Multitask Centroid Twin Support Vector Machines

Xijiong Xie, Shiliang Sun[*]

Department of Computer Science and Technology, East China Normal University,
500 Dongchuan Road, Shanghai 200241, P.R. China

Abstract

Twin support vector machines are a recently proposed learning method for binary classification. They learn two hyperplanes rather than one as in conventional support vector machines and often bring performance improvements. However, an inherent shortage of twin support vector machines is that the resultant hyperplanes are very sensitive to outliers in data. In this paper, we propose centroid twin support vector machines to overcome this disadvantage. Furthermore, inspired by the recent success of multitask learning which trains multiple related tasks simultaneously, we also extend them to the multitask learning scenario and propose multitask centroid twin support vector machines. Experimental results demonstrate that our proposed methods are effective.

Key words: Twin support vector machine, Support vector machine, Multitask learning, Kernel method

[*]Corresponding author. Tel.: +86-21-54345186; fax: +86-21-54345119.
Email address: `slsun@cs.ecnu.edu.cn` (Shiliang Sun)

Preprint submitted to Neurocomputing *April 21, 2014*

1. Introduction

Support vector machines (SVMs) have been developed rapidly during recent years [1, 2]. They are a powerful tool for pattern classification and regression. The SVM method outputs a hyperplane that has the largest distance to the nearest training data, whose optimization involves the minimization of a quadratic programming (QP) problem. By the use of the kernel trick, SVMs can learn a nonlinear decision function which is linear in a potentially high-dimensional feature space [3]. SVMs have been applied to a variety of practical problems such as object detection, text categorization, bioinformatics and image classification [4].

Recently, the research of nonparallel hyperplane classifiers has been a new hot spot. Mangasarian and Wild [5] proposed generalized eigenvalue proximal SVMs (GEPSVMs) for binary classification. Instead of finding a single hyperplane as in SVMs, GEPSVMs find two nonparallel hyperplanes such that each hyperplane is as close as possible to examples from one class and as far as possible to examples from the other class. The two hyperplanes are obtained by eigenvectors corresponding to the smallest eigenvalues of two related generalized eigenvalue problems. Particularly when data consist of points that are close to one of two intersecting "cross planes", the performance of GEPSVMs is better than the performance of SVMs [5]. Jayadeva et al. [6] proposed another nonparallel hyperplane classifier called twin SVMs (TSVMs), which aim to generate two nonparallel hyperplanes such that one of the hyperplanes is closer to one class and has a certain distance to the other class. The formulation of TSVMs is different from that of GEPSVMs and is similar to SVMs. TSVMs solve a pair of QP, whereas SVMs solve a single QP. The strategy of solving two smaller sized QP rather than one large QP makes TSVMs work faster than standard SVMs [7]. Experimental results [6] show that nonparallel hyperplane classifiers given by TSVMs can indeed improve the performance of traditional SVMs. Researchers also proposed some improved versions of TSVMs such as TBSVMs [8, 9]. The significant advantage of TBSVM over TSVMs is that the structural risk minimization principle is implemented by introducing the regularization term. Least squares twin support vector machines (LS-TSVM) [10] and least squares twin parametric-margin support vector machines [11] have been proposed, which can lead to simple and fast algorithms for generating binary classifiers by replacing inequality constraints with equality constraints. Recently, some works [12, 13, 14] commonly attempted to use the centroid of the class, such that the examples of one class are closest to its class centroid while the examples of different classes are separated as far as possible. However in this paper, we use the centroid of the class in a more convenient way, such that we can tradeoff two distances, one distance between the obtained hyperplane and its class centroid and another distance between the obtained hyperplane and examples.

In many practical problems, a learning task can involve multiple related tasks. The standard methodology in machine learning is to learn those tasks separately. But solving them together is expected to be more advantageous because the knowledge from some tasks can help to improve the generalization ability of the other tasks [15, 16, 17, 18, 34]. Consequently, multitask learning, whose principal purpose is to improve the overall generalization performance by leveraging the knowledge contained in multiple related tasks [20, 21, 22, 23, 24], has been investigated extensively and emerged as a very promising research direction [25, 26, 27, 28]. One approach to multitask learning, which is also exploited in this paper, assumes that the tasks share a common underlying representation and each task further has its own bias [29, 30]. Past empirical work has shown that this kind of multitask learning mechanism usually improves over its single task counterpart, e.g., the recent multitask SVMs outperform traditional single-task SVMs [24, 31, 32].

In this paper, after reviewing related work in Section 2, we analyze shortcomings of TSVMs and propose centroid TSVMs (CTSVMs) that are based on class centroids in Section 3.1 and kernel CTSVMs in Section 3.2. We then extend CTSVMs to multitask CTSVMs (MCTSVMs) in Section 3.3. MCTSVMs are easy to implement by an appropriate modification of the optimization problem in CTSVMs, which casts MCTSVMs as a constrained optimization problem with a quadratic objective function. Section 4 gives kernel MCTSVMs which combine the kernel trick and MCTSVMs. After reporting experimental results in Section 5, we give conclusions and future work in Section 6.

2. Related work

In this section, we briefly review SVMs, TSVMs, multitask SVMs and multitask TSVMs. They constitute the foundation of our subsequent proposed methods.

2.1. SVMs and TSVMs

SVMs have been introduced in the framework of structural risk minimization and in the theory of VC bounds [1, 2]. Suppose there are m examples represented by $T = \{(x_1, y_1), ..., (x_m, y_m)\}$. Let $y_i \in \{1, -1\}$ denote the class to which the ith example belongs. First we review the linearly separable case. Classifier parameters $w \in R^d$ and $b \in R$ need to satisfy

$$y_i(w^T x_i + b) \geq 1.$$

The hyperplane described by $w^T x + b = 0$ lies midway between the bounding hyperplanes given by $w^T x + b = 1$ and $w^T x + b = -1$. The margin of separation between the two classes is given by $\frac{2}{\|w\|_2}$, where $\|w\|_2$ denotes the L_2 norm of w. Support vectors are those training examples lying on the above two hyperplanes. The standard SVMs are obtained by solving the following problem

$$\begin{aligned} &\min_{w,b} \ \frac{1}{2} w^T w \\ &\text{s.t.} \ \ \forall i : y_i(w^T x_i + b) \geq 1. \end{aligned} \tag{1}$$

