Incremental Learning Algorithm of Least Square Twin KSVC

Wang Yaru , Yang Ling , Chen Mohan , Xi Jikui

School of Information Science and Engineering, Lanzhou University, Lanzhou 730000, China

Abstract. In view of the batch implementations of standard support vector machine must be retrained from scratch every time when the training set is incremental modified, an incremental learning algorithm based on least squares twin multi-class classification support vector machine (ILST-KSVC) is proposed by solving two inverse matrix. The method will be applied on online environment to update initial data, which avoided cumbersome double counting. ILST-KSVC inherited the advantages of the basic algorithm and has some merits of Least square twin support vector machine for excellent performance on training speed and support vector classification regression for K-class's well classification accuracy. The result will be confirmed no matter in low dimension or in high dimension in UCI datasets.

1 Introduction

Support vector machine is a type of machine learning algorithm, which aims at finding one optimal hyperplane to make two kinds of training data as far as possible away from the hyperplane in the high dimensional space[1].It has great advantages of solving small sample, nonlinear and high dimensional pattern recognition problem, therefore, its been successfully applied to many aspects, such as, medical image detection[2], network information classification[3,4], pattern recognition[5], etc. Classical SVM needs to solving a Quadratic Programming Problem (QPP), which computational complexity is O(l3)(l refers to the total number of training data), so when dealing with large-scale data is restricted. Twin support vector machines (TSVM) been proposed by Jayadeva, etc in 2007. It divides one QPP into two smaller QPPs from SVM. By solving two smaller QPPs to construct two non-parallel hyperplanes, and makes each hyperplane as far as possible close to the kind of sample data and away from the other. TSVM simplifies the computational complexity of SVM and theory has been shown that its computational complexity is O(l3/4), about a quarter of the SVM[6]. Least squares twin support vector machine (LS-TSVM)[7] greatly simplify the computational complexity of TSVM. It constructs a squared loss function modifying with convex linear system to replace the TSVM convex QPPs and substitutes equality constraints for inequality constraints. Thus, solving two equations improves the solving efficiency and accuracy of classification.

Classical SVM and TSVM are designed for binary classification problems; however, we often encounter many classification problems in real life. We usually use '1-versus-1'[8] and '1-versus-1'[9] to dealing with muti-class classification problems in these methods. For K-class, the first method is to construct K two classifiers, each class corresponding to one of them been separated from other classes, the method is faster but easy to cause the imbalance of classifier. The second method construct all the possible binary classifiers in K class of training samples , so need to construct K(K-1)/2 binary classifiers. Each class training data only taken from the corresponding two classes, without considering other sample information, which leading to classification's result is not ideal.2003, Angulo proposed the '1-versus-1-versus-rest 'multi-class classification algorithm---support vector classification regression for K-class classification(K-SVCR)[10]. It constructs K(K-1)/2 K-SVCR triple classifier to deal with K class classification problem, this structure improve accuracy of muti-class problems, but the computational complexity is high. Twin multi-class classification support vector machine (Twin-KSVC) has been come up with, based on TSVM and K-SVCR [11]. It takes advantages of TSVM and SVCR that running time is close to TSVM but far less than K-SVCR. In order to reduce the computational complexity and simplified the structure of classifier further, Jalal a. etc. combined with the LS-TSVM and K-SVCR put forward Least squares twin multi-class classification support vector machine (LST-KSVC)[12].LST-KSVCR inherited the faster training speed of LS-TSVM and high accuracy of KSVC.

In many online scenario, the new data will be continuously added to the data set, while the standard SVM, TSVM will no longer be applicable to the classification of the real-time data. In order to solve the classification problem of the real-time data, many incremental learning algorithm based on SVM and TSVM [13]-[16] has been proposed. In this paper, we propose an incremental learning algorithm suitable for an online environment called Incremental Learning

Algorithm for LST-KSVC; we also call its ILST-KSVC, briefly. The Algorithm introduces SMW (Sherman - Morrison - Woodbury) formula to solve the inverse of the matrix, and every time a new sample data increases, the only need to update inverse matrix on the basis of original weight matrix, without calculating weight matrix inverse matrix, the process will not have an impact on the result of the classification. Decremental is the inverse process of incremental, therefore, the algorithm also applies to the decremental process.

2 Least square twin K-SVCR

2.1 1-versus-1-versus-rest

'one-versus-one-versus-rest' method is a new breakthrough to solve the multiple classification problems. It combines the standard support vector classifier and support vector regression machine together to solve the triple class directly.

Set training set $Q=\{(x_p,y_p)\mid p=1,2,..k\}$

Here, $y_p=\begin{cases}1 & p=1,\ldots,l_1\\ -1 & p=l_1+1,\ldots,l_1+l_2\\ 0 & p=l_1+1,\ldots,l_1+l_2,\ldots,l\end{cases}$

Let matrix $\boldsymbol{A}\subset\boldsymbol{R}^{l_1\times n}$ indicates left class training data labeled '1', matrix $\boldsymbol{B}\subset\boldsymbol{R}^{l_2\times n}$ indicates right class training data labeled '-1' and matrix $\boldsymbol{C}\subset\boldsymbol{R}^{l_3\times n}$ is the other class training data labeled '0'. l is the total training data and $l=l_1+l_2+l_3$.

