NLS-TSTM: A Novel and Fast Nonlinear Image Classification Method

XI ZHAN GAO
Liaocheng University
School of Mathematics Sciences
252059,Liaocheng
CHINA
gaoxizhan123@126.com;

LIYA FAN
Liaocheng University
School of Mathematics Sciences
252059,Liaocheng
CHINA
Corresponding author:fanliya63@126.com;

HAITAO XU
Liaocheng University
School of Mathematics Sciences
252059,Liaocheng
CHINA
xuhaitao@live.com

Abstract: This paper is devoted to study nonlinear image classification methods based on support tensor machine (STM). Firstly, a new linear method named linear least squares twin support tensor machine (LLS-TSTM) is proposed, which is an improvement of linear STM. The utility of twin skill and least squares technology aims to speed up the computation time (sum of training time and testing time). Secondly, in order to study nonlinear version of LS-TSTM, a new matrix kernel function is introduced and then based on which, a nonlinear LS-TSTM (NLS-TSTM) classification method is suggested with detailed theoretical derivation. Finally, in order to examine the effectiveness of LLS- and NLS-TSTM, we perform a series of comparative experiments with linear STM and linear TSTM on ORL and Yale face databases. Experiment results show that the proposed methods are effective and efficient.

Key–Words: Least squares twin support tensor machine; image classification; matrix kernel function; iterative algorithm; classification accuracy

1 Introduction

In real world, high-dimensional image data with structural information are often encountered when we deal with pattern classification problems. How to utilize the structural information is an important work. The most existing classification methods are oriented to vector, such as support vector machine (SVM) [1], twin support vector machine (TSVM) [2], least squares support vector machine (LS-SVM) [3-4], least squares twin support vector machine (LS-TSVM) [5] and so on [6-17]. SVM aims to find an optimal separating hyperplane by maximizing the margin of a pair of parallel boundary hyperplanes for binary classification. Different from SVM, TSVM aims to find a pair of nonparallel hyperplanes by solving two smaller quadratic programming problems (QPPs) such that each plane is closer to one of two classes and at least one distance from the other. TSVM has the similar formulation with SVM except that not all the patterns appear in the constraints of either problem at the same time, which makes the learning speed of TSVM is more faster than that of SVM. LS-TSVM is a least squares version of TSVM, in which the inequality constraints are replaced by equality constraints and the 1-norm of slack variables is replaced by the square of 2-norm. This leads to solve two modified primal problems instead of dual problems in LS-TSVM. Experiments in [5] show that LS-TSVM is faster than TSVM. In addition to this, there also exist some vector pretreatment methods before we use the above classifiers directly, such as linear discriminant analysis (LDA) [18], principal component analysis (PCA) [19], kernel discriminant analysis (KDA) [20], Fisher discriminant analysis (FDA) [21]and so on.

In recent years, some interests about tensor representation of image data have been gain attention. A tensor-based learning framework linear support tensor machine (STM) was proposed by Tao et al [22] and Cai et al [23], which directly accepts tensors as input in the learning model without vectorization. The use of tensor representation helps overcome the overfitting problem encountered mostly in vector-based learning. Some linear classification algorithms have been developed for pattern classification [24-32]. However, up to now nonlinear classification methods based on STM for image data are not seen more.

Motivated by works above, in this paper, we will first introduce a new tensor version extension for linear LS-TSVM (LLS-TSVM), termed as linear least squares twin support tensor machine (LLS-TSTM). The utility of twin skill and least squares technology aims to speed up the computation time (sum of training time and testing time). Then, we introduce a novel matrix kernel function and, base on which, suggest a nonlinear LS-TSTM (NLS-TSTM) classification method with detailed theoretical derivation. In order to examine the effectiveness of LLS-TSTM and NLS-TSTM, we perform a series of comparative experiments with linear STM and linear TSTM on ORL and Yale face databases.

The rest of the paper is organized as follows. In Section 2, background and related works are introduced. In Section 3, based on a new defined matrix kernel function, a LLS-TSTM classification algorithm is proposed in an iterative framework and a NLS-TSTM method is suggested with detailed theoretical derivation. Experiments and results analysis are performed in Section 4. Section 5 gives some conclusions.

2 Background and related works

2.1 Linear twin support vector machines (Linear TSVM)

The basic idea of linear TSVM is constructing a pair of nonparallel hyperplanes such that each one is as close as possible to one class, and as far as possible from the other class. A new input will be assigned to one of the classes depending on its proximity to each hyperplane. Let $T = \{(x_i, y_i)\}_{i=1}^m$ be a set of binary sample data, where $x_i \in R^n$ and $y_i \in \{\pm 1\}$ are the input and the class label of the ith sample, respectively. Let m_1 and m_2 be the numbers of positive and negative samples, respectively, then $m = m_1 + m_2$. We denote $A \in R^{n\times m_1}$ and $B \in R^{n\times m_2}$ the matrices composed of positive and negative samples, respectively. Linear TSVM seeks a pair of nonparallel hyperplanes $f_1(x) = w_+^T x + b_+ = 0$ and $f_2(x) = w_-^T x + b_- = 0$ by considering the following two quadratic programming problems (QPPs):

$$\begin{aligned} \min_{w_+, b_+, \xi_2} \ & \tfrac{1}{2}||A^T w_+ + e_1 b_+||^2 + c_1 e_2^T \xi_2 \\ s.t. \ & -(B^T w_+ + e_2 b_+) \geq e_2 - \xi_2, \\ & \xi_2 \geq 0, \end{aligned} \tag{1}$$

$$\begin{aligned} \min_{w_-, b_-, \xi_1} \ & \tfrac{1}{2}||B^T w_- + e_2 b_-||^2 + c_2 e_1^T \xi_1 \\ s.t. \ & (A^T w_- + e_1 b_-) \geq e_1 - \xi_1, \\ & \xi_1 \geq 0, \end{aligned} \tag{2}$$

where $c_1, c_2 > 0$ are tradeoff parameters, $\xi_1 \in R^{m_1}, \xi_2 \in R^{m_2}$ are slack vectors and $e_1 \in R^{m_1}, e_2 \in R^{m_2}$ are vectors of ones. By solving the Wolfe dual forms of the problems (1) and (2), we can obtain (w_+, b_+) and (w_-, b_-). A new input $\widetilde{x}$ can be assigned the class $k (k = +, -)$ depending on which of the two hyperplanes is closer to, that is, $k = \arg\min\{\frac{|f_1(\widetilde{x})|}{||w_+||}, \frac{|f_2(\widetilde{x})|}{||w_-||}\}$.

2.2 Linear least squares twin support vector machines (LLS-TSVM)

LLS-TSVM is an improvement of linear TSVM, in which inequality constraints are replaced by equality constraints and 1-norm with weight c of slack variables is replaced by the square of 2-norm with weight $\frac{c}{2}$. The optimization problems corresponding to LLS-TSVM can be stated as follows.

$$\begin{aligned} \min_{w_+, b_+, \xi_2} \ & \tfrac{1}{2}||A^T w_+ + e_1 b_+||^2 + \tfrac{c_1}{2}\xi_2^T \xi_2 \\ s.t. \ & -(B^T w_+ + e_2 b_+) = e_2 - \xi_2, \end{aligned} \tag{3}$$

$$\begin{aligned} \min_{w_-, b_-, \xi_1} \ & \tfrac{1}{2}||B^T w_- + e_2 b_-||^2 + \tfrac{c_2}{2}\xi_1^T \xi_1 \\ s.t. \ & (A^T w_- + e_1 b_-) = e_1 - \xi_1. \end{aligned} \tag{4}$$

By solving directly the problems (3) and (4) not their Wolfe dual forms, we can obtain (w_+, b_+) and (w_-, b_-).