The decision function is

$$f(x) = \text{sign}(w^T x + b). \tag{2}$$

When the two classes are not strictly linearly separable, classifier parameters w and b need to satisfy

$$y_i(w^T x_i + b) \geq 1 - \xi_i.$$

The optimization problem of (1) can be modified to

$$\begin{aligned} &\min_{w,b}\ \frac{1}{2}w^Tw + c\sum_{i=1}^{m}\xi_i \\ &\text{s.t.}\ \ \forall i : y_i(w^Tx_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0, \end{aligned} \tag{3}$$

where c is a penalty parameter and ξ_i are the slack variables. The Wolfe dual of (3) can be expressed as

$$\begin{aligned} &\min_{\alpha}\ \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} y_iy_j(x_i \cdot x_j)\alpha_i\alpha_j - \sum_{i=1}^{m}\alpha_i \\ &\text{s.t.}\ \sum_{i=1}^{m} y_i\alpha_i = 0, \\ &\qquad 0 \leq \alpha_i \leq c, i = 1, \cdots, m, \end{aligned} \tag{4}$$

where α_i are Lagrangian multipliers. The optimal solution is

$$w = \sum_{i=1}^{m}\alpha_i^*y_ix_i,\ b = \frac{1}{N_{sv}}(y_j - \sum_{i=1}^{N_{sv}}\alpha_i^*y_i(x_i \cdot x_j)), \tag{5}$$

where α^* is the solution of the dual problem (4), and N_{sv} represents the number of support vectors satisfying $0 < \alpha < c$. The decision function is

$$f(x) = \text{sign}(w^Tx + b). \tag{6}$$

Then we introduce TSVMs. Suppose examples belonging to classes 1 and -1 are represented by matrices A_+ and B_-, and the size of A_+ and B_- are $(m_1 \times d)$ and $(m_2 \times d)$, respectively. We define two matrices A, B and four vectors v_1, v_2, e_1, e_2, where e_1 and e_2 are vectors of ones of appropriate dimensions and

$$A = (A_+, e_1),\ B = (B_-, e_2),\ v_1 = \begin{pmatrix} w_1 \\ b_1 \end{pmatrix},\ v_2 = \begin{pmatrix} w_2 \\ b_2 \end{pmatrix}.$$

TSVMs obtain two nonparallel hyperplanes

$$w_1^T x + b_1 = 0 \quad \text{and} \quad w_2^T x + b_2 = 0 \tag{7}$$

around which the examples of the corresponding class get clustered. The classifier is given by solving the following QP separately

(TSVM1)

$$\begin{aligned} \min_{v_1,q_1} \ & \frac{1}{2}(Av_1)^T(Av_1) + c_1 e_2^T q_1 \\ \text{s.t.} \ & -(Bv_1) + q_1 \geq e_2, \ q_1 \geq 0, \end{aligned} \tag{8}$$

(TSVM2)

$$\begin{aligned} \min_{v_2,q_2} \ & \frac{1}{2}(Bv_2)^T(Bv_2) + c_2 e_1^T q_2 \\ \text{s.t.} \ & (Av_2) + q_2 \geq e_1, \ q_2 \geq 0, \end{aligned} \tag{9}$$

where c_1, c_2 are nonnegative parameters and q_1, q_2 are slack vectors of appropriate dimensions. The label of a new example x is determined by the minimum of $|x^T w_r + b_r|$ $(r = 1, 2)$ which are the perpendicular distances of x to the two hyperplanes given in (7).

2.2. Multitask SVMs

Suppose there are T related learning tasks and all data come from the same space $R^d \times \{-1, 1\}$. The input-output pair (x_{it}, y_{it}) $(i \in \{1, 2, \cdots, m\}, t \in \{1, 2, \cdots, T\})$ stands for the ith example of the tth task's training data. Regularized multitask learning learns T classifiers $w_1, \cdots, w_T$. All w_t can be written as $w_t = w_0 + v_t$, where w_0 is described as the common vector and v_t is described as the own bias vector of each hyperplane. The vectors v_t

are “small” when the tasks are similar to one another. Multitask SVMs [23] solve the following optimization problem

$$\begin{aligned} \min_{w_0, v_t, \xi_{it}} \quad & \sum_{t=1}^{T}\sum_{i=1}^{m} \xi_{it} + \frac{\lambda_1}{T}\sum_{t=1}^{T} ||v_t||_2^2 + \lambda_2 ||w_0||_2^2 \\ \text{s.t.} \quad & \forall t, i: \; y_{it}(w_0 + v_t)^T x_{it} \geq 1 - \xi_{it}, \\ & \qquad \xi_{it} \geq 0, \end{aligned} \tag{10}$$

where λ_1 and λ_2 are nonnegative parameters controlling the tradeoff among tasks and ξ_{it} are slack variables. The dual optimization problem tends out to be

$$\begin{aligned} \max_{\alpha_{it}} \; & \sum_{i=1}^{m}\sum_{t=1}^{T} \alpha_{it} - \frac{1}{2}\sum_{i=1}^{m}\sum_{s=1}^{T}\sum_{j=1}^{m}\sum_{t=1}^{T} \alpha_{is} y_{is} \alpha_{jt} y_{jt} G_{st}(x_{is}, x_{jt}) \\ \text{s.t.} \; & 0 \leq \alpha_{it} \leq c, \end{aligned} \tag{11}$$

where

$$\begin{aligned} & G_{st}(x_{is}, x_{jt}) = (\frac{1}{u} + \delta_{st}) K_{st}(x_{is}, x_{jt}), \\ & u = \frac{T\lambda_2}{\lambda_1}, \; c = \frac{T}{2\lambda_1}, \end{aligned} \tag{12}$$

and δ_{st} is the Kronecker delta kernel

$$\delta_{st} = \begin{cases} 1 & \text{if } s = t, \\ 0 & \text{if } s \neq t. \end{cases} \tag{13}$$

The decision function for each task is given by

$$f_t(x) = \text{sign}(\sum_{i=1}^{m}\sum_{s=1}^{T} \alpha_{is} G_{st}(x_{is}, x)). \tag{14}$$