For K-class classification, the method chooses each i, $j(i\neq j)$ as left class and right class, the rest K-2 classes as class 0, called 'one-versus-one-versus-rest'. Training each i, j class to get K (K - 1) / 2 ternary classifier, corresponding to get K (K - 1) / 2 the output of the decision function. In order to determine the category of the test sample, using the voting as follows: when the decision function to output 1,cast class i one vote; When the decision function output of -1, cast class j one vote; When the decision function output is 0, do not vote. Inspection after all the K (K - 1) / 2 the output of the decision function, put the test sample into the category with the most votes

2.2 linear LST-KSVC

Twin-KSVC needs to address two QPPs, which costs high computational. And LST-KSVCR reduced computation cost by solving two reversible matrix, make it can be used for processing data of high dimension matrices. LST-KSVC change Twin-KSVC's inequality constraints to the linear equality constraint conditions, thus, the original optimization problem as follows.

$$\begin{aligned}&\min_{w_1,b_1}\frac{1}{2}\left\|\boldsymbol{A}w_1+e_1b_1\right\|^2+\frac{c_1}{2}y^{\mathrm{T}}y+\frac{c_2}{2}z^{\mathrm{T}}z\\ &\text{s.t.}\ -(\boldsymbol{B}w_1+e_2b_1)+y=e_2\\ &\quad -(\boldsymbol{C}w_1+e_3b_1)+z=e_3(1-\varepsilon)\end{aligned}\tag{1}$$

$$\begin{aligned}&\min_{w_2,b_2}\frac{1}{2}\left\|\boldsymbol{B}w_2+e_2b_2\right\|^2+\frac{c_3}{2}y^{\mathrm{T}}y+\frac{c_4}{2}z^{\mathrm{T}}z\\ &\text{s.t.}\ (\boldsymbol{A}w_2+e_1b_2)+y=e_1\\ &\quad (\boldsymbol{C}w_2+e_3b_2)+z=e_3(1-\varepsilon)\end{aligned}\tag{2}$$

Where, c_1, c_2, c_3, c_4 are penalty parameter, $y>0$, $z>0$ is slack variable, ε is between 0 and 1. w, b is weights and bias for solving. Substituting the equality constraint into the objective function in formula (1), we obtain

$$\min\frac{1}{2}\left\|\boldsymbol{A}w_1+e_1b_1\right\|^2+\frac{c_1}{2}\left\|\boldsymbol{B}w_1+e_2b_1+e_2\right\|^2+\frac{c_2}{2}\left\|\boldsymbol{C}w_1+e_3b_1+e_3(1-\varepsilon)\right\|^2\tag{3}$$

Setting the gradient of (3) with respect to w_1, b_1 to zeros gives

$$\begin{aligned}\begin{bmatrix}w_1\\ b_1\end{bmatrix}=&-[c_1\begin{bmatrix}\boldsymbol{B}^{\mathrm{T}}\\ e_2^{\mathrm{T}}\end{bmatrix}[\boldsymbol{B}\quad e_2]+\begin{bmatrix}\boldsymbol{A}^{\mathrm{T}}\\ e_1^{\mathrm{T}}\end{bmatrix}[\boldsymbol{A}\quad e_1]+c_2\begin{bmatrix}\boldsymbol{C}^{\mathrm{T}}\\ e_3^{\mathrm{T}}\end{bmatrix}\\ &\times[\boldsymbol{C}\quad e_3]]^{-1}\times[c_1\begin{bmatrix}\boldsymbol{B}^{\mathrm{T}}\\ e_2^{\mathrm{T}}\end{bmatrix}e_2+c_2\begin{bmatrix}\boldsymbol{C}^{\mathrm{T}}\\ e_3^{\mathrm{T}}\end{bmatrix}e_3(1-\varepsilon)]\end{aligned}\tag{4}$$

In brief, let $\boldsymbol{E}=[\boldsymbol{A}\ \ e_1]$, $\boldsymbol{F}=[\boldsymbol{B}\ \ e_2]$, $\boldsymbol{G}=[\boldsymbol{C}\ \ e_3]$, then,

$$[w_1,b_1]^T=-(c_1F^{\mathrm{T}}F+E^{\mathrm{T}}E+c_2G^{\mathrm{T}}G)^{-1}(c_1F^{\mathrm{T}}e_2+c_2G^{\mathrm{T}}e_3(1-\varepsilon))\tag{5}$$

In the same way, we get

$$[w_2,b_2]^T=(c_3E^{\mathrm{T}}E+F^{\mathrm{T}}F+c_4G^{\mathrm{T}}G)^{-1}(c_3E^{\mathrm{T}}e_1+c_4G^{\mathrm{T}}e_3(1-\varepsilon))\tag{6}$$

By formula (5) and formula (6), we can obtain two linear separable non parallel hyperplanes.

$$xw_1^{\mathrm{T}}+b_1=0\ \text{and}\ xw_2^{\mathrm{T}}+b_2=0\tag{7}$$

LST-KSVC use 'one-versus-one-versus-rest' method to training dataset. Linear LST-KSVC use the following decision function in formula (8) to classify the class with output {1, -1, 0}.

$$f(x_i)=\begin{cases}+1 & if\ x_iw_1+eb_1>-1+\varepsilon\\ -1 & if\ x_iw_1+eb_1<1-\varepsilon\\ 0 & otherwise\end{cases}\tag{8}$$

For K class classification problems, choose section 2.1 of the classified voting method.

