https://doi.org/10.1007/s10489-017-1087-9

The aLS-SVM based multi-task learning classifiers

Liyun Lu[1] · Qiang Lin[1] · Huimin Pei[1] · Ping Zhong[1]

Published online: 22 November 2017

Abstract The multi-task learning support vector machines (SVMs) have recently attracted considerable attention since the conventional single task learning ones usually ignore the relatedness among multiple related tasks and train them separately. Different from the single task learning, the multi-task learning methods can capture the correlation among tasks and achieve an improved performance by training all tasks simultaneously. In this paper, we make two assumptions on the relatedness among tasks. One is that the normal vectors of the related tasks share a certain common parameter value; the other is that the models of the related tasks are close enough and share a common model. Under these assumptions, we propose two multi-task learning methods, named as MTL-aLS-SVM I and MTL-aLS-SVM II respectively, for binary classification by taking full advantages of multi-task learning and the asymmetric least squared loss. MTL-aLS-SVM I seeks for a trade-off between the maximal expectile distance for each task model and the closeness of each task model to the averaged model. MTL-aLS-SVM II can use different kernel functions for different tasks, and it is an extension of the MTL-aLS-SVM I. Both of them can be easily implemented by solving quadratic programming. In addition, we develop their special cases which include L2-SVM based multi-task learning methods (MTL-L2-SVM I and MTL-L2-SVM II) and the least squares SVM (LS-SVM) based multi-task learning methods (MTL-LS-SVM I and MTL-LS-SVM II). Although the MTL-L2-SVM II and MTL-LS-SVM II appear in the form of special cases, they are firstly proposed in this paper. The experimental results show that the proposed methods are very encouraging.

Keywords Multi-task learning · Support vector machine · Asymmetric least squared loss

1 Introduction

Multi-task learning which is an important and ongoing issue in machine learning has attracted growing attention in many regions, such as multi-level analysis [1], semi-supervised learning [2], medical diagnosis [3], speech recognition [4], web search ranking [5], and cell biology [6]. The basic idea of multi-task learning is to obtain the satisfactory performance for each task by simultaneously learning multiple tasks with underlying cross relatedness [7, 8]. Different from single task learning, multi-task learning shares the useful knowledge among multiple tasks, which is helpful to improve the generalization performance. And determining the relatedness among the multiple tasks is important for establishing the formulations of the multi-task learning approaches [9–11]. Although the single task learning methods have achieved successful applications in many areas, they train each task independently and ignore the potential relatedness among tasks, which may reduce the accuracy of prediction. When there are correlations between tasks, it is more reasonable to learn all tasks simultaneously rather than separately [7].

The regularized multi-task learning methods proposed by Evgenious and Pontil [12, 13] generalize the kernel-based methods from single task learning to multi-task learning. Recently, the multi-task learning strategy has been applied in evolutionary algorithm [6], deep neural network [14], pattern recognition [15], support vector machine and so on.

✉ Ping Zhong
zping@cau.edu.cn

[1] College of Science, China Agricultural University, Beijing, 100083, China

Thereinto, multi-task SVM is a power tool of machine learning, and a lot of literature reveal that the SVM-based multi-task learning methods are effective when the related tasks are trained simultaneously [16–24]. Yang et al. [17] presented a one-class SVM-based multi-task learning method by constraining the solutions of multiple tasks close to each other, and the resulting formulation is a conic programming [16]. He et al. [18] proposed an improved SVM-based multi-task learning method for the one-class classification under the assumption that the parameter value of each task model is close to a mean value [12]. A general formulation which has the ability to employ the different kernels for different tasks was then proposed under the assumption that the models of different tasks are close enough [19]. Sun et al. established a multi-task multi-class SVM approach with a constrained optimization instead of the decomposition methods, which can learn both label-compatible and label-incompatible scenarios [20, 21]. Based on the LS-SVM [25], Xu et al. generalized a multi-task LS-SVM that makes use of the advantages of LS-SVM and multi-task learning [22]. Li et al. proposed a multi-task proximal SVM with looser constraints to improve the training speed [23]. Song et al. proposed a novel formulation for multi-task learning by extending the relative margin machine (RMM) to the multi-task learning paradigm [24].

