, proceedings of the *Advanced Web Technologies and Applications, the 6th Asia-Pacific Web Conference* (APWeb 2004), Lecture Notes in Computer Scienece 3007, pages 426–435, Springer-Verlag, 2004.

[7] Y.G. Liu and Q.M. He, Concept updating with support vector mahines. In W.F. Fan, Z.H. Wu, J. Yang, editors, proceedings of the 6th *International Conference on Advances in Web-Age Information Management* (WAIM 2005), Lecture Notes in Computer Science 3739, pages 492–501, Springer-Verlag, 2005.

[8] R.E. Fan, P.H. Chen and C.J. Lin, Working set selection using the second order information for training SVM, *Journal of Machine Learning Research*, 6:1889–1918, Springer, 2005.

[9] W. Kienzle, K. Chellapilla, Personalized handwriting recognition via biased regularization, In W. Cohen and A. Moore, editors, *proceedings of the 23rd International Conference on Machine Learning* (ICML 2006), ACM, pages 457–464, June 25-29, Pittsburgh(USA), 2006.