2.3 non-linear LST-KSVC

By introducing a gaussian kernel function, linear LST - K SVC can be extended to non-linear. The original optimiza tion problem of non-linear LST-KSVC as follows.

$$\begin{aligned}&\min_{u_1,\gamma_1}\ \frac{1}{2}||K(\boldsymbol{A},\boldsymbol{D}^{\mathrm{T}})u_1+e_1\gamma_1||^2+\frac{c_1}{2}y^{\mathrm{T}}y+\frac{c_2}{2}z^{\mathrm{T}}z\\ &\text{s.t.}\ \ -(K(\boldsymbol{B},\boldsymbol{D}^{\mathrm{T}})u_1+e_2\gamma_1)+y=e_2\\ &\text{s.t.}\ \ -(K(\boldsymbol{C},\boldsymbol{D}^{\mathrm{T}})u_1+e_3\gamma_1)+z=e_3(1-\varepsilon)\end{aligned}\tag{9}$$

$$\begin{aligned}&\min_{u_2,\gamma_2}\ \frac{1}{2}||K(\boldsymbol{B},\boldsymbol{D}^{\mathrm{T}})u_2+e_2\gamma_2||^2+\frac{c_3}{2}y^{\mathrm{T}}y+\frac{c_4}{2}z^{\mathrm{T}}z\\ &\text{s.t.}\ \ (K(\boldsymbol{A},\boldsymbol{D}^{\mathrm{T}})u_2+e_1\gamma_2)+y=e_1\\ &\text{s.t.}\ \ (K(\boldsymbol{C},\boldsymbol{D}^{\mathrm{T}})u_2+e_3\gamma_2)+z=e_3(1-\varepsilon)\end{aligned}\tag{10}$$

Here, $K(\cdot)$ is arbitrary kernel function. $\boldsymbol{D}=[\boldsymbol{A};\ \boldsymbol{B};\ \boldsymbol{C}]$, u,γ is weights and bias for solving. Similarly in the solution process of linear LST-KSVC, the solution of formula (9) and formula (10) are as follows.

$$[u_1,\gamma_1]^T=-(c_1\boldsymbol{N}^{\mathrm{T}}\boldsymbol{N}+\boldsymbol{M}^{\mathrm{T}}\boldsymbol{M}+c_2\boldsymbol{O}^{\mathrm{T}}\boldsymbol{O})^{-1}(c_1\boldsymbol{N}^{\mathrm{T}}e_2+c_2\boldsymbol{O}^{\mathrm{T}}e_3(1-\varepsilon))\tag{11}$$

$$[u_2,\gamma_2]^T=(c_3\boldsymbol{M}^{\mathrm{T}}\boldsymbol{M}+\boldsymbol{N}^{\mathrm{T}}\boldsymbol{N}+c_4\boldsymbol{O}^{\mathrm{T}}\boldsymbol{O})^{-1}(c_3\boldsymbol{M}^{\mathrm{T}}e_1+c_4\boldsymbol{O}^{\mathrm{T}}e_3(1-\varepsilon))\tag{12}$$

Here, $\boldsymbol{M}$=[K($\boldsymbol{A}$, $\boldsymbol{D}^{\mathrm{T}}$) e_1], $\boldsymbol{N}$=[K($\boldsymbol{B}$, $\boldsymbol{D}^{\mathrm{T}}$) e_2], $\boldsymbol{O}$=[K($\boldsymbol{C}$, $\boldsymbol{D}^{\mathrm{T}}$) e_3].

By formula (11) and formula (12), we can obtain two linear separable non parallel hyperplanes.

$$K(x^{\mathrm{T}}, \boldsymbol{D}^{\mathrm{T}})u_1 + \gamma_1 = 0 \quad \text{and} \quad K(x^{\mathrm{T}}, \boldsymbol{D}^{\mathrm{T}})u_2 + \gamma_2 = 0 \quad (13)$$

Non-linear LST-KSVC use this decision function in formula (14).

$$f(x_i) = \begin{cases} +1 & \text{if } K(x_i, \boldsymbol{D}^{\mathrm{T}})u_1 + e\gamma_1 > -1+\varepsilon \\ -1 & \text{if } K(x_i, \boldsymbol{D}^{\mathrm{T}})u_2 + e\gamma_2 < 1-\varepsilon \\ 0 & otherwise \end{cases} \quad (14)$$

For K class classification problems, also choose section 2.1 of the classified voting method.

3 Incremental learning algorithm of LST-KSVC (ILST-KSVC)

In this section, we extend the LS-TSVCR to incremental learning and decremental learning version from linear to non-linear as well as LS-TSVCR. First, we need to simplify some symbols for convenience. Let w_1 replace $[w_1, b_1]^{\mathrm{T}}$ in formula (5), w_2 replace $[w_2, b_2]^{\mathrm{T}}$ in formula (6), so $\hat{w}_1$ is the update of w_1, and $\hat{w}_2$ is the update of w_2.