2.3. Multitask TSVMs

In [33], we have extended TSVMs to multitask learning and call the resultant method direct multitask TSVMs (DMTSVMs). Suppose there are a total of T tasks which are assumed to be related. Here examples of class 1 from the tth task are represented by $\tilde{A}_t$ and examples of class -1 from this task are represented by $\tilde{B}_t$. Examples of class 1 from all tasks are collectively represented by $\tilde{A}$ and examples of class -1 from all tasks are represented by $\tilde{B}$. For simplicity, suppose e is a vector of ones of appropriate dimensions

$$A_t = (\tilde{A}_t, e),\ B_t = (\tilde{B}_t, e),\ A = (\tilde{A}, e),\ B = (\tilde{B}, e).$$

We consider that all tasks have two common vectors $v = \begin{pmatrix} w_1 \\ b_1 \end{pmatrix}$, $u = \begin{pmatrix} w_2 \\ b_2 \end{pmatrix}$ corresponding to two hyperplanes. Suppose that v_t, u_t mean the deviation between task t and common vectors. The classifier parameter of class 1 of the tth task is $(v+v_t)$, where $(v+v_t) = \begin{pmatrix} w_{1t} \\ b_{1t} \end{pmatrix}$. The classifier parameter of class -1 of the tth task is $(u+u_t)$, where $(u+u_t) = \begin{pmatrix} w_{2t} \\ b_{2t} \end{pmatrix}$. The optimization problems can be written as

$$\begin{aligned} \min_{v_t, v, q_{1t}}\ & \frac{1}{2}\sum_{t=1}^{T} \rho_t \|A_t v_t\|_2^2 + \frac{1}{2}\|Av\|_2^2 + c_1 \sum_{t=1}^{T} e_{1t}^T q_{1t} \\ \text{s.t.}\ \ & \forall t:\ -B_t(v+v_t) + q_{1t} \geq e_{1t},\ q_{1t} \geq 0, \end{aligned} \tag{15}$$

$$\begin{aligned} \min_{u_t, u, q_{2t}}\ & \frac{1}{2}\sum_{t=1}^{T} \lambda_t \|B_t u_t\|_2^2 + \frac{1}{2}\|Bu\|_2^2 + c_2 \sum_{t=1}^{T} e_{2t}^T q_{2t} \\ \text{s.t.}\ \ & \forall t:\ A_t(u+u_t) + q_{2t} \geq e_{2t},\ q_{2t} \geq 0, \end{aligned} \tag{16}$$

where c_1, c_2, ρ_t, λ_t are nonnegative parameters and e_{1t}, e_{2t} are vectors of ones of appropriate dimensions. If $\rho_t \gg 0$ and $\lambda_t \gg 0$, it will tend to make the models to be the same model. If $\rho_t \to 0$ and $\lambda_t \to 0$, it will tend to make all the tasks unrelated.

3. Our proposed methods

3.1. Centroid twin support vector machines

In this section, we present a method to improve TSVMs. The optimization problem of TSVMs is to minimize the sum of squared distances from the hyperplane to examples of one class and a regularized term. In normal situations, the obtained optimal hyperplanes are usually close to the respective class centroids and examples and give good performance. If there exist outliers which are far from the respective class, the obtained hyperplanes will deviate far from the ideal locations and lead to poor performance. In order to eliminate this defect in TSVMs, we propose CTSVMs to weight the distances from class centroids to hyperplanes.

Now we formally introduce the optimization problem of CTSVMs. Suppose examples of class 1 are represented by $\tilde{A}$ and examples of class -1 are represented by $\tilde{B}$. The centroid of class 1 is defined as s and the centroid of class -1 is defined as h. We define two matrices E, F and four vectors u, v, e_1, e_2, where $u = z_1 s$, $v = z_2 h$ (z_1 and z_2 are nonnegative parameters to be adjusted), e_1 and e_2 are vectors of ones of appropriate dimensions and

$$A = \begin{pmatrix} \tilde{A} \\ u^T \end{pmatrix}, \ B = \begin{pmatrix} \tilde{B} \\ v^T \end{pmatrix}, \ H = (A, e_1), \ G = (B, e_2).$$

CTSVMs obtain two nonparallel hyperplanes

$$w_1^T x + b_1 = 0 \quad \text{and} \quad w_2^T x + b_2 = 0 \tag{17}$$

around which the examples of the corresponding class get clustered. The optimization problems can be written as
(CTSVM1)

$$\begin{aligned} &\min_{v_1, q_1} \ \frac{1}{2}(Hv_1)^T (Hv_1) + c_1 e_2^T q_1 \\ &\text{s.t.} \ -(Gv_1) + q_1 \geq e_2, \ q_1 \geq 0, \end{aligned} \tag{18}$$

(CTSVM2)

$$\begin{aligned} &\min_{v_2, q_2} \ \frac{1}{2}(Gv_2)^T (Gv_2) + c_2 e_1^T q_2 \\ &\text{s.t.} \ (Hv_2) + q_2 \geq e_1, \ q_2 \geq 0, \end{aligned} \tag{19}$$

where c_1, c_2 are parameters and q_1, q_2 are slack vectors of appropriate dimensions.

The Lagrangian of the problem CTSVM1 is given by

$$L(v_1, q_1, \alpha, \beta) = \frac{1}{2}(Hv_1)^T (Hv_1) + c_1 e_2^T q_1 - \alpha^T (-Gv_1 + q_1 - e_2) - \beta^T q_1, \tag{20}$$

where $\alpha = (\alpha_1, \alpha_2 \cdots, \alpha_{m_2+1})^T$, $\beta = (\beta_1, \beta_2 \cdots, \beta_{m_2+1})^T$ are the vectors of Lagrange multipliers. The Karush-Kuhn-Tucker (KKT) optimality conditions for (CTSVM1) are given by

$$H^T H v_1 + G^T \alpha = 0, \tag{21}$$

$$c_1 e_2 - \alpha - \beta = 0, \tag{22}$$

$$-G v_1 + q_1 \geq e_2, q_1 \geq 0, \tag{23}$$

$$\alpha^T(-G v_1 + q_1 - e_2) = 0,\ \beta^T q_1 = 0, \tag{24}$$

$$\alpha \geq 0,\ \beta \geq 0. \tag{25}$$

Since $\beta \geq 0$, from (22), we have $0 \leq \alpha \leq c_1$. From (21), v_1 can be given by