2.3 Linear support tensor machines (Linear STM)

In this subsection, we recall briefly linear support tensor machines which is a tensor generalization of SVM in the tensor space, for details see [18-19]. Different from SVM, the main idea of STM is to deal with tensor inputs directly without vectorization, which keeps the structure information of image data and has not increase time complexity. Let $T = \{(X_i, y_i)\}_{i=1}^m$ be a set of second tensor sample data, where $X_i \in R^{n_1\times n_2}$ and $y_i \in \{\pm 1\}$ are the input and the class label of the ith second tensor sample , respectively. Linear STM finds a tensor classifier $f(X) = u^T X v + b$, where $u \in R^{n_1}, v \in R^{n_2}$ and $b \in R$, such that the two classes can be separated with maximum margin. The implement of STM is actually based on an iteration procedure, which is stated as follows.

Algorithm 1. Linear STM

1. Initialization. Let $t = 0, \varepsilon > 0$ be small enough and take $u^t = [1, \cdots, 1]^T \in R^{n_1}$.

2. Compute v^t. Let $x_i^t = X_i^T u^t, i = 1, \cdots, m$ and $\beta_1^t = ||u^t||^2$. (v^t, b_1^t) can be computed by solving the following optimization problem with $\beta_1 = \beta_1^t$ and $x_i = x_i^t$:

$$\begin{aligned} \min_{v, b_1, \xi} \ & \tfrac{1}{2}\beta_1 v^T v + C e^T \xi \\ s.t. \ & y_i(v^T x_i + b_1) + \xi_i \geq 1, \\ & \xi_i \geq 0, i = 1, \cdots, m, \end{aligned} \tag{5}$$

where $C > 0$ is a parameter, $\xi \in R^m$ is a slack vector and $e \in R^m$ is the vector of ones.

3. Update u^t. Let $\widetilde{x}_i^t = X_i v^t, i = 1, \cdots, m$ and $\beta_2^t = ||v^t||^2$. (u^{t+1}, b_2^{t+1}) can be computed by solving

the following optimization problem with $\beta_2 = \beta_2^t$ and $\widetilde{x}_i = \widetilde{x}_i^t$:

$$\begin{array}{ll} \min\limits_{u,b_2,\xi} \frac{1}{2}\beta_2 u^T u + Ce^T\xi \\ s.t.\ y_i(u^T\widetilde{x}_i + b_2) + \xi_i \geq 1, \\ \quad \xi_i \geq 0, i = 1, \cdots, m. \end{array} \tag{6}$$

4. Update v^t. Let $x_i^{t+1} = X_i^T u^{t+1}, i = 1, \cdots, m$ and $\beta_1^{t+1} = ||u^{t+1}||^2$. (v^{t+1}, b_1^{t+1}) can be computed by solving the problem (5) with $\beta_1 = \beta_1^{t+1}$ and $x_i = x_i^{t+1}$.

5. If $\|u^{t+1} - u^t\| < \varepsilon, \|v^{t+1} - v^t\| < \varepsilon$, or maximum number of iteration is achieved, put $u^* \leftarrow u^{t+1}$, $b_1^* \leftarrow b_1^{t+1}$, $v^* \leftarrow v^{t+1}$ and $b_2^* \leftarrow b_2^{t+1}$; otherwise, put $t \leftarrow t+1$ and return back step 2.

6. Compute $b^* = \frac{1}{2}(b_1^* + b_2^*)$ and construct the decision function $f(X) = (u^*)^T X v^* + b^*, \forall X \in R^{n_1 \times n_2}$.

3 Least squares twin support tensor machines (LS-TSTM)

We know that classifying image data by using vector-based classification methods must transform image inputs into vectors firstly. When a $m \times n$ image data is scanned into a mn dimensional vector, its structure information will be lost. In order to protect the structure information of image data, in this section, we study the tensor version extensions of LLS-TSVM. We first propose a linear version extension named as linear least squares twin support tensor machine (LLS-TSTM), and then introduce a new matrix kernel function and based on which suggest a nonlinear version extension named as nonlinear least squares twin support tensor machine (NLS-TSTM) with detailed theoretical derivation.

Let $T = \{(X_i, y_i)\}_{i=1}^m$ be a set of second order tensor sample data, where $X_i \in R^{n_1 \times n_2}$ and $y_i \in \{\pm 1\}$ are the input and the class label of the ith second tensor sample, respectively. Let m_1 and m_2 be the numbers of positive and negative samples and I_1 and I_2 be the index sets of positive and negative samples, respectively, then $m = m_1 + m_2, |I_1| = m_1$ and $|I_2| = m_2$. The norm of a matrix C is defined by $\|C\|^2 = Tr(C^T C)$, where $Tr(C)$ denotes the trace of the matrix C.

3.1 Linear LS-TSTM (LLS-TSTM)

LLS-TSTM aims to seek a pair of nonparallel hyperplanes $f_1(x) = u_+^T X v_+ + b_+ = 0$ and $f_2(x) = u_-^T X v_- + b_- = 0$, where $u_+, u_- \in R^{n_1}, v_+, v_- \in R^{n_2}$ and $b_+, b_- \in R$, by considering the following two QPPs:

$$\begin{array}{ll} \min\limits_{u_+,v_+,b_+,\xi} \frac{1}{2}\sum\limits_{i\in I_+}(u_+^T X_i v_+ + b_+)^2 + \frac{c_1}{2}\sum\limits_{j\in I_-}\xi_j^2 \\ s.t. \quad -(u_+^T X_j v_+ + b_+) + \xi_j = 1, j \in I_-, \end{array} \tag{7}$$

$$\begin{array}{ll} \min\limits_{u_-,v_-,b_-,\eta} \frac{1}{2}\sum\limits_{j\in I_-}(u_-^T X_j v_- + b_-)^2 + \frac{c_2}{2}\sum\limits_{i\in I_+}\eta_i^2 \\ s.t. \quad (u_-^T X_j v_- + b_-) + \eta_j = 1, i \in I_+. \end{array} \tag{8}$$

where $c_1, c_2 > 0$ are tradeoff parameters and $\xi \in R^{m_2}, \eta \in R^{m_1}$ are slack vectors. Once (w_+, b_+) and (w_-, b_-) are solved, a new input $\widetilde{X}$ can be assigned the class $k(k = +, -)$ depending on which of the two hyperplanes is closer to, that is, $k = \arg\min\{\frac{|f_1(\widetilde{X})|}{||u_+ v_+^T||}, \frac{|f_2(\widetilde{X})|}{||u_- v_-^T||}\}$.

Next, we solve (u_+, v_+, b_+) by an iterative framework. Similarly, we can solve (u_-, v_-, b_-).