3.1 linear ILST-KSVC

In this paper, we adopt a kind of split matrix inverse, and reorganization of the inverse of the matrix. Finally obtain incremental learning algorithm which has a simple structure and small amount of calculation.

Adding new positive training data $\boldsymbol{S}_i$=[x_i^{T}, 1]$^{\mathrm{T}}$, for example. Then the matrix $\boldsymbol{E}$ can be extended to $\widetilde{\boldsymbol{E}}$.

$$\widetilde{\boldsymbol{E}}^{\mathrm{T}}\widetilde{\boldsymbol{E}} = \begin{bmatrix} \boldsymbol{E} \\ \boldsymbol{S} \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} \boldsymbol{E} \\ \boldsymbol{S} \end{bmatrix} = [\boldsymbol{E}^{\mathrm{T}} \quad \boldsymbol{S}^{\mathrm{T}}] \begin{bmatrix} \boldsymbol{E} \\ \boldsymbol{S} \end{bmatrix} = \boldsymbol{E}^{\mathrm{T}}\boldsymbol{E} + \boldsymbol{S}^{\mathrm{T}}\boldsymbol{S} \quad (15)$$

According to formula (5) and formula (6), we get

$$\begin{aligned} \hat{\boldsymbol{w}}_1 &= -(c_1\boldsymbol{F}^{\mathrm{T}}\boldsymbol{F} + \boldsymbol{E}^{\mathrm{T}}\boldsymbol{E} + c_2\boldsymbol{G}^{\mathrm{T}}\boldsymbol{G} + \boldsymbol{S}^{\mathrm{T}}\boldsymbol{S})^{-1} \\ &\quad \times (c_1\boldsymbol{F}^{\mathrm{T}}e_2 + c_2\boldsymbol{G}^{\mathrm{T}}e_3(1-\varepsilon)) \end{aligned} \quad (16)$$

$$\begin{aligned} \hat{\boldsymbol{w}}_2 &= (c_3\boldsymbol{E}^{\mathrm{T}}\boldsymbol{E} + \boldsymbol{F}^{\mathrm{T}}\boldsymbol{F} + c_4\boldsymbol{G}^{\mathrm{T}}\boldsymbol{G} + \boldsymbol{S}^{\mathrm{T}}\boldsymbol{S})^{-1} \\ &\quad \times (c_3\boldsymbol{E}^{\mathrm{T}}e_1 + c_4\boldsymbol{G}^{\mathrm{T}}e_3(1-\varepsilon)) \end{aligned} \quad (17)$$

For solving formula (16) and formula (17), we need to solve two inverse matrices by introducing the SMW theorem.

Let

$$\boldsymbol{H}_1 = c_1\boldsymbol{F}^{\mathrm{T}}\boldsymbol{F} + \boldsymbol{E}^{\mathrm{T}}\boldsymbol{E} + c_2\boldsymbol{G}^{\mathrm{T}}\boldsymbol{G} \quad (18)$$

$$\boldsymbol{H}_2 = c_3\boldsymbol{E}^{\mathrm{T}}\boldsymbol{E} + \boldsymbol{F}^{\mathrm{T}}\boldsymbol{F} + c_4\boldsymbol{G}^{\mathrm{T}}\boldsymbol{G} \quad (19)$$

Thus,

$$\begin{aligned} \hat{\boldsymbol{H}}_1^{-1} &= (\boldsymbol{H}_1 + \boldsymbol{S}^{\mathrm{T}}\boldsymbol{S})^{-1} = \boldsymbol{H}_1^{-1} - \frac{\boldsymbol{H}_1^{-1}\boldsymbol{S}^{\mathrm{T}}}{1+\boldsymbol{S}\boldsymbol{H}_1^{-1}\boldsymbol{S}^{\mathrm{T}}}\boldsymbol{S}\boldsymbol{H}_1^{-1} \\ &= \boldsymbol{H}_1^{-1} - \frac{\alpha_1}{1+\boldsymbol{S}\alpha_1}\boldsymbol{S}\boldsymbol{H}_1^{-1} = \boldsymbol{H}_1^{-1} - \beta_1\boldsymbol{S}\boldsymbol{H}_1^{-1} \\ &= (\boldsymbol{I} - \beta_1\boldsymbol{S})\boldsymbol{H}_1^{-1} = \eta_1\boldsymbol{H}_1^{-1} \end{aligned} \quad (20)$$

$$\begin{aligned} \hat{\boldsymbol{H}}_2^{-1} &= (\boldsymbol{H}_2 + \boldsymbol{S}^{\mathrm{T}}\boldsymbol{S})^{-1} = \boldsymbol{H}_2^{-1} - \frac{\boldsymbol{H}_2^{-1}\boldsymbol{S}^{\mathrm{T}}}{1+\boldsymbol{S}\boldsymbol{H}_2^{-1}\boldsymbol{S}^{\mathrm{T}}}\boldsymbol{S}\boldsymbol{H}_2^{-1} \\ &= \boldsymbol{H}_2^{-1} - \frac{\alpha_2}{1+\boldsymbol{S}\alpha_2}\boldsymbol{S}\boldsymbol{H}_2^{-1} = \boldsymbol{H}_2^{-1} - \beta_2\boldsymbol{S}\boldsymbol{H}_2^{-1} \\ &= (\boldsymbol{I} - \beta_2\boldsymbol{S})\boldsymbol{H}_2^{-1} = \eta_2\boldsymbol{H}_2^{-1} \end{aligned} \quad (21)$$

Where, $\alpha_1 = \boldsymbol{H}_1^{-1}\boldsymbol{S}^{\mathrm{T}}$, $\alpha_2 = \boldsymbol{H}_2^{-1}\boldsymbol{S}^{\mathrm{T}}$, $\beta_i = \frac{\alpha_i}{1+\boldsymbol{S}\alpha_i}, (i=1,2)$, $\eta_i = \boldsymbol{I} - \beta_i\boldsymbol{S}, (i=1,2)$.