As an important part of machine learning, SVM has been widely studied in many fields, such as multi-class classification [26], feature selection [27], multi-instance multi-label learning [28], and nonparallel least square support vector machine (NLSSVM) [29]. A wide spectrum of successful applications show that SVM is an advanced classifier. As is well known, the loss function plays a key role in SVM, and the different support vector approaches can be established by using the corresponding loss functions [25, 30–35]. The typical loss functions include hinge loss function, least squared loss function, and insensitive loss function. All of these functions are convex and convenient to make calculations and theoretical analysis. Recently, a novel asymmetric squared loss function and the corresponding asymmetric least squares SVM (aLS-SVM) were proposed by Huang et al. [31]. Compared with LS-SVM, the aLS-SVM is more flexible since it introduces the expectile value in the asymmetric squared loss function. The aLS-SVM has the advantage of considerable robustness to the noise around the decision boundary and stability to re-sampling.

In this paper, we propose two aLS-SVM based multi-task learning methods and their special cases by integrating the merits of multi-task learning and the asymmetric squared loss function. We first make the assumption as in [12, 18, 20, 22, 23] that the normal vector of the hyperplane corresponding to each task is expressed as the sum of a certain common vector and a private vector, and establish the new method MTL-aLS-SVM I. We prove that the new method strikes a balance between the maximal expectile distance for each task model and the closeness of each task model to the averaged model. Then, we relax the assumption and suppose that each task model is expressed as the sum of a common model and a private model, and establish the second multi-task learning method MTL-aLS-SVM II. Compared with MTL-aLS-SVM I, MTL-aLS-SVM II is more flexible as it can use different kernel functions for different tasks. These two new methods can be easily implemented by solving quadratic programming and simultaneously obtain the decision functions for all tasks. In addition, we also present their special cases: LS-SVM based multi-task learning methods (denoted correspondingly by MTL-LS-SVM I [22] and MTL-LS-SVM II) and L2-SVM based multi-task learning methods (denoted correspondingly by MTL-L2-SVM I and MTL-L2-SVM II). The special cases MTL-LS-SVM II and MTL-L2-SVM II are also our newly proposed methods. We compare these multi-task learning methods with several related effective single-task learning methods including aLS-SVM [31], LS-SVM, L2-SVM, and NLSSVM [29]. The experimental results verify the effectiveness of our proposed multi-task learning methods.

In summary, by incorporating the properties of the multi-task learning and the asymmetric squared loss function, the advantages of our proposed methods are:

- To have a good ability to process multi-task learning problems directly;
- To have the potential to capture the relatedness among multiple related tasks;
- To effectively exploit different kernel functions for different tasks;
- To be more flexible by using the asymmetric squared loss function;
- To be easily implemented by solving quadratic programming.

We organize the rest of this paper as follows. A brief introduction of the aLS-SVM is given in Section 2. Then we detail the MTL-aLS-SVM I and MTL-aLS-SVM II formulations in Section 3. Meanwhile, we give their corresponding special cases in this section. In Section 4, we evaluate the proposed methods by the numerical experiments. Finally, we conclude the paper in Section 5.

2 The aLS-SVM

The asymmetric least squares support vector machine (aLS-SVM) [31] is proposed based on the following asymmetric squared loss function:

$$L_\rho(r) = \begin{cases} \rho r^2, & r \geq 0 \\ (1-\rho) r^2, & r < 0 \end{cases} \tag{1}$$

where ρ ($0 \leq \rho \leq 1$) is the expectile value. Unlike the general SVMs, the aLS-SVM maximizes the expectile distance instead of the minimal distance between two classes and solves the following optimization problem:

$$\begin{aligned} \min_{\boldsymbol{\omega},b,\boldsymbol{\zeta}} \quad & \frac{1}{2}\|\boldsymbol{\omega}\|^2 + \frac{C}{2}\sum_{k=1}^{m} L_\rho(\zeta_i) \\ \text{s.t.} \quad & \zeta_i = 1 - y_i(\boldsymbol{\omega}^T\phi(\boldsymbol{x}_i)+b),\ i = 1, 2, \cdots, m \end{aligned} \tag{2}$$

where $\boldsymbol{\zeta}$ is the error variable vector; $\phi(\cdot)$ is a nonlinear mapping from the input space $\mathbb{R}^d$ into the feature space $\mathbb{R}^h$; C is the regularization parameter. According to the asymmetric squared loss function (1), the optimization problem (2) can be equivalently written as

$$\begin{aligned} \min_{\boldsymbol{\omega},b,\boldsymbol{\zeta}} \quad & \frac{1}{2}\|\boldsymbol{\omega}\|^2 + \frac{C}{2}\sum_{k=1}^{m} {\zeta_i}^2 \\ \text{s.t.} \quad & y_i(\boldsymbol{\omega}^T\phi(\boldsymbol{x}_i)+b) \geq 1 - \frac{1}{\rho}\zeta_i,\ i = 1, 2, \cdots, m \\ & y_i(\boldsymbol{\omega}^T\phi(\boldsymbol{x}_i)+b) \leq 1 + \frac{1}{1-\rho}\zeta_i,\ i = 1, 2, \cdots, m \end{aligned} \tag{3}$$