We first discuss the relationships among u_+, v_+ and b_+. Substituting the equality constraints into the objective function of the problem (7), we have

$$\begin{array}{ll} \min\limits_{u_+,v_+,b_+,\xi} \frac{1}{2}\sum\limits_{i\in I_+}(u_+^T X_i v_+ + b_+)^2 \\ \quad + \frac{c_1}{2}\sum\limits_{j\in I_-}(u_+^T X_j v_+ + b_+ + 1)^2. \end{array} \tag{9}$$

Setting the gradient of the objective function of the problem (9) with respect to (u_+, v_+, b_+) be zero, we can deduce that

$$\begin{array}{l} \sum\limits_{i\in I_+} X_i v_+ v_+^T X_i^T u_+ + b_+ \sum\limits_{i\in I_+} X_i v_+ \\ \quad + c_1 \sum\limits_{j\in I_-} X_j v_+ v_+^T X_j^T u_+ \\ \quad + (b+1)c_1 \sum\limits_{j\in I_-} X_j v_+ = 0, \end{array}$$

$$\begin{array}{l} \sum\limits_{i\in I_+} X_i{}^T u_+ u_+^T X_i v_+ + b_+ \sum\limits_{i\in I_+} X_i{}^T u_+ \\ \quad + c_1 \sum\limits_{j\in I_-} X_j{}^T u_+ u_+^T X_j v_+ \\ \quad + (b+1)c_1 \sum\limits_{j\in I_-} X_j{}^T u_+ = 0, \end{array}$$

$$\begin{array}{l} b_+ = -c_0 \sum\limits_{i\in I_+} u_+^T X_i v_+ - c_0 c_1 \sum\limits_{j\in I_-} u_+^T X_j v_+ \\ \quad - c_0 c_1 m_2, \end{array}$$

where $c_0 = \frac{1}{m_1 + c_1 m_2}$. Proceeding to the next step, we can get $Hu_+ = Fv_+$ and $Gv_+ = F^T u_+$, where

$$\begin{aligned} H = &\sum_{i\in I_+} X_i v_+ v_+^T X_i^T + c_1 \sum_{j\in I_-} X_j v_+ v_+^T X_j^T \\ &-c_0 \sum_{i_1,i_2\in I_+} X_{i_1} v_+ v_+^T X_{i_2}^T \\ &-c_0 c_1 \sum_{i\in I_+, j\in I_-} X_i v_+ v_+^T X_j^T \\ &-c_0 c_1^2 \sum_{j_1,j_2\in I_-} X_{j_1} v_+ v_+^T X_{j_2}^T \\ &-c_0 c_1 \sum_{i\in I_+, j\in I_-} X_j v_+ v_+^T X_i^T, \end{aligned}$$

$$\begin{aligned} G = &\sum_{i\in I_+} {X_i}^T u_+ u_+^T X_i + c_1 \sum_{j\in I_-} {X_j}^T u_+ u_+^T X_j \\ &-c_0 \sum_{i_1,i_2\in I_+} X_{i_1}^T u_+ u_+^T X_{i_2} \\ &-c_0 c_1 \sum_{i\in I_+, j\in I_-} X_i^T u_+ u_+^T X_j \\ &-c_0 {c_1}^2 \sum_{j_1,j_2\in I_-} X_{j_1}^T u_+ u_+^T X_{j_2} \\ &-c_0 c_1 \sum_{i\in I_+, j\in I_-} X_j^T u_+ u_+^T X_i, \end{aligned}$$

$$\begin{aligned} F = &c_0 c_1 m_2 \sum_{i\in I_+} X_i + c_0 {c_1}^2 m_2 \sum_{j\in I_-} X_j \\ &-c_1 \sum_{j\in I_-} X_j. \end{aligned}$$

This indicates that u_+, v_+ and b_+ are dependent on each other and cannot be solved independently. Hence, we utilize an iteration method to solve (u_+, v_+, b_+), which can be described as follows.

For any given nonzero vector $v_+ \in R^{n_2}$, let $x_i^T = X_i v_+, i = 1, \cdots, m$ and $A_1 \in R^{m_1\times n_1}$ and $B_1 \in R^{m_2\times n_1}$ be matrices composed of all $x_i, i \in I_+$ and all $x_j, j \in I_-$, respectively, then the problem (7) can be simplified as

$$\begin{aligned} \min_{u_+,b_+,\xi} &\ \tfrac{1}{2}\|A_1u_+ + e_1b_+\|^2 + \tfrac{c_1}{2}\xi^T\xi \\ s.t. &\ -(B_1u_+ + e_2b_+) + \xi = e_2, \end{aligned} \quad (10)$$

where $e_1 \in R^{n_1}, e_2 \in R^{n_2}$ are vectors of ones. Substituting the equality constraints into the objective function of the problem (10) and setting the gradient of the objective function with respect to u_+ and b_+ to zero, we can deduce that

$$\begin{aligned} &A_1^T(A_1u_+ + e_1b_+) + c_1B_1^T(B_1u_+ + e_2b_+ + e_2) \\ &= 0, \\ &e_1^T(A_1u_+ + e_1b_+) + c_1e_2^T(B_1u_+ + e_2b_+ + e_2) \\ &= 0. \end{aligned} \quad (11)$$

Let $E_1 = [A_1\ e_1]$ and $F_1 = [B_1\ e_2]$. We can obtain from (11) that

$$\begin{bmatrix} u_+ \\ b_+ \end{bmatrix} = -\left[\tfrac{1}{c_1}E_1^TE_1 + F_1^TF_1\right]^{-1}F_1^Te_2. \quad (12)$$

Here, if $\frac{1}{c_1}E_1^TE_1 + F_1^TF_1$ is singular, we use $\frac{1}{c_1}E_1^TE_1 + F_1^TF_1 + \varepsilon I$ instead of $\frac{1}{c_1}E_1^TE_1 + F_1^TF_1$, where $\varepsilon > 0$ is a constant and I is an identity matrix of appropriate dimension.

For any given nonzero vector $u_+ \in R^{n_1}$, let $\tilde{x}_i^T = X_i^T u_+, i = 1, \cdots, m$ and $A_2 \in R^{m_1\times n_2}$ and $B_2 \in R^{m_2\times n_2}$ be matrices composed of all $\tilde{x}_i, i \in I_+$ and all $\tilde{x}_j, j \in I_-$, respectively. In this case, the problem (7) can be simplified as

$$\begin{aligned} \min_{v_+,d_+,\xi} &\ \tfrac{1}{2}\|A_2v_+ + e_1d_+\|^2 + \tfrac{c_1}{2}\xi^T\xi \\ s.t. &\ -(B_2v_+ + e_2d_+) + \xi = e_2. \end{aligned} \quad (13)$$

By solving the problem (13), we can obtain

$$\begin{bmatrix} v_+ \\ d_+ \end{bmatrix} = -\left[\tfrac{1}{c}{}_1E_2^TE_2 + F_2^TF_2\right]^{-1}F_2^Te_2, \quad (14)$$

where $E_2 = [A_2, e_1]$ and $F_2 = [B_2, e_2]$.

By solving the problem (8) with the similar way above, we can get that

$$\begin{bmatrix} u_- \\ b_- \end{bmatrix} = \left[\tfrac{1}{c_2}F_1^TF_1 + E_1^TE_1\right]^{-1}E_1^Te_1. \quad (15)$$

$$\begin{bmatrix} v_- \\ d_- \end{bmatrix} = \left[\tfrac{1}{c_2}F_2^TF_2 + E_2^TE_2\right]^{-1}E_2^Te_1. \quad (16)$$

Summing up the discussion above, we know that u_i, v_i and $b_i, i = \pm$ can be obtained by iteratively solving (12) and (14), and(15)-(16). The specific procedure is as follows.