$\boldsymbol{H}_1^{-1}$ and $\boldsymbol{H}_2^{-1}$ can be gained by solving inverse matrixs. Therefor, we only need to calculate η_1 and η_2 to get the after updating matrixs $\hat{\boldsymbol{H}}_1^{-1}$ and $\hat{\boldsymbol{H}}_2^{-1}$ of $\boldsymbol{H}_1^{-1}$and $\boldsymbol{H}_2^{-1}$. Finally, we get the updating $\hat{\boldsymbol{w}}_1$ and $\hat{\boldsymbol{w}}_2$.

$$\hat{\boldsymbol{w}}_1 = -\hat{\boldsymbol{H}}_1^{-1}(c_1\boldsymbol{F}^{\mathrm{T}}e_2 + c_2\boldsymbol{G}^{\mathrm{T}}e_3(1-\varepsilon)) \quad (22)$$

$$\hat{\boldsymbol{w}}_2 = \hat{\boldsymbol{H}}_2^{-1}(c_3\boldsymbol{E}^{\mathrm{T}}e_1 + c_4\boldsymbol{G}^{\mathrm{T}}e_3(1-\varepsilon)) \quad (23)$$

3.2 Non-linear ILST-KSVC

Getting Nonlinear LST - KSVC incremental algorithm we also add a right class training sample, for instance. Mapping training sample to high-dimensional $\boldsymbol{S}_i' = [K(x_i, \boldsymbol{D}^{\mathrm{T}}) \quad e]$.

According to formula (11) and formula (12), in nonlinear LST-KSVC, we can easily get the new weight and bias after updating, like formula (24) -(25).

$$\hat{\boldsymbol{u}}_1 = -(c_1\boldsymbol{N}^{\mathrm{T}}\boldsymbol{N} + \boldsymbol{M}^{\mathrm{T}}\boldsymbol{M} + c_2\boldsymbol{O}^{\mathrm{T}}\boldsymbol{O} + \boldsymbol{S}'^{\mathrm{T}}\boldsymbol{S}')^{-1}(c_1\boldsymbol{N}^{\mathrm{T}}e_2 + c_2\boldsymbol{O}^{\mathrm{T}}e_3(1-\varepsilon)) \quad (24)$$

$$\hat{\boldsymbol{u}}_2 = (c_3\boldsymbol{M}^{\mathrm{T}}\boldsymbol{M} + \boldsymbol{N}^{\mathrm{T}}\boldsymbol{N} + c_4\boldsymbol{O}^{\mathrm{T}}\boldsymbol{O} + \boldsymbol{S}'^{\mathrm{T}}\boldsymbol{S}')^{-1}(c_3\boldsymbol{M}^{\mathrm{T}}e_1 + c_4\boldsymbol{O}^{\mathrm{T}}e_3(1-\varepsilon)) \quad (25)$$

Now we found two $(l+1)\times(l+1)$ dimension invertible matrices in the above formula. In order to reduce the computation cost, we use SMW method theorem to solve the inverse matrix.

Let

$$\boldsymbol{L} = c_1\boldsymbol{N}^{\mathrm{T}}\boldsymbol{N} + \boldsymbol{M}^{\mathrm{T}}\boldsymbol{M} + c_2\boldsymbol{O}^{\mathrm{T}}\boldsymbol{O} \quad (26)$$

$$\boldsymbol{R} = c_3\boldsymbol{M}^{\mathrm{T}}\boldsymbol{M} + \boldsymbol{N}^{\mathrm{T}}\boldsymbol{N} + c_4\boldsymbol{O}^{\mathrm{T}}\boldsymbol{O} \quad (27)$$

Then

$$\begin{aligned} \hat{\boldsymbol{L}}^{-1} &= (\boldsymbol{L} + \boldsymbol{S}'^{\mathrm{T}}\boldsymbol{S}')^{-1} = \boldsymbol{L}^{-1} - \boldsymbol{L}^{-1}\boldsymbol{S}'^{\mathrm{T}}(1+\boldsymbol{S}'\boldsymbol{L}^{-1}\boldsymbol{S}'^{\mathrm{T}})\boldsymbol{S}'\boldsymbol{L}^{-1} \\ &= \boldsymbol{L}^{-1} - \frac{\boldsymbol{L}^{-1}\boldsymbol{S}'^{\mathrm{T}}}{1+\boldsymbol{S}'\boldsymbol{L}^{-1}\boldsymbol{S}'^{\mathrm{T}}}\boldsymbol{S}'\boldsymbol{L}^{-1} = \boldsymbol{L}^{-1} - \frac{\varphi_1}{1+\boldsymbol{S}'\varphi_1}\boldsymbol{S}'\boldsymbol{L}^{-1} \\ &= \boldsymbol{L}^{-1} - \phi_1\boldsymbol{S}'\boldsymbol{L}^{-1} = (\boldsymbol{I} - \phi_1\boldsymbol{S}')\boldsymbol{L}^{-1} = \xi_1\boldsymbol{L}^{-1} \end{aligned} \quad (28)$$