$$v_1 = -(H^T H)^{-1} G^T \alpha. \tag{26}$$

To avid ill-conditioning of $H^T H$, we use a regularization term ϵI, where $\epsilon > 0$, I is an identity matrix of appropriate dimensions. Therefore, (26) is modified to

$$v_1 = -(H^T H + \epsilon I)^{-1} G^T \alpha. \tag{27}$$

Using (20), (26) and the KKT conditions, the Wolfe dual is

$$\begin{aligned} \max_{\alpha} \quad & e_2^T \alpha - \frac{1}{2} \alpha^T G (H^T H)^{-1} G^T \alpha \\ \text{s.t.} \quad & 0 \leq \alpha \leq c_1. \end{aligned} \tag{28}$$

Similarly, we consider CTSVM2 and obtain its dual as

$$\begin{aligned} \max_{\gamma} \quad & e_1^T \gamma - \frac{1}{2} \gamma^T H (G^T G)^{-1} H^T \gamma \\ \text{s.t.} \quad & 0 \leq \gamma \leq c_2. \end{aligned} \tag{29}$$

The augmented vector v_2 is given by

$$v_2 = (G^T G)^{-1} H^T \gamma. \tag{30}$$

The label of a new example x is determined by the minimum of $|x^T w_r + b_r|$ $(r = 1, 2)$ which are the perpendicular distances of x to the two hyperplanes given in (17).

3.2. Kernel centroid twin support vector machines

We also extend CTSVMs to kernel CTSVMs. We deal with the examples and define

$$E = (K\{A, C^T\}, e), F = (K\{B, C^T\}, e),$$

where for example, $K\{A, C^T\}$ is the kernel matrix defined by $K\{x_i, x_j\} = (\Phi(x_i), \Phi(x_j))$ with x_i being the ith row of A and x_j being the jth column of C^T. $\Phi(\cdot)$ is a nonlinear mapping from a low-dimensional feature space to a high-dimensional feature space and C denotes all training examples, that is, $C = (A^T, B^T)^T$. The optimization problems can be written as

$$\begin{aligned} &\min_{v_1, q_1} \ \frac{1}{2}(Ev_1)^T(Ev_1) + c_1 e_2^T q_1 \\ &\text{s.t.} \ -(Fv_1) + q_1 \geq e_2, \ q_1 \geq 0, \end{aligned} \tag{31}$$

$$\begin{aligned} &\min_{v_2, q_2} \ \frac{1}{2}(Fv_2)^T(Fv_2) + c_2 e_1^T q_2 \\ &\text{s.t.} \ (Ev_2) + q_2 \geq e_1, \ q_2 \geq 0, \end{aligned} \tag{32}$$

where c_1 , c_2 are parameters and q_1, q_2 are slack vectors of appropriate dimensions. Then we can get the classifier parameters from the above derivation. The label of a new example x is determined by the minimum of $|K\{x, C^T\}w_r + b_r|$ $(r = 1, 2)$.

3.3. Multitask centroid twin support vector machines

In this part, we extend CTSVMs to multitask learning. The centroid of class 1 of all tasks is defined as g_1 and the centroid of class 1 of the tth task is defined as g_{1t}. The centroid of class -1 of all tasks is defined as f_1 and the centroid of class -1 of the tth task is defined as f_{1t}. We define four vectors g, f, g_t, f_t, where $g = p_1 g_1$, $g_t = p_{1t} g_{1t}$, $f = q_1 f_1$ and $f_t = q_{1t} f_{1t}$ (p_1, p_{1t}, q_1 and q_{1t} are nonnegative parameters which need to be adjusted). We use matrices E, F, E_t, F_t as

$$A = \begin{pmatrix} \tilde{A} \\ g^T \end{pmatrix}, \ B = \begin{pmatrix} \tilde{B} \\ f^T \end{pmatrix}, \ A_t = \begin{pmatrix} \tilde{A}_t \\ g_t^T \end{pmatrix}, \ B_t = \begin{pmatrix} \tilde{B}_t \\ f_t^T \end{pmatrix},$$

$$E = (A, e), \ F = (B, e), \ E_t = (A_t, e), \ F_t = (B_t, e).$$

The optimization problems can be written as

$$\begin{aligned} \min_{v_t, v, q_{1t}} \ & \frac{1}{2} \sum_{t=1}^{T} \rho_t \|E_t v_t\|_2^2 + \frac{1}{2} \|Ev\|_2^2 + c_1 \sum_{t=1}^{T} e_{1t}^T q_{1t} \\ \text{s.t.} \ & \forall t: -F_t(v + v_t) + q_{1t} \geq e_{1t}, \ q_{1t} \geq 0, \end{aligned} \tag{33}$$

$$\begin{aligned} \min_{u_t, u, q_{2t}} \ & \frac{1}{2} \sum_{t=1}^{T} \lambda_t \|F_t u_t\|_2^2 + \frac{1}{2} \|Fu\|_2^2 + c_2 \sum_{t=1}^{T} e_{2t}^T q_{2t} \\ \text{s.t.} \ & \forall t: E_t(u + u_t) + q_{2t} \geq e_{2t}, \ q_{2t} \geq 0, \end{aligned} \tag{34}$$

where ρ_t, λ_t, c_1, c_2 are nonnegative parameters and e_{1t}, e_{2t}, e are vectors of ones of appropriate dimensions. The Lagrangian of (33) is given by

$$\begin{aligned} L(v, v_t, q_{1t}, \alpha_t, \beta_t) = & \frac{1}{2} \sum_{t=1}^{T} \rho_t \|E_t v_t\|_2^2 + \frac{1}{2} \|Ev\|_2^2 + c_1 \sum_{t=1}^{T} e_{1t}^T q_{1t} \\ & - \sum_{t=1}^{T} \alpha_t^T [-F_t(v + v_t) + q_{1t} - e_{1t}] - \sum_{t=1}^{T} \beta_t^T q_{1t}, \end{aligned} \tag{35}$$

where α_t and β_t are the vectors of Lagrange multipliers. We take partial derivatives of the above equation and let them be zero

$$\frac{\partial L}{\partial v} = E^T E v + \sum_{i=1}^{T} F_t^T \alpha_t = 0, \tag{36}$$

$$\frac{\partial L}{\partial v_t} = \rho_t E_t^T E_t v_t + F_t^T \alpha_t = 0, \tag{37}$$

$$\frac{\partial L}{\partial q_{1t}} = c_1 e_{1t} - \alpha_t - \beta_t = 0. \tag{38}$$