Algorithm 2. Linear LS-TSTM

1. Initialization. Let $t = 0, \varepsilon > 0$ small enough and take $v_+^t = v_-^t = [1, \cdots, 1]^T \in R^{n_2}$.

2. Calculate u and b. Calculate (u_+^t, b_+^t) by using (12) and (u_-^t, b_-^t) by using (15) with $v_+ = v_+^t$ and $v_- = v_-^t$.

3. Update v and d. Calculate (v_+^{t+1}, d_+^{t+1}) by using (14) and (v_-^{t+1}, d_-^{t+1}) by using (16) with $u_+ = u_+^t$ and $u_- = u_-^t$.

4. Update u and b. Calculate (u_+^{t+1}, b_+^{t+1}) by using (12) and (u_-^{t+1}, b_-^{t+1}) by using (15) with $v_+ = v_+^{t+1}$ and $v_- = v_-^{t+1}$.

5. If $\|u_+^{t+1} - u_+^t\|, \|u_-^{t+1} - u_-^t\|, \|v_+^{t+1} - v_+^t\|$ and $\|v_-^{t+1} - v_-^t\|$ are all less than ε or the maximum number of iterations is achieved, put $u_+^* \leftarrow u_+^{t+1}, v_+^* \leftarrow v_+^{t+1}$, $u_-^* \leftarrow u_-^{t+1}, v_-^* \leftarrow v_-^{t+1}$, $b_+^* \leftarrow \frac{b_+^{t+1}+d_+^{t+1}}{2}$ and $b_-^* \leftarrow \frac{b_-^{t+1}+d_-^{t+1}}{2}$; otherwise, set $t \leftarrow t + 1$ and return to step 2.

6. Construct decision functions $f_i(X), i = 1,2$ by $f_1(X) = (u_+^*)^T \widetilde{X} v_+^* + b_+^*$ and $f_2(X) = (u_-^*)^T \widetilde{X} v_-^* + b_-^*$.

7. For a new input $\widetilde{X}$, its label $y_{\widetilde{X}}$ can be obtained by $y_{\widetilde{X}} = \arg\min_{i=\pm} \frac{|f_i(\widetilde{X})|}{\|u_i^*(v_i^*)^T\|}$.

3.2 Nonlinear LS-TSTM (NLS-TSTM)

In this subsection, we discuss a nonlinear tensor version extension for LS-TSVM by introducing a new matrix kernel function. Kernel skill is used to preprocess image data. Let $k : R^{n_1} \times R^{n_1} \to R$ be a kernel function, H (for the sake of consistency, denoted as R^∞) the reproducing kernel Hilbert space (RKHS) of the kernel k and $\varphi : R^{n_1} \to R^\infty$ the corresponding feature mapping. Let $\aleph = \{X_1, \cdots, X_m\}, A = \{X_1^1, \cdots, X_{m_1}^1\}$ and $B = \{X_1^2, \cdots, X_{m_2}^2\}$, where X_i^1 and X_j^2 express the ith input belonging to positive class and the jth input belonging to negative class, respectively.

We first introduce a matrix kernel function. For any matrix $X \in R^{n_1 \times n_2}$, it can be divided by columns as $X = [x_1, , \cdots, x_{n_2}]$, where $x_i \in R^{n_1}$ is the ith column of X. Put $\varphi(X) = [\varphi(x_1), \cdots, \varphi(x_{n_2})] \in R^{\infty \times n_2}$. For any given nonzero vector $v \in R^{n_2}$, we define a matrix kernel function $k_v : R^{n_1 \times n_2} \times R^{n_1 \times n_2} \to R$ with respect to v by

$$\begin{aligned} k_v(X,Z) &= <\varphi(X)v, \varphi(Z)v> = v^T \varphi(X)^T \varphi(Z) v \\ &= v^T K_{XZ} v, \ \forall X, Z \in R^{n_1 \times n_2}, \end{aligned}$$

where

$$\begin{aligned} K_{XZ} &= \varphi(X)^T \varphi(Z) = [\varphi(x_i)^T \varphi(z_j)]_{n_2 \times n_2} \\ &= [k(x_i, z_j)]_{n_2 \times n_2}. \end{aligned}$$

To facilitate the following derivation, we set

$$K_v(A,\aleph) = \begin{bmatrix} k_v(X_1^1, X_1) & \cdots & k_v(X_1^1, X_m) \\ \vdots & \ddots & \vdots \\ k_v(X_{m_1}^1, X_1) & \cdots & k_v(X_{m_1}^1, X_m) \end{bmatrix},$$

$$K_v(B,\aleph) = \begin{bmatrix} k_v(X_1^2, X_1) & \cdots & k_v(X_1^2, X_m) \\ \vdots & \ddots & \vdots \\ k_v(X_{m_2}^2, X_m) & \cdots & k_v(X_{m_2}^2, X_m) \end{bmatrix},$$

$$K_v(\aleph,\aleph) = \begin{bmatrix} k_v(X_1, X_1) & \cdots & k_v(X_1, X_m) \\ \vdots & \ddots & \vdots \\ k_v(X_m, X_1) & \cdots & k_v(X_m, X_m) \end{bmatrix},$$

$$\begin{aligned} K_v(X,\aleph) &= [k_v(X, X_1), \cdots, k_v(X, X_m)] \in R^{1 \times m}, \\ &\forall X \in R^{n_1 \times n_2}. \end{aligned}$$

We consider u_k in the subspace $span\{\varphi(X_1)v_k, \cdots, \varphi(X_m)v_k\}$ of R^∞ and assume that $u_k = \varphi_{v_k}(\aleph)\beta_k$, where $\varphi_{v_k}(\aleph) = [\varphi(X_1)v_k, \cdots, \varphi(X_m)v_k] \in R^{\infty \times m}, \beta_k \in R^m$ and $k = \pm$. Similarly to LLS-TSTM, NLS-TSTM aims to seek a pair of nonparallel hyperplanes $u_+^T \varphi(X) v_+ + b_+ = 0$ and $u_-^T \varphi(X) v_- + b_- = 0$, where $u_+, u_- \in R^\infty, v_+, v_- \in R^{n_2}$ and $b_+, b_- \in R$, by considering the following two QPPs:

$$\begin{aligned} \min_{u_+, v_+, b_+, \xi} \ & \tfrac{1}{2} \sum_{i \in I_+} (u_+^T \varphi(X_i) v_+ + b_+)^2 + \tfrac{c_1}{2} \sum_{j \in I_-} \xi_j^2 \\ s.t. \ & -(u_+^T \varphi(X_j) v_+ + b_+) + \xi_j = 1, \ j \in I_-, \end{aligned} \tag{17}$$

$$\begin{aligned} \min_{u_-, v_-, b_-, \eta} \ & \tfrac{1}{2} \sum_{j \in I_-} (u_-^T \varphi(X_j) v_- + b_-)^2 + \tfrac{c_2}{2} \sum_{i \in I_+} \eta_i^2 \\ s.t. \ & (u_-^T \varphi(X_i) v_- + b_-) + \eta_i = 1, \ i \in I_+. \end{aligned} \tag{18}$$

Next, we utilize an iterative method to solve $(u_i, v_i, b_i), i = \pm$.