$$\begin{aligned} \hat{\boldsymbol{R}}^{-1} &= (\boldsymbol{R} + \boldsymbol{S}'^{\mathrm{T}}\boldsymbol{S}')^{-1} = \boldsymbol{R}^{-1} - \boldsymbol{R}^{-1}\boldsymbol{S}'^{\mathrm{T}}(1+\boldsymbol{S}'\boldsymbol{R}^{-1}\boldsymbol{S}'^{\mathrm{T}})\boldsymbol{S}'\boldsymbol{R}^{-1} \\ &= \boldsymbol{R}^{-1} - \frac{\boldsymbol{R}^{-1}\boldsymbol{S}'^{\mathrm{T}}}{1+\boldsymbol{S}'\boldsymbol{R}^{-1}\boldsymbol{S}'^{\mathrm{T}}}\boldsymbol{R}\boldsymbol{L}^{-1} = \boldsymbol{R}^{-1} - \frac{\varphi_2}{1+\boldsymbol{S}'\varphi_2}\boldsymbol{S}'\boldsymbol{R}^{-1} \\ &= \boldsymbol{R}^{-1} - \phi_2\boldsymbol{S}'\boldsymbol{R}^{-1} = (\boldsymbol{I} - \phi_2\boldsymbol{S}')\boldsymbol{R}^{-1} = \xi_2\boldsymbol{R}^{-1} \end{aligned} \quad (29)$$

Where, $\varphi_1 = \boldsymbol{L}^{-1}\boldsymbol{S}'^{\mathrm{T}}$, $\varphi_2 = \boldsymbol{R}^{-1}\boldsymbol{S}'^{\mathrm{T}}$, $\phi_i = \frac{\varphi_i}{1+\boldsymbol{S}'\varphi_i}, (i=1,2)$, $\xi_i = \boldsymbol{I} - \phi_i\boldsymbol{S}', (i=1,2)$ .Then the formula (25) and formula (26) can be

$$\hat{\boldsymbol{u}}_1 = -\hat{\boldsymbol{L}}^{-1}(c_1\boldsymbol{N}^{\mathrm{T}}e_2 + c_2\boldsymbol{O}^{\mathrm{T}}e_3(1-\varepsilon)) \quad (30)$$

$$\hat{\boldsymbol{u}}_2 = \hat{\boldsymbol{R}}^{-1}(c_3\boldsymbol{M}^{\mathrm{T}}e_1 + c_4\boldsymbol{O}^{\mathrm{T}}e_3(1-\varepsilon)) \quad (31)$$

Thus, we just need to solve the $\boldsymbol{L}^{-1}$ and $\boldsymbol{R}^{-1}$ to get these inverse matrixes updating matrixes.

The solving process of $\boldsymbol{L}^{-1}$ is as follows.

Let $\boldsymbol{J}=(\boldsymbol{M}^{\mathrm{T}}\boldsymbol{M}+c_2\boldsymbol{O}^{\mathrm{T}}\boldsymbol{O})^{-1}$, $\boldsymbol{P}=(\boldsymbol{N}^{\mathrm{T}}\boldsymbol{N}+c_1\boldsymbol{O}^{\mathrm{T}}\boldsymbol{O})^{-1}$, using SMW method theorem, there is

$$\boldsymbol{L}^{-1}=\boldsymbol{J}-\boldsymbol{J}\boldsymbol{N}^{\mathrm{T}}(\frac{\boldsymbol{I}}{c_1}+\boldsymbol{N}\boldsymbol{J}\boldsymbol{N}^{\mathrm{T}})^{-1}\boldsymbol{N}\boldsymbol{J} \tag{32}$$

Similarly to L^{-1} , R^{-1} can be expressed as

$$\boldsymbol{R}^{-1}=\boldsymbol{P}-\boldsymbol{P}\boldsymbol{M}^{\mathrm{T}}(\frac{\boldsymbol{I}}{c_3}+\boldsymbol{M}\boldsymbol{P}\boldsymbol{M}^{\mathrm{T}})^{-1}\boldsymbol{M}\boldsymbol{P} \tag{33}$$