From the above equations, we obtain

$$v = -(E^T E)^{-1} \sum_{t=1}^{T} F_t^T \alpha_t, \tag{39}$$

$$v_t = -\frac{1}{\rho_t}(E_t^T E_t)^{-1} F_t^T \alpha_t. \tag{40}$$

We substitute (39), (40) into (35) using $\alpha = (\alpha_1^T, \alpha_2^T, \cdots, \alpha_t^T)^T$, $M = (E_1^T, \cdots, E_t^T)^T$, $N = (F_1^T, \cdots, F_t^T)^T$ and get

$$\begin{aligned} L(v, v_t, \alpha, \rho_t) &= \frac{1}{2}\sum_{t=1}^{T} \rho_t v_t^T E_t^T E_t v_t + \frac{1}{2} v^T E^T E v + \sum_{t=1}^{T} \alpha_t^T e_{1t} + \sum_{t=1}^{T} \alpha_t^T F_t (v + v_t) \\ &= \sum_{t=1}^{T} \alpha_t^T e_{1t} - \frac{1}{2}\sum_{t=1}^{T}\sum_{s=1}^{T} \alpha_t^T F_t (E^T E)^{-1} F_s^T \alpha_s - \frac{1}{2\rho_t}\sum_{t=1}^{T} \alpha_t^T F_t (E_t^T E_t)^{-1} F_t^T \alpha_t \\ &= \alpha^T e_{1t} - \frac{1}{2}\alpha^T \left[N(E^T E)^{-1} N^T + \begin{pmatrix} \frac{F_1(E_1^T E_1)^{-1} F_1^T}{\rho_1} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & 0 & \frac{F_t(E_t^T E_t)^{-1} F_t^T}{\rho_t} \end{pmatrix} \right] \alpha. \end{aligned} \tag{41}$$

The Wolfe dual is

$$\max_{\alpha} \ \alpha^T e_{1t} - \frac{1}{2}\alpha^T \left[N(E^T E)^{-1} N^T + \begin{pmatrix} \frac{F_1(E_1^T E_1)^{-1} F_1^T}{\rho_1} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & 0 & \frac{F_t(E_t^T E_t)^{-1} F_t^T}{\rho_t} \end{pmatrix} \right] \alpha \tag{42}$$

$$\text{s.t.} \ \ 0 \le \alpha \le c_1.$$

The classifier parameter of class 1 of the tth task can be obtained. Similarly, we can deal with the other QP

$$\begin{aligned} L(u, u_t, \gamma, \lambda_t) &= \frac{1}{2}\sum_{t=1}^{T} \lambda_t u_t^T F_t^T F_t u_t + \frac{1}{2} u^T F^T F u + \sum_{t=1}^{T} \gamma_t^T e_{2t} + \sum_{t=1}^{T} \gamma_t^T E_t(u + u_t) \\ &= \sum_{t=1}^{T} \gamma_t^T e_{2t} - \frac{1}{2}\sum_{t=1}^{T}\sum_{s=1}^{T} \gamma_t^T E_t (F^T F)^{-1} E_s^T \gamma_s - \frac{1}{2\lambda_t}\sum_{t=1}^{T} \gamma_t^T E_t (F_t^T F_t)^{-1} E_t^T \gamma_t \\ &= \gamma^T e_{2t} - \frac{1}{2}\gamma^T \left[M(F^T F)^{-1} M^T + \begin{pmatrix} \frac{E_1(F_1^T F_1)^{-1} E_1^T}{\lambda_1} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & 0 & \frac{E_t(F_t^T F_t)^{-1} E_t^T}{\lambda_t} \end{pmatrix} \right] \gamma, \end{aligned} \tag{43}$$

where γ_t are the vectors of Lagrange multipliers, $\gamma = (\gamma_1^T, \gamma_2^T, \cdots, \gamma_t^T)^T$. The Wolfe dual is described by

$$\max_{\gamma} \ \gamma^T e_{2t} - \frac{1}{2}\gamma^T \left[M(F^T F)^{-1} M^T + \begin{pmatrix} \frac{E_1(F_1^T F_1)^{-1} E_1^T}{\lambda_1} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & 0 & \frac{E_t(F_t^T F_t)^{-1} E_t^T}{\lambda_t} \end{pmatrix} \right] \gamma \tag{44}$$

$$\text{s.t.} \ \ 0 \leq \gamma \leq c_2.$$

Then we can get the classifier parameter of class -1 of the tth task. The label of a new example x of the tth task is determined by the minimum of $|x^T w_{rt} + b_{rt}|$ $(r = 1, 2)$.

Now we compare the time complexities of MSVMs,, DMTSVMs and MCTSVMs. Suppose the number of samples from all tasks is equal to l. MSVMs solve a single QP and has the computational complexity of $O(l^3)$, while DMTSVMs and MCTSVMs solve a pair of QP and have the computational complexity of $O(2 \times (l/2)^3)$ and $O(2 \times ((l + T)/2)^3)$, respectively.

Generally speaking, $l \gg T$. Therefore, MCTSVMs and DMTSVMs are more efficient for multi-task learning in computational complexity.