For any given nonzero vector $v_1 \in R^{n_2}$, let

$$\varphi_{v_+}(A) = [\varphi(X_1^1)v_+, \cdots, \varphi(X_{m_1}^1)v_+],$$

$$\varphi_{v_+}(B) = [\varphi(X_1^2)v_+, \cdots, \varphi(X_{m_2}^2)v_+],$$

$$\varphi_{v_+}(\aleph) = [\varphi(X_1)v_+, \cdots, \varphi(X_m)v_+].$$

then the problem (17) can be simplified as

$$\begin{aligned} \min_{\beta_+, b_+, \xi} \ & \tfrac{1}{2} \|K_{v_+}(A,\aleph)\beta_+ + b_+ e_1\|^2 + \tfrac{c_1}{2} \xi^T \xi \\ s.t. \ & -(K_{v_+}(B,\aleph)\beta_+ + b_+ e_2) + \xi = e_2, \end{aligned} \tag{19}$$

where $\varphi_{v_+}^T(A)u_+ = K_{v_+}(A,\aleph)\beta_+$ and $\varphi_{v_+}^T(B)u_+ = K_{v_+}(B,\aleph)\beta_+$. Substituting the equality constraint into the objective function of the problem (19) and setting the gradient of the objective function with respect to β_+ and b_+ to zero, we can deduce that

$$\begin{bmatrix} \beta_+ \\ b_+ \end{bmatrix} = -(\tfrac{1}{c_1} G_1^T G_1 + H_1^T H_1)^{-1} H_1^T e_2, \tag{20}$$

where $G_1 = [K_{v_+}(A,\aleph), e_1]$ and $H_1 = [K_{v_+}(B,\aleph), e_2]$. Once β_+ is calculated, we let

$$u_+ = \varphi_{v_+}(\aleph)\beta_+,$$

$$\varphi_{\beta_+}(A) = [\varphi^T(X_1^1)\varphi_{v_+}(\aleph)\beta_+, \cdots, \varphi^T(X_{m_1}^1)\varphi_{v_+}(\aleph)\beta_+],$$

$$\varphi_{\beta_+}(B) = [\varphi^T(X_1^2)\varphi_{v_+}(\aleph)\beta_+, \cdots, \varphi^T(X_{m_2}^2)\varphi_{v_+}(\aleph)\beta_+],$$

and set

$$G_2 = [\varphi^T(X_1^1)\varphi_{v_+}(\aleph)\beta_+, \cdots, \varphi^T(X_{m_1}^1)\varphi_{v_+}(\aleph)\beta_+^T, e_1],$$

$$H_2 = [\varphi^T(X_1^2)\varphi_{v_+}(\aleph)\beta_+, \cdots, \varphi^T(X_{m_2}^2)\varphi_{v_+}(\aleph)\beta_+^T, e_2].$$

In this case, the problem (17) can be simplified as

$$\begin{aligned} \min_{v_+, d_+, \xi} \ & \tfrac{1}{2}\|\varphi^T_{\beta_+}(A)v_+ + d_+e_1\|^2 + \tfrac{c_1}{2}\xi^T\xi \\ s.t. \ & -(\varphi^T_{\beta_+}(B)v_+ + d_+e_2) + \xi = e_2, \end{aligned}$$

and then it can be deduced that

$$\left[\begin{array}{c} v_+ \\ d_+ \end{array}\right] = -(\tfrac{1}{c_1}G_2^TG_2 + H_2^TH_2)^{-1}H_2^Te_2. \quad (21)$$

By solving the problem (18) with the similar way above, we can get $u_- = \varphi_{v_-}(\aleph)\beta_-$ and

$$\left[\begin{array}{c} \beta_- \\ b_- \end{array}\right] = \left(\tfrac{1}{c_2}P_1^TP_1 + Q_1^TQ_1\right)^{-1}Q_1^Te_2, \quad (22)$$

$$\left[\begin{array}{c} v_- \\ d_- \end{array}\right] = \left(\tfrac{1}{c_2}P_2P_2^T + Q_2Q_2^T\right)^{-1}Q_2e_2, \quad (23)$$

where $P_1 = [K_{v_-}(B,\aleph), e_1], Q_1 = [K_{v_-}(A,\aleph), e_2]$, $P_2 = \left[\begin{array}{c} \varphi_{\beta_-}(B) \\ e_1^T \end{array}\right]$ and $Q_2 = \left[\begin{array}{c} \varphi_{\beta_-}(A) \\ e_2^T \end{array}\right]$.

Consequently, (β_+, v_+) and (β_-, v_-) can be obtained by iteratively solving (20)-(21) and (22)-(23), respectively. The specific procedure is as follows.

Algorithm 3. nonlinear LS-TSTM

```
1. Initialization: Let $t = 0, \varepsilon > 0$ small enough and take $v_+^t = v_-^t = [1, \cdots, 1]^T$.
2. Calculate $\beta$ and $b$. Calculate $(\beta_+^t, b_+^t)$ by using (20) and $(\beta_-^t, b_-^t)$ by using (22) with $v_+ = v_+^t$ and $v_- = v_-^t$.
3. Update $v$ and $d$. Calculate $(v_+^{t+1}, d_+^{t+1})$ by using (21) and $(v_-^{t+1}, d_-^{t+1})$ by using (23) with $u_+ = u_+^t$ and $u_- = u_-^t$.
4. Update $\beta$ and $b$. Calculate $(\beta_+^{t+1}, b_+^{t+1})$ by using (20) and $(\beta_-^{t+1}, b_-^{t+1})$ by using (22) with $v_+ = v_+^{t+1}$ and $v_- = v_-^{t+1}$.
5. If $\|\beta_+^{t+1} - \beta_+^t\|, \|\beta_-^{t+1} - \beta_-^t\|, \|v_+^{t+1} - v_+^t\|$ and $\|v_-^{t+1} - v_-^t\|$ are all less than $\varepsilon$ or the maximum number of iterations is achieved, put $\beta_+^* \leftarrow \beta_+^{t+1}, v_+^* \leftarrow v_+^{t+1}$, $\beta_-^* \leftarrow \beta_-^{t+1}, v_-^* \leftarrow v_-^{t+1}$, $b_+^* \leftarrow \frac{b_+^{t+1}+d_+^{t+1}}{2}$ and $b_-^* \leftarrow \frac{b_-^{t+1}+d_-^{t+1}}{2}$; otherwise, set $t \leftarrow t+1$ and return to step 2.
6. Calculate $\|u_i^*(v_i^*)^T\|^2$ by $\|u_i^*(v_i^*)^T\|^2 = (\beta_i^*)^TK_{v_i^*}(\aleph,\aleph)\beta_i^*\|v_i^*\|^2, i = \pm$.
7. Construct decision functions $f_i(X), i = \pm$ by $f_i(X) = [k_{v_i^*}(X_1, X), \cdots, k_{v_i^*}(X_m, X)]\beta_i^* + b_i^*$.
8. For a new input $\widetilde{X}$, its label $y_{\widetilde{X}}$ can be obtained by $y_{\widetilde{X}} = \arg\min_{i=\pm} \frac{|f_i(\widetilde{X})|}{\|u_i^*(v_i^*)^T\|}$.
```

4 Experiments

In this section, in order to demonstrate the classification accuracy and computation time of LLS-TSTM and NLS-TSTM, we perform a series of comparative experiments with linear TSTM and linear STM on ORL and Yale face databases [33-34]. ORL face database contains 400 face images of 40 individuals taken between April 1992 and April 1994 at different times, light and facial expressions. Each individual has 10 face images. Yale face database contains 165 face images of 15 individuals with 11 images for each one. For the convenience of calculation, we chose 11 individuals from ORL database and 7 individuals from Yale database according to different facial details for our experiments, which are described in Tables 1 and 2. We use successive over relaxation (SOR) algorithm [35-36] for solving all QPPs and utilize 5-fold cross-validation method in all experiments. All the experiments are implemented in MATLAB (R2012b) running on a PC with system configuration Intel(R) Core(TM) i3 (2.53GHz) with 2GB of RAM.