3.3 Decremental learning of LST-KSVC

When constantly add new data to a certain degree, storage space will be occupied larger. Besides the original data is no longer useful, so need to delete the data to reduce the storage space. Decremental learning algorithm delete data without affecting the original classification results and is the inverse process of incremental learning. In this section, we imitate the incremental learning algorithm to conduct decremental learning algorithm. In the case of nonlinear, removing a class $\boldsymbol{S}_i'=[K(x_i,D^T)\quad e]^T$, for example. Similarly to formula (28)-(29), the deremenrtal expression is

$$\begin{aligned}\hat{\boldsymbol{L}}_{-}^{-1}&=(\boldsymbol{L}-\boldsymbol{S}'^{\mathrm{T}}\boldsymbol{S}')^{-1}=\boldsymbol{L}^{-1}+\boldsymbol{L}^{-1}\boldsymbol{S}'^{\mathrm{T}}(1-\boldsymbol{S}'\boldsymbol{L}^{-1}\boldsymbol{S}'^{\mathrm{T}})\boldsymbol{S}'\boldsymbol{L}^{-1}\\&=\boldsymbol{L}^{-1}+\frac{\boldsymbol{L}^{-1}\boldsymbol{S}'^{\mathrm{T}}}{1-\boldsymbol{S}'\boldsymbol{L}^{-1}\boldsymbol{S}'^{\mathrm{T}}}\boldsymbol{S}'\boldsymbol{L}^{-1}=\boldsymbol{L}^{-1}+\frac{\varphi_{-1}}{1-\boldsymbol{S}'\varphi_1}\boldsymbol{S}'\boldsymbol{L}^{-1}\\&=\boldsymbol{L}^{-1}+\beta_{-1}\boldsymbol{S}'\boldsymbol{L}^{-1}=(\boldsymbol{I}+\beta_{-1}\boldsymbol{S}')\boldsymbol{L}^{-1}=\phi_{-1}\boldsymbol{L}^{-1}\end{aligned} \tag{34}$$

$$\begin{aligned}\hat{\boldsymbol{R}}_{-}^{-1}&=(\boldsymbol{R}-\boldsymbol{S}'^{\mathrm{T}}\boldsymbol{S}')^{-1}=\boldsymbol{R}^{-1}+\boldsymbol{R}^{-1}\boldsymbol{S}'^{\mathrm{T}}(1-\boldsymbol{S}'\boldsymbol{R}^{-1}\boldsymbol{S}'^{\mathrm{T}})\boldsymbol{S}'\boldsymbol{R}^{-1}\\&=\boldsymbol{R}^{-1}+\frac{\boldsymbol{R}^{-1}\boldsymbol{S}'^{\mathrm{T}}}{1-\boldsymbol{S}'\boldsymbol{R}^{-1}\boldsymbol{S}'^{\mathrm{T}}}\boldsymbol{S}'\boldsymbol{R}^{-1}=\boldsymbol{R}^{-1}+\frac{\varphi_{-2}}{1-\boldsymbol{S}'\varphi_2}\boldsymbol{S}'\boldsymbol{R}^{-1}\\&=\boldsymbol{R}^{-1}+\beta_{-2}\boldsymbol{S}'\boldsymbol{R}^{-1}=(\boldsymbol{I}+\beta_{-2}\boldsymbol{S}')\boldsymbol{R}^{-1}=\phi_{-2}\boldsymbol{R}^{-1}\end{aligned} \tag{35}$$

Thus, we only need to recalculate ϕ_{-1} and ϕ_{-2} , and make use of $\boldsymbol{L}^{-1}$ and $\boldsymbol{R}^{-1}$, which has been required in the incremental learning process to gain the updating result.

4 Experiment and result

The validity of this algorithm has been tested in two experiments. All experiments are based on a 2.5 GHz pentium dual-core processor, 2GB memory hardware platform and using MATLAB 2013b to write programs.

Table 1. datasets in experiment.

datasets	Data scale	class
Hayes-roth	132×6=792	3
Iris	150×5=450	3
wine	178×14=2492	3
Soybean	47×35=1645	4
Ecoli	361×7=2289	5
Glass	214×9=1926	6
Determatology	357×34=12138	6

Experiment 1: selecting 10 group on UCI datasets, which is widely used for classification as showed in Table 1. Noting the dimensions of the car dataset is very high. We will leave it in the second experiments. Article in reference [12] compared LST - KSVC, K - SVCR, TSVM, Twin - KSVC and one-to-rest TSVM algorithms, the results show that the LST - KSVC has a great advantage in classification accuracy and the algorithm running time. Therefore, the first experiment mainly compares for ILST - KSVC and LST - KSVC training time and recognition accuracy.

We use 10-fold cross-validation and Gaussian kernel function $K(x_i,x_j)=\exp(-\left\|x_i-x_j\right\|^2/p)$ to convey the effect of different datasets' classification. In this paper, we choose penalty parameters c_1=c_3, c_2=c_4, c_1, c_2 are belong to $\{2^i\mid i=-4,-2,0,1,2,4,6,8\}$, ε is belonging to {0.1, 0.2, 0.3, 0.4, 0.5}, choose p in $\{2^i\mid i=-4,-2,0,1,2,4,6,8\}$.

Table 2 shows the LST - KSVC and ILST - KSVC classification results. Analysis of Table 2, on the left side of the ternary classification data, ILST KSVC can always get with LST - KSVC compare to the classification accuracy and didn't take too much running time.Table 2 on the right is the training data, which category number is more than 3. In the same way, it's easy to observe and obtain the same conclusion as the left side. Experiment show that ILST KSVC recognition accuracy can match with LST - KSVC, and training time is not that long for multiple classification.