4. Kernel multitask centroid twin support vector machines

Now we extend MCTSVMs to nonlinear classification via the kernel trick. In some situations, a liner classifier may not be suitable when training sets are not linearly separable. The kernel trick can be used for solving this problem. We deal with the examples and define

$$\begin{aligned}
E &= (K\{A, C^T\}, e), E_t = (K\{A_t, C^T\}, e), \\
F &= (K\{B, C^T\}, e), F_t = (K\{B_t, C^T\}, e),
\end{aligned}$$

where C denotes training examples of all tasks, that is, $C = (A_1^T, B_1^T, A_2^T, \cdots, A_t^T, B_t^T)^T$. The optimization problems can be written as

$$\begin{aligned}
\min_{v_t, v, q_{1t}} \ & \frac{1}{2}\sum_{t=1}^{T} \rho_t \|E_t v_t\|_2^2 + \frac{1}{2}\|Ev\|_2^2 + c_1 \sum_{t=1}^{T} e_{1t}^T q_{1t} \\
\text{s.t.} \ \ & \forall t: -F_t(v + v_t) + q_{1t} \geq e_{1t}, \ q_{1t} \geq 0,
\end{aligned} \tag{45}$$

$$\begin{aligned}
\min_{u_t, u, q_{2t}} \ & \frac{1}{2}\sum_{t=1}^{T} \lambda_t \|F_t u_t\|_2^2 + \frac{1}{2}\|Fu\|_2^2 + c_2 \sum_{t=1}^{T} e_{2t}^T q_{2t} \\
\text{s.t.} \ \ & \forall t: E_t(u + u_t) + q_{2t} \geq e_{2t}, \ q_{2t} \geq 0,
\end{aligned} \tag{46}$$

where ρ_t, λ_t, c_1, c_2 are nonnegative parameters and e_{1t}, e_{2t} are vectors of ones of appropriate dimensions. Then we can get the classifier parameters of every task from the above derivation. The label of a new example x of the tth task is determined by the minimum of $|K\{x, C^T\}w_{rt} + b_{rt}|$ $(r = 1, 2)$.

5. Experimental results

In this section, first, we perform experiments on a toy data which shows CTSVMs are less susceptible to the impact of outliers compared to TSVMs. Then we implement experiments of binary classification problems using real-world datasets Isolet spoken alphabet recognition and Monk taken from the UCI Machine Learning Repository and Landmine detection based on Airborne Radar Data [1]. Details about the three datasets are listed in Table 1.

5.1. Toy data

The datasets are created according to two Gaussian distribution for two classes. For the two classes, the first class has 200 points and the second class has 194 points and six outliers. The means are (2.5, 4.5), (5, 2.5) and covariance matrices are $\sum_1 = \left(\begin{smallmatrix} 0.35 & 0.2 \\ 0.2 & 0.35 \end{smallmatrix}\right)$ and $\sum_2 = \left(\begin{smallmatrix} 0.3 & 0.2 \\ 0.2 & 0.3 \end{smallmatrix}\right)$, respectively. We conduct experiments using TSVMs and CTSVMs on this dataset. We use a grid search strategy to select best parameters (c_1, c_2, z_1, z_2) in the region $[2^{-7}, 2^7]$ with exponential growth 0.5. When z_1 and z_2 are zeros, CTSVMs are equivalent to TSVMs. In Figure 1(a), '*' represents the first class. '+' represents the second class. We choose 194 points as training examples without outliers and 200 points as test examples. The two lines are very close to the respective class centroid and class examples and the test accuracy is 1. In Figure 1(b) and Figure 1(c) where outliers exist, we choose 200 points as training examples and 200 points as test examples. As can be seen from

[1] http://www.ece.duke.edu/∼lcarin/LandmineData.zip

Figure 1, the classifiers obtained by TSVMs on training examples with outliers are largely different from that obtained on training examples, while for CTSVMs, the difference is much less. The test accuracies for Figure 1(b) and Figure 1(c) are 0.87 and 0.94, respectively. From this experiment, we conclude that CTSVMs can improve performance compared to TSVMs.

[Table 1 about here.]

[Figure 1 about here.]

5.2. Speech recognition

The Isolet dataset is collected from 150 subjects speaking each letter of the alphabet twice. Hence, we have 52 training examples from each speaker. Due to the lack of three examples, there are 7797 examples in total. These speakers are grouped into five sets of 30 speakers each. These groups are referred to as isolet1-isolet5. Each of these datasets has 26 classes. We treat each of the subsets as its own classification task. Therefore, there are five tasks that are highly related with each other because they are taken from the same utterances. They are different from each other because they come from different groups that vary largely in the way of speaking the English alphabets. The attribute information include spectral coefficients, contour features, sonorant features, pre-sonorant features and post-sonorant features.

In Isolet dataset, we choose two classes (m, n) from them for classification, since TSVMs are designed for binary classification while Isolet contains 26 classes and (m, n) is hard to discriminate in practical communications.

Then we capture 98% of the data variance while reducing the dimensionality from 617 to 276 with PCA. “1-NN” represents the algorithm of one nearest neighbor. “MSVM” represents multitask SVMs. “MTGP” represents multitask Gaussian process [34]. We use three-fold cross-validation to get the average classification accuracy rates and employ a polynomial kernel function with degree two. The kernel can be written as

$$K(x_i, x_j) = (1 + x_i^T x_j)^2. \tag{47}$$

It is often necessary to choose other best parameters. We use a grid search strategy to select best parameters for all involved methods in the region $[2^{-7}, 2^7]$ with exponential growth 0.5. For example, in MCTSVM, various pairs of $(p_1, p_{1t}, q_1, q_{1t}, c_1, c_2, \rho_t, \lambda_t)$ are considered. From the experimental results in Table 2, we can find that DMTSVM outperforms TSVM. MCTSVM is a little better than DMTSVM and outperforms CTSVM. Although the performance of SVM is a little better than TSVM, the performance of our proposed CTSVM is better than SVM and TSVM. However, MSVM is worse than DMTSVM and MCTSVM. DMTSVM and MCTSVM perform better than the corresponding single task learning methods. MTGP has the worst performance on this dataset.

[Table 2 about here.]

5.3. Landmine detection

The Landmine detection dataset is collected from a real landmine field. There are a total of 19 datasets for which 1-10 are collected at foliated regions and 11-19 are collected at regions that are bare earth or desert. It is collected from various landmine fields by an actual synthetic-aperture radar system. Each example is represented by a 9-dimensional feature vector extracted from radar images and the corresponding binary label (1 for landmine and 0 for clutter).