Table 1: 11 individuals chosen from ORL database

Facial detail	1	2	3	4	5	6	7	8	9	10	11
pic											
Bald	no	no	no	no	no	yes	yes	no	no	no	yes
Mustache	no	no	no	no	light	no	light	heavy	no	heavy	heavy
FaceSize	thin	mid	mid	mid	thin	mid	thin	mid	mid	fat	mid
Age	young	mid	young	young	young	old	young	mid	young	mid	mid
Glass	no	no	yes	no	no	yes	no	yes	yes	no	yes

Table 2: 7 individuals chosen from Yale database

Facial detail	1	2	3	4	5	6	7
pic							
Mustache	no	light	heavy	light	light	no	light
FaceSize	fat	mid	mid	mid	mid	fat	thin
Age	old	mid	young	young	mid	old	young
Glass	no	no	no	yes	no	no	yes

4.1 Experiments on ORL database

In this subsection, we perform the comparative experiments on 8 pairs composed of 11 individuals taken from ORL database. In each pair, we select 1, 5 and 9 images from each individual as training samples and the others as testing samples. In all experiments, take $\varepsilon = 10^{-3}, c_1 = c_2 = 0.15$ and each image is cropped into 28×23 pixels. In NLS-TSTM, Gaussian RBF kernel $k(x, y) = exp\{-\frac{\|x-y\|^2}{\sigma^2}\}$ is used and grid search approach from 10^3 to 10^5 is employed for selecting the optimal kernel parameter σ. Experiment results are shown in Tables 3-4, in which 'acc', 'time' and 'train' represent classification accuracy (%), calculation time (second) and the number of training samples, respectively. The experiment results of three linear classifiers are showed in Table 3 and The results of LLS- and NLS-TSTM are showed in Table 4.

Table 3: Experiment results of three linear classifiers on ORL database

Pairs	Classifiers	acc(%) time(s)	acc(%) time(s)	acc(%) time(s)
		train=1	train=5	train=9
(1,3)	LLS-TSTM	98.50±0.06	**100.00±0.00**	**100.00±0.00**
		0.80	**0.96**	**1.26**
	Linear TSTM	**99.00±0.04**	97.50±0.07	90.50±0.53
		1.65	3.32	3.09
	Linear STM	97.50±0.29	99.50±0.03	100.00±0.00
		1.33	9.60	4.83
(1,4)	LLS-TSTM	**92.50±0.35**	**99.50±0.03**	**100.00±0.00**
		0.80	**1.40**	**1.22**
	Linear TSTM	85.00±0.89	98.50±0.11	92.50±0.13
		1.23	4.19	5.59
	Linear STM	86.00±0.82	97.50±0.18	99.50±0.03
		2.09	6.39	12.43
(2,10)	LLS-TSTM	**95.50±0.75**	99.00±0.04	**100.00±0.00**
		1.34	**0.98**	**1.24**
	Linear TSTM	94.00±0.93	99.00±0.10	99.50±0.03
		1.13	2.99	3.73
	Linear STM	88.00±3.23	**100.00±0.00**	100.00±0.00
		2.11	5.04	10.68
(3,9)	LLS-TSTM	**93.50±0.23**	**100.00±0.00**	**100.00±0.00**
		0.80	**0.97**	2.10
	Linear TSTM	86.50±1.17	81.50±1.00	75.50±0.36
		1.12	3.18	**1.20**
	Linear STM	87.50±0.35	99.50±0.03	100.00±0.00
		1.13	5.65	12.71
(5,7)	LLS-TSTM	95.50±0.47	**100.00±0.00**	**100.00±0.00**
		0.82	**0.97**	**1.12**
	Linear TSTM	**97.00±0.07**	97.50±0.24	94.00±0.38
		1.32	3.21	4.11
	Linear STM	89.50±1.80	100.00±0.00	100.00±0.00
		1.44	4.00	8.68
(6,7)	LLS-TSTM	91.50±0.11	**99.00±0.04**	**100.00±0.00**
		0.81	**1.95**	**1.14**
	Linear TSTM	**92.00±0.18**	93.00±0.23	82.50±1.01
		0.85	4.67	4.53
	Linear STM	79.50±1.53	99.00±0.10	100.00±0.00
		1.43	12.77	7.73
(6,11)	LLS-TSTM	**94.50±0.64**	**100.00±0.00**	**100.00±0.00**
		1.00	**1.47**	**1.92**
	Linear TSTM	92.00±0.51	97.00±0.40	83.50±1.84
		1.10	3.72	3.35
	Linear STM	82.50±1.07	100.00±0.00	100.00±0.00
		1.29	6.77	10.79
(7,8)	LLS-TSTM	**90.00±0.56**	**100.00±0.00**	**100.00±0.00**
		0.80	**0.98**	**1.14**
	Linear TSTM	87.00±0.34	90.00±0.56	86.50±1.06
		1.24	2.24	4.20
	Linear STM	86.00±1.66	94.50±0.30	99.50±0.03
		0.98	6.09	13.29

From Tables 3-4, we can see that the computation time of LLS-TSTM is obviously less than that of linear TSTM and linear STM. For 5 or 9 training samples case, the computation time of LLS-TSTM is almost faster 2-3 times than that of linear TSTM and 4-8 times than that of linear STM except the pair (3,9) with 9 training samples, and the classification accuracy of LLS-TSTM is better than that of linear TSTM and almost the same with that of linear STM. For 1 training sample case, the classification accuracies of three linear classifiers are the competition each other. In addition, the classification accuracy of NLS-TSTM is higher than that of three linear classifiers in general. Especially, the accuracies of NLS-TSTM are all 100% on 8 binary classification problems in the case of 9 training samples.