Table 2. Muti-class classification.

dataset	LST-KSVC acc% ±std time/s(×10-2)	ILST-KSVC acc% ±std time/s(×10-2)
Hayes-roth	90.0±8.15 0.028	91.5±5.67 0.96
Iris	98.0±3.22 0.024	98.5±4.51 2.50
wine	98.86 ±4.80 0.03	98.86 ±4.11 1.49
Soybean	100±0.00 0.08	100±0.00 1.50
Ecoli	88.93±7.91 0.032	88.21±4.05 8.30
Glass	68.40±6.40 0.040	57.14±1.96 8.20
Determatology	95.67±2.89 0.17	95.07±3.83 34.57

Experiment 2: the increment algorithm of binary classification time-consuming is very big when dealing with high-dimensional data, this paper puts forward the incremental algorithm based on ternary classification problem to do with thousands of dimensional data, which only need a short period of time.This Experiment will increase the dimension from low to high gradually on a data named car which dimension is 1728 * 6.Figure 1 and Figure 2 compare LST KSVC, which apply to offline environment for ILST KSVC that suitable for the online environment in classification accuracy and running time, respectively.From figure 1 we can see: with increasing of data dimension, the accuracy of both methods is reduced, but always stay above 90%, and has little difference. Figure 2 is the contrast of these two algorithms, it can be

seen that with the continuous improvement of data dimension, training time also increases, but still at a low level.Thus, the incremental algorithm proposed in this paper can be employed for a number of high-dimensional data processing without affecting the classification results.

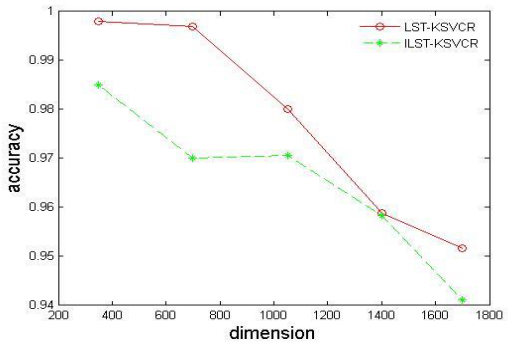

Figure 1. Contrast on accuracy.

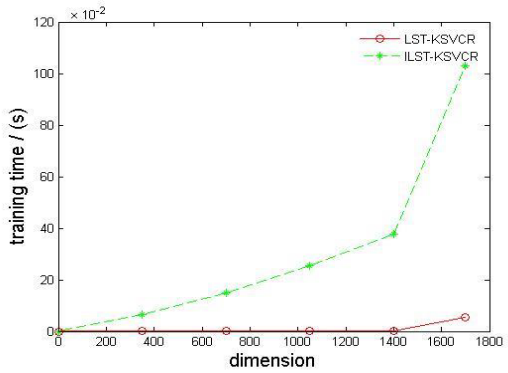

Figure 2. Contrast on training time.

5 Ending

This paper proposed an incremental learning algorithm (ILST - KSVC) suitable for the online environment by solving inverse matrixs based on least squares twin multi-class classification support vector machine (LST - KSVC). ILST - KSVC use historical data to update training data avoiding the repeated calculation, and reducing computational complexity. Selecting 10 groups of UCI data sets, based on MATLAB simulation platform for the two groups of experiments, the experimental results show that the ILST - KSCV inherits the advantages of LST - KSVC that has faster training speed and a high classification recognition rate when dealing with multiple classification data not only suitable for low dimensional data processing, also can be applied to the processing of high-dimensional data.

References

1. C. Cortes, V. Vapnik. M. L., *Support vector networks* **20**, 273-297 (1995)
2. W. Qingzhu, Z. Wenchao, W. Bin, Journal of Medical Systems **39**, 171 (2015)
3. Z. Ruijie, L. Bicheng, W. Fushan, Chinese Jounal of Electronics **42,** 646-652 (2014)
4. G. Farong, W. Jiajia, X. Xu-gang, et al., Jounal of Electronics & Information Technology **37**, 1154-1159 (2015)
5. D. K. Renuka, and P. Visalakshi, Journal of Scientific & Industrial Research **73**, 437-442 (2014)
6. K. R. Jayadeva, S. Chandra, Pattern Analysis & Machine Intelligence IEEE Transactions on **29**, 905-910 (2007)
7. K. M. Arun, M. Gopal, Expert Systems with Applications **36**, 7535-7543 (2009)
8. L. Bottou, C. Cortes, J. S. Denker, et al., *International Conferenceon Pattern Recognition, IEEE Computer Society Press*, 77-82(1994)
9. H. G. Kreßel, *Advances in Kernel Methods*, 255-268 (1999)
10. C. Angulo, X. Parra, A. Catala, Neurocomputing, KSVCR: a support vector machine for multi-class classification 55, 57-77 (2003)
11. X. Yitian, G. Rui and W. Laisheng, Cognitive computation **5**, 580-588 (2013)
12. J. A. Nasiri, N. M. Charkari and S. Jalili, Pattern Recognition **48**, 984-992 (2015)
13. C. P. Diehl, and G. Cauwenberghs, *International Joint Conference on Neural Networks*, 2685-2690 (2003)
14. Z. Guanhua, and H. Min, *16th International Conference on Management Science and Engineering*, 95-100 (2009)
15. F. Orabona, C. Castellini, B. Caputo, et al., Pattern Recognition On-line independent support vector machines **43**, 1402-1412 (2010)
16. L. Yang, L. Kai, et al., ICACI, 18-20 (2012)