In Landmine dataset, we choose two tasks from foliated regions. Due to the unbalanced labels in Landmine dataset, for each task, we select 150 examples for which the number of positive examples is almost the same as one of negative examples in our experiments. We use three-fold cross-validation to get the average classification accuracy rates and employ an RBF kernel. The kernel can be written as

$$K(x_i, x_j) = exp(-\frac{\|x_i - x_j\|_2^2}{2\sigma_0^2}). \tag{48}$$

We use a grid search strategy to select best parameters for all involved methods in the region $[2^{-7}, 2^7]$ with exponential growth 0.5. The experimental setting and parameters selection are the same as in the above experiment. From the experimental results in Table 3, we can find that DMTSVM outperforms TSVM. MCTSVM is a little better than DMTSVM and outperforms CTSVM. Although the performance of SVM is a little better than TSVM, the performance of our proposed CTSVM is better than SVM and TSVM. However, MCTSVM gives almost the same performance as MSVM and MTGP, and is a little better than DMTSVM. DMTSVM and MCTSVM outperform the corresponding single task learning methods.

[Table 3 about here.]

5.4. Monk

The Monk dataset is the basis of a first international comparison of learning algorithms. There are a total of three problems corresponding to three tasks. In Monk dataset, for each task, we select 120 examples in our experiments. We use three-fold cross-validation to get the average classification accuracy rates and employ an RBF kernel. We use a grid search strategy to select best parameters for all involved methods in the region $[2^{-7}, 2^{7}]$ with exponential growth 0.5. The experimental setting and parameters selection are the same as in the above experiment. From the experimental results in Table 4, we can find that DMTSVM outperforms TSVM. MCTSVM is a little better than DMTSVM and outperforms CTSVM. Although the performance of SVM is a little better than TSVM, the performance of our proposed CTSVM is better than SVM and TSVM. However, MCTSVM is a little better than DMTSVM and MSVM. DMTSVM and MCTSVM outperform the corresponding single task learning methods. MTGP has the best performance on this dataset.

[Table 4 about here.]

6. Conclusion and future work

In this paper, we have proposed CTSVMs and MCTSVMs. CTSVMs overcome the shortage of TSVMs by introducing class centroids to reduce the sensitivity of classifiers with respect to outliers. MCTSVMs are an extension of CTSVMs to the multitask learning scenario. Experimental results on synthetic data indicate that CTSVMs can be more robust to outliers than TSVMs. Experimental results on real-world data validate the good performance of CTSVMs and MCTSVMs. It would be interesting for future work to consider the extension of MCTSVMs to the situation that uses different feature spaces for different tasks and even for different hyperplanes from the same task.

Acknowledgements

This work is supported by the National Natural Science Foundation of China under Projects 61370175 and 61075005, and Shanghai Knowledge Service Platform Project (No. ZF1213).

References

[1] J. Shawe-Taylor, S. Sun, A review of optimization methodologies in support vector machines, Neurocomputing, 74 (2011) 3609-3618.

[2] V.N. Vapnik, The Nature of Statistical Learning Theory, New York: Springer-Verlag, 1995.

[3] B. Scholkopf, A. Smola, Learning with Kernels, Cambridge: MIT Press, 2003.

[4] Q. Song, W. Hu, W. Xie, Robust support vector machine with bullet hole image classification, IEEE Transactions on Systems, 32 (2002) 440-448.

[5] O.L. Mangasarian, E.W. Wild, MultisurFace proximal support vector machine classification via generalized eigenvalues, IEEE Transactions on Pattern Analysis and Machine Intelligence, 28 (2006) 69-74.

[6] R. Jayadeva, S. Khemchandani, Chandra, Twin support vector machines for pattern classification, IEEE Transactions on Pattern Analysis and Machine Intelligence, 74 (2007) 905-910.

[7] S. Ghorai, Mukherjee, P.K. Dutta, Nonparallel plane proximal classifier, Signal Processing, 89 (2009) 510-522.

[8] Y. Shao, C. Zhang, X. Wang, N. Deng, Improvements on twin support vector machines, IEEE Trans on Neural Networks, 22 (2011) 962-968.

[9] S. Ding, Y. Zhao, B. Qi, H. Huang, An overview on twin support vector machines, Artificial Intelligence Review, 2012.

[10] M.A. Kumar, M. Gopal, Least squares twin support vector machines for pattern classification, Expert Systems with Applications, 36 (2009) 7535-7543.

[11] Y.H. Shao, Z. Wang, W.J. Chen, N.Y. Deng, Least squares twin parametric-margin support vector machines for classification, Applied Intelligence, 39 (2013) 451-464.

[12] Y.H. Shao, N.Y. Deng, Z.M. Yang, Least squares recursive projection twin support vector machine for classification, Pattern Recognition, 45 (2012) 2299-2307.

[13] 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) 203-210.

[14] X. Chen, J. Yang, Q. Ye, J. Liang, Recursive projection twin support vector machine via within-class variance minimization, Pattern Recognition, 44 (2011) 2643-2655.

[15] T. Kato, H. Kashima, M. Sugiyama, Multi-task learning via conic programming, Advances in Neural Information Processing Systems, 20 (2008) 737-744.

[16] S. Ben-David, R. Schuller, Exploiting task relatedness for multiple task learning, in: Proceedings of the International Conference on Learning Theory, 2003, pp. 567-580.

[17] S. Parameswaran, K.Q. Weinberger, Large margin multi-task metric learning, Advances in Neural Information Processing Systems, 23 (2010) 1867-1875.

[18] A. Argyriou, T. Evgeniou, M. Pontil, Convex multi-task feature learning, Machine Learning, 73 (2008) 243-272.

[19] S. Sun, Multitask learning for EEG-based biometrics, in: Proceedings of the 19th International Conference on Pattern Recognition, 2008, pp. 1-4.

[20] R. Caruana, Multitask learning, Machine Learning, 28 (1997) 41-75.

[21] T. Evgeniou, C.A. Micchelli, Learning multiple tasks with kernel methods, Journal of Machine Learning Research, 6 (2005) 615-637.

[22] R.K. Ando, T. Zhang, A framework for learning predictive structures from multiple tasks and unlabeled data, Journal of Machine Learning Research, 6 (2005) 1817-1853.

[23] O. Chapelle, P. Shivaswamy, S. Vadrevu, K. Weinberger, Boosted multitask learning, Machine Learning, 85 (2011) 149-173.