Table 4: Experiment results of LLS- and NLS-TSTM on ORL database

Pairs	Classifiers	acc(%)	acc(%)	acc(%)
		train=1	train=5	train=9
(1,3)	NLS-TSTM	**100.00±0.00**	**100.00±0.00**	**100.00±0.00**
	LLS-TSTM	98.50±0.06	100.00±0.00	100.00±0.00
(1,4)	NLS-TSTM	**99.50±0.03**	**100.00±0.00**	**100.00±0.00**
	LLS-TSTM	92.50±0.35	99.50±0.03	100.00±0.00
(2,10)	NLS-TSTM	**97.50±0.18**	**100.00±0.00**	**100.00±0.00**
	LLS-TSTM	95.50±0.75	99.00±0.04	100.00±0.00
(3,9)	NLS-TSTM	**94.00±0.27**	**100.00±0.00**	**100.00±0.00**
	LLS-TSTM	93.50±0.23	100.00±0.00	100.00±0.00
(5,7)	NLS-TSTM	**96.50±0.89**	**100.00±0.00**	**100.00±0.00**
	LLS-TSTM	95.50±0.47	100.00±0.00	100.00±0.00
(6,7)	NLS-TSTM	**93.50±0.34**	95.00±0.28	**100.00±0.00**
	LLS-TSTM	91.50±0.11	**99.00±0.04**	100.00±0.00
(6,11)	NLS-TSTM	**99.00±0.04**	**100.00±0.00**	**100.00±0.00**
	LLS-TSTM	94.50±0.64	100.00±0.00	100.00±0.00
(7,8)	NLS-TSTM	**92.50±1.13**	94.50±0.19	**100.00±0.00**
	LLS-TSTM	90.00±0.56	**100.00±0.00**	100.00±0.00

4.2 Experiments on Yale database

In this subsection, we perform the comparative experiments on 6 pairs composed of 7 individuals taken from Yale database. In each pair, we select 1, 5 or 9 images from each individual as training samples and the others as testing samples. In all experiments, take $\varepsilon = 10^{-3}, c_1 = c_2 = 0.5$ and each image is cropped into 25×20 pixels. In NLS-TSTM, Gaussian RBF kernel $k(x, y) = exp\{-\frac{\|x-y\|^2}{\sigma^2}\}$ is used and grid search approach from 2^{-8} to 2^8 is employed for selecting the optimal kernel parameter σ. Experiment results are shown in Tables 5-6. The experiment results of three linear classifiers are showed in Table 5 and the results of LLS- and NLS-TSTM are showed in Table 6.

From Tables 5-6, we can see that the computation time of LLS-TSTM is apparently less than that of linear TSTM and linear STM except the pair (1,6). For 5 or 9 training samples case, the classification accuracies of both LLS-TSTM and NLS-TSTM are almost the same, and they are higher than that of linear TSTM and linear STM. For 1 training sample case, the classification accuracies of three linear classifiers are the competition each other, and the classification accuracy of NLS-TSTM is higher than that of LLS-

Table 5: Experiment results of three linear classifiers on Yale database

Pairs	Classifiers	acc(%) time(s)	acc(%) time(s)	acc(%) time(s)
		train=1	train=5	train=9
(1,6)	LLS-TSTM	**84.50±3.19**	**97.00±0.23**	**99.00±0.10**
		0.60	1.77	**2.23**
	Linear TSTM	80.50±2.08	78.00±1.68	80.50±2.03
		0.42	**1.71**	3.07
	Linear STM	78.00±1.84	90.00±0.61	98.50±0.11
		1.65	2.95	7.36
(2,5)	LLS-TSTM	**90.00±3.17**	**96.50±0.17**	**98.50±0.06**
		0.47	**1.97**	**2.27**
	Linear TSTM	86.00±3.38	85.00±1.56	80.00±4.83
		0.68	3.38	4.18
	Linear STM	67.00±3.40	94.50±0.25	98.50±0.06
		2.65	2.91	4.80
(2,7)	LLS-TSTM	**90.00±4.44**	**100.00±0.00**	**100.00±0.00**
		0.47	**0.83**	**0.88**
	Linear TSTM	80.50±5.41	71.50±3.39	67.50±2.79
		0.95	2.77	1.70
	Linear STM	84.50±3.53	99.50±0.03	100.00±0.00
		1.05	3.08	6.76
(3,7)	LLS-TSTM	**79.50±5.97**	**100.00±0.00**	**100.00±0.00**
		0.46	**2.05**	**2.45**
	Linear TSTM	71.00±4.32	77.00±5.07	93.00±1.96
		0.77	3.83	4.06
	Linear STM	79.00±3.38	98.00±0.07	98.50±0.06
		1.38	3.09	3.17
(4,7)	LLS-TSTM	70.50±3.97	**99.00±0.04**	**100.00±0.00**
		0.47	**1.16**	**1.21**
	Linear TSTM	**83.50±4.17**	70.50±4.58	67.50±2.18
		0.69	2.70	2.79
	Linear STM	69.50±2.80	94.50±0.08	97.50±0.13
		0.84	3.58	5.60
(5,7)	LLS-TSTM	87.00±4.51	**100.00±0.00**	**100.00±0.00**
		0.49	**1.02**	**2.56**
	Linear TSTM	**89.00±4.32**	68.50±3.06	55.50±0.47
		1.07	5.40	3.58
	Linear STM	82.50±3.63	94.00±0.10	98.50±0.06
		1.29	3.90	5.04

Table 6: Experiment results of LLS- and NLS-TSTM on Yale database

Pairs	Classifiers	acc(%)	acc(%)	acc(%)
		train=1	train=5	train=9
(1,6)	NLS-TSTM	**95.00±0.28**	**98.50±0.11**	**100.00±0.00**
	LLS-TSTM	84.50±3.19	97.00±0.23	99.00±0.10
(2,5)	NLS-TSTM	86.00±4.66	**98.50±0.11**	**100.00±0.00**
	LLS-TSTM	**90.00±3.17**	96.50±0.17	98.50±0.06
(2,7)	NLS-TSTM	**98.00±0.18**	**100.00±0.00**	**100.00±0.00**
	LLS-TSTM	90.00±4.44	100.00±0.00	100.00±0.00
(3,7)	NLS-TSTM	**82.50±2.85**	99.50±0.03	**100.00±0.00**
	LLS-TSTM	79.50±5.97	**100.00±0.00**	100.00±0.00
(4,7)	NLS-TSTM	**91.00±1.43**	**99.50±0.03**	**100.00±0.00**
	LLS-TSTM	70.50±3.97	99.00±0.04	100.00±0.00
(5,7)	NLS-TSTM	**99.00±0.04**	**100.00±0.00**	**100.00±0.00**
	LLS-TSTM	87.00±4.51	100.00±0.00	100.00±0.00

TSTM in general.

5 Conclusion

In this paper, in order to protect the structure information of image data, we first propose a tensor version extension LLS-TSTM for linear LS-TSVM to deal with image data classification. In LLS-TSTM, we seek a pair of nonparallel hyperplanes by solving 4 equalities rather than solving two dual QPPs, which reduces greatly the computation time of LLS-TSTM and at the same time protect the classification accuracy. It notes that up to now nonlinear classification methods based on STM are not seen more. In order to consider the nonlinear extension of LLS-TSTM, we first introduce a new matrix kernel function and then suggest a NLS-TSTM algorithm with detailed theoretical derivation by means of the defined matrix kernel function. In addition, the successive overrelaxation technique is used to solve QPPs in all algorithms.

In order to check the effectiveness of LLS- and NLS-TSTM, we perform comparative experiments with linear TSTM and linear STM on 14 group binary classification problems taken from ORL and Yale face databases. From the experiment results in Tables 3-6, we can conclude that for classification accuracy, NLS-TSTM is more effective than three linear classifiers and LLS-TSTM is more effective than linear TSTM and linear STM in general. The computation time of LLS-TSTM is the fastest. In addition, along with the increase of the number of training samples, the classification accuracies of LLS- and NLS-TSTM are increased obviously.