[24] Y. Ji, S. Sun, Multitask multiclass support vector machines: Model and experiments, Pattern Recognition, 46 (2012) 914-924.

[25] X. Yuan, S. Yuan, Visual classification with multi-task joint sparse representation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2010, pp. 3493-3500.

[26] Z. Zhang, J. Yan, T. Li, B. Rao, S. Fang, K. Sungeun, S.L. Risacher, A.J. Saykin, L. Shen, Sparse Bayesian multi-task learning for predicting cognitive outcomes from neuroimaging measures in Alzheimer's disease, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2012, pp. 940-947.

[27] T. Zhang, B. Ghanem, S. Liu, N. Ahuja, Roubst visual tracking via multi-task sparse learning, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2012, pp. 2042-2049.

[28] X. Wang, C. Zhang, Z. Zhang, Boosted multi-task learning for face verification with applications to web image and video search, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2009, pp. 142-149.

[29] J. Baxter, A model for inductive bias learning, Journal of Artificial Intelligence Research, 12 (2000) 149-198.

[30] B. Bakker, T. Heskes, Task clustering and gating for Bayesian multitask learning, Journal of Machine Learning Research, 4 (2003) 83-99.

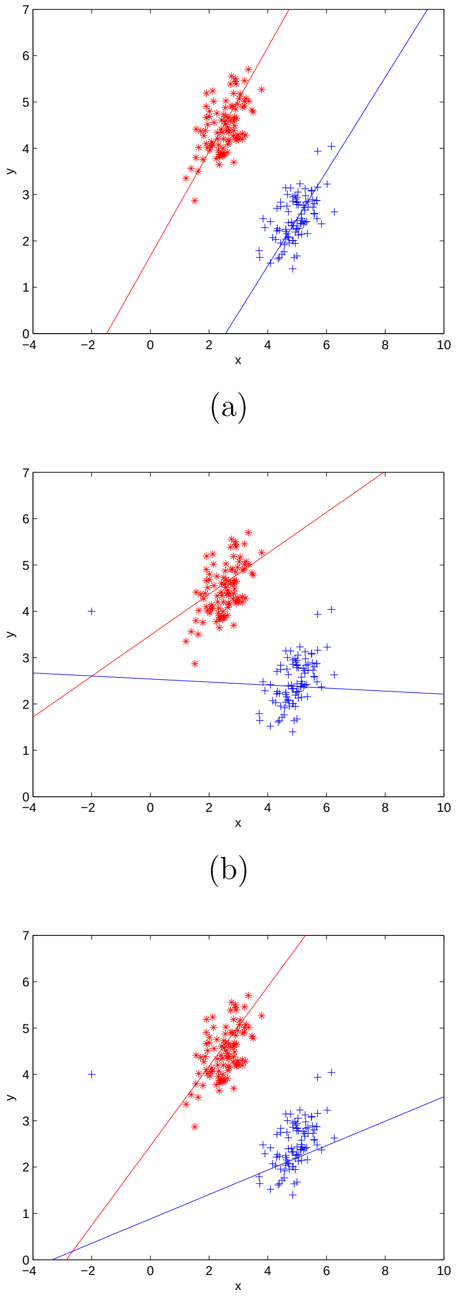
[31] T. Evgeniou, M. Pontil, Regularized multi-task learning, in: Proceeding of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2004, pp. 109-117.

[32] T. Jebara, Multi-task feature and kernel selection for SVMs, in: Proceedings of the 21th International Conference on Machine Learning, 2004, pp. 1-8.

[33] X. Xie, S. Sun, Multitask twin support vector machines, in: Proceeding of the 19th International Conference on Neural Information Processing, 2012, pp. 341-348.

[34] G. Skolidis, G. Sanguinetti, Bayesian multitask classification with Gaussian process priors, IEEE Tranctions on Neural Networks, 22 (2011) 2011-2021.

List of Figures



(a)

(b)

(c)

Figure 1: The training examples and classifiers obtained by different methods : (a) TSVMs without outliers (b) TSVMs with outliers (c) CTSVMs with outliers

List of Tables

Table 1: Datasets.

Name	Attributes	Instances	Classes	Tasks
Isolet	276	7977	26	5
Landmine	9	9674	2	19
Monk	7	432	2	3

Table 2: Classification accuracies (%) on Isolet.

Method	Task1	Task2	Task3	Task4	Task5	All Tasks
1-NN	88.33	85.83	85.00	83.33	76.67	83.83±1.43
SVM	95.00	95.83	90.83	90.00	85.83	91.50±1.08
MSVM	95.00	95.83	94.17	92.5	88.33	93.17±0.62
TSVM	95.00	95.00	90.83	87.5	85.83	90.83±1.03
DMTSVM	**96.67**	**97.50**	95.83	91.67	90.83	94.50±1.63
MTGP	79.13	80.25	76.75	80.13	80.62	79.38±18.47
CTSVM	95.83	95.00	90.83	88.33	90.00	92.00±1.63
MCTSVM	95.83	**97.50**	**96.67**	**98.33**	**92.5**	**96.17**±0.47

Table 3: Classification accuracies (%) on Landmine.

Method	Task1	Task2	All Tasks
1-NN	72.67	68.00	70.33±10
SVM	82.67	74.67	78.67±2.87
MSVM	84.67	76.67	**80.67**±5.25
TSVM	82.67	72.00	77.33 ± 6.85
DMTSVM	**85.33**	75.33	80.33 ± 7.59
MTGP	81.33	**80.00**	**80.67**±2.00
CTSVM	82.67	76.00	79.33±6.18
MCTSVM	84.67	76.67	**80.67**±7.32

Table 4: Classification accuracies (%) on Monk.

Method	Task1	Task2	Task3	All Tasks
1-NN	67.50	52.50	53.33	57.78±4.11
SVM	69.17	57.50	79.17	68.61±3.47
MSVM	69.17	**60.83**	76.67	68.89±2.93
TSVM	67.50	60.00	76.67	68.06±3.94
DMTSVM	73.33	59.17	78.33	70.28±4.88
MTGP	**94.17**	54.17	**87.5**	**78.6**±0.48
CTSVM	69.17	**60.83**	80.83	70.28±4.27
MCTSVM	73.33	59.17	79.10	70.55±4.73