At present, there are a lot of work to do in this research field, such as, definition method of matrix kernel function, generalization and improvement of algorithms, optimization selection of parameters in modelings and so on.

Acknowledgements: The research was supported by Science and Technology Project of Higher Education of Shandong Province, P. R. China (grant No. J13LI10).

References:

[1] V. N. Vapnik, *Statistical learning theory.* Springer–New York 1998

[2] Jayadeva, R. Khemchandani, S. Chandra, Twin support vector machines for pattern classification, *IEEE Transactions on Pattern Analysis and Machine Intelligence.* 29, 2007, pp. 905–910.

[3] J. A. K. Suykens, L. Lukas, P. V. Dooren, et al., Least squares support vector machine classifiers:

A large scale algorithm, *In Proceedings of European conference circuit theory design.* 1999, pp. 839–842.

[4] J. A. K. Suykens, J. Vandewalle, Least squares support vector machine classifiers, *Neural Processing Letters.* 9, 1999, pp. 293–300.

[5] M. A. Kumar, M. Gopal, Least squares twin support vector machines for pattern classification, *Expert Systems with Applications.* 36, 2009, pp. 7535–7543.

[6] Y. H. Shao, C. H. Zhang, X. B. Wang, N. Y. Deng, Improvements on twin support vector machines, *IEEE Transactions on Neural Networks.* 22, 2011, pp. 962–968.

[7] S. Balasundaram, Kapil, Application of lagrangian twin support vector machines for classification, *2010 Second International Conference on Machine Learning and Computing.* 2010, pp. 193–197.

[8] Z. Q. Qi, Y. J. Tian, Y. Shi, Structural twin support vector machine for classification, *Knowledge-Based Systems.* 43, 2013, pp. 74-81.

[9] X. J. Pen, Least squares twin support vector hypersphere(LS-TSVH) for pattern recognition, *Expert Systems with Applications.* 37, 2010, pp. 8371–8378.

[10] S. Ghorai, A. Mukherjee, P. K. Dutta, Nonparallel plane proximal classifier, *Signal Processing.* 89, 2009, pp. 510–522.

[11] M. A. Kumar, M. Gopal, Application of smoothing technique on twin support vector machines, *Pattern Recognition Letters.* 29, 2008, pp. 1842–1848.

[12] O. L. Mangasarian, E. W. Wild, Multisurface proximal support vector classification via generalized eigenvalues, *IEEE Transactions on Pattern Analysis and Machine Intelligence.* 28, 2006, pp. 69–74.

[13] Y. H. Shao, N. Y. Deng, A coordinate descent margin based-twin support vector machine for classification, *Neural Networks.* 25, 2012, pp. 114–121.

[14] X. Peng, Twin support vector hypersphere (TSVH) classifier for pattern recognition, *Neural Computing and Applications.* 24(5), 2014, pp. 1207–1220.

[15] Z. Qi, Y. Tian, S. Yong, Robust twin support vector machine for pattern classification, *Pattern Recognition.* 46, 2013, pp. 305–316.

[16] Z. Qi, Y. Tian, S. Yong, Laplacian twin support vector machine for semi–supervised classification, *Neural Networks.* 35, 2012 pp. 46–53.

[17] Y. H. Shao, Z. Wang, W. J. Chen, N. Y. Deng, A regularization for the projection twin support vector machine, *Knowledge-Based Systems.* 37, 2013, pp. 203–210.

[18] J. Yang, L. Y. Fan, Weighted generalized kernel discriminant analysis using fuzzy memberships, *WSEAS Transactions on Mathematics.* 10(10), 2011, pp. 346–357.

[19] J. Q. Gao, L. Y. Fan, L. Xu, Solving the face recognition problem using QR factorization, *WSEAS Transactions on Mathematics.* 11(8), 2012, pp. 728–737.

[20] J. Q. Gao, L. Y. Fan, Kernel-based Weighted Discriminant Analysis with QR Decomposition and Its Application to Face Recognition, *WSEAS Transactions on Mathematics.* 10(10), 2011, pp. 358–367.

[21] M. M. Ge, L. Y. Fan, Learning Optimal Kernel for Pattern Classification, *WSEAS Transactions on Mathematics.* 12(5), 2013, pp. 491–500.

[22] D. Tao, X. Li, X. wu, et al., Supervised tensor learning, *Knowledge and Information Systems.* 13, 2007, pp. 1-42.

[23] D. Cai, X. He, J. R. Wen, J. Han, W. Y. Ma, Support tensor machines for text categorization, *Department of Computer Science.* 2006

[24] C. Hou et al., Multiple rank multi-linear SVM for matrix data classification, *Pattern Recognition.* 47, 2014, pp. 454–469.

[25] R. Khemchandani, A. Karpatne, S. Chandra, Proximal support tensor machines, *International Journal of Machine Learning and Cybernetics.* 4, 2013, pp. 703–712.

[26] X. Zhang, X. Gao, Y. Wang, Twin support tensor machines for MCs detection, *Journal of Electronics (China).* 26, 2009, pp. 318–325.

[27] G. Fung, O.L. Mangasarian, Proximal support vector machine classifiers, *In Proceedings of seventh international conference on knowledge and data discovery.* 2001, pp. 77–86.

[28] D. Tao, X. Li, W. Hu, S.J. Maybank, X. Wu, General tensor discriminant analysis and Gabor features for gait recognition, *IEEE Trans Pattern Anal Mach Intell.* 29, 2007, PP. 1700–1715.

[29] D. Tao, X. Li, W. Hu, S. J. Maybank, X. Wu, Tensor rank one discriminant analysis: a convergent method for discriminative multilinear subspace selection,*Neurocomputing.* 71, 2008, PP. 1866–1882.

[30] D. Tao, M. Song, X. Li, J. Shen, J. Sun, X. Wu, C. Faloutsos, S.J. Maybank, Bayesian tensor approach for 3-D face modeling, *IEEE Trans Circuits Syst Video Technol.* 18, 2008, PP. 1397–1410.

[31] D. Tao, J. Sun, J. Shen, X. Wu, X. Li, S. J. Maybank, C. Faloutsos, Bayesion tensor analysis, *IEEE international joint conference on date of conference neural networks.* 1–8, 2008, PP. 1402–1409.

[32] Z. Hao, L. He, B. Chen, X. Yang, A Linear Support Higher-Order Tensor Machine for Classification, *IEEE Transactions on Image Processing.* 22(7), 2013, pp. 2911–2920.

[33] AT&T Labs Cambridge, The Olivetti and Oracle Research Laboratory database of faces, http://www.cl.cam.ac.uk/research/dtg/attarchive /facedatabase.html, 1994.

[34] http://cvc.yale.edu/projects/yalefaces/yalefaces .html.

[35] Z .Q. Luo, P. Tseng, Error bounds and convergence analysis of feasible descent methods: A general approach, *Ann. Oper. Res.* 46, 1993, pp. 157–178.

[36] O. L. Mangasarian, D. R. Musicant, Successive overrelaxation for support vector machines, *IEEE Trans. Neural Netw.* 10, 1999, pp. 1032–1037.