Compared with the usual SVMs, the aLS-SVM is robust to noise around the decision boundary and stable to resampling because of the maximization of the expectile distance. It is also an extension of L2-SVM and LS-SVM [25]. More details about the aLS-SVM can be seen in [31].

3 The aLS-SVM based multi-task learning formulations

In this section, we propose two aLS-SVM based multi-task learning methods—MTL-aLS-SVM I and MTL-aLS-SVM II according to the different task relatedness assumptions. Meanwhile, we develop two types of special cases of these two multi-task learning methods. In the multi-task learning scenario, we are given N different but related tasks. For each task k, we have m_k training data $\{(\boldsymbol{x}_{ki}, y_{ki})\}_{i=1}^{m_k}$, where $\boldsymbol{x}_{ki} \in \mathbb{R}^d$ and $y_{ki} \in \{1, -1\}$. Thus, we totally have $m = \sum_{k=1}^{N} m_k$ training data. Our aim is to learn N different decision functions (hyperplanes) for each task simultaneously.

3.1 MTL-aLS-SVM I

In the light of the method presented in [12], when the related tasks share a common function $\boldsymbol{\omega}_0$, the normal vector $\boldsymbol{\omega}_k \in \mathbb{R}^h$ for the specific task k can be expressed as $\boldsymbol{\omega}_k = \boldsymbol{\omega}_0 + \boldsymbol{v}_k$, where $\boldsymbol{v}_k$ represents the private information of task k. Under this assumption, we elaborate the primal optimization problem of MTL-aLS-SVM I as follows.

$$\begin{aligned} \min_{\boldsymbol{\omega}_0,\boldsymbol{v}_k,b_k,\boldsymbol{\zeta}_k} \quad & \frac{1}{2}\|\boldsymbol{\omega}_0\|^2 + \frac{C_1}{2}\sum_{k=1}^{N}\|\boldsymbol{v}_k\|^2 + \frac{C_2}{2}\sum_{k=1}^{N}\|\boldsymbol{\zeta}_k\|^2 \\ \text{s.t.} \quad & \boldsymbol{Z}_k^T(\boldsymbol{\omega}_0 + \boldsymbol{v}_k) + b_k\boldsymbol{y}_k \geq \boldsymbol{e}_{m_k} - \frac{1}{\rho}\boldsymbol{\zeta}_k, \\ & k = 1, 2, \cdots, N \\ & \boldsymbol{Z}_k^T(\boldsymbol{\omega}_0 + \boldsymbol{v}_k) + b_k\boldsymbol{y}_k \leq \boldsymbol{e}_{m_k} + \frac{1}{1-\rho}\boldsymbol{\zeta}_k, \\ & k = 1, 2, \cdots, N \end{aligned} \tag{4}$$

where $\boldsymbol{Z}_k = \big(y_{k1}\phi(\boldsymbol{x}_{k1}), y_{k2}\phi(\boldsymbol{x}_{k2}), \cdots, y_{km_k}\phi(\boldsymbol{x}_{km_k})\big) \in \mathbb{R}^{h\times m_k}$ with $\phi(\cdot)$ having the same meaning as in (3); $\boldsymbol{y}_k = (y_{k1}, y_{k2}, \cdots, y_{km_k})^T$; $\boldsymbol{\zeta}_k = (\zeta_{k1}, \zeta_{k2}, \cdots, \zeta_{km_k})^T \in \mathbb{R}^{m_k}$ is the slack variable vector for task k; $\boldsymbol{e}_{m_k} = (1, 1, \cdots, 1)^T \in \mathbb{R}^{m_k}$; C_1 and C_2 are the positive regularization parameters. We introduce C_1 to control the trade-off between the public classification information $\boldsymbol{\omega}_0$ and the dissimilarity among all tasks. Specifically, bigger C_1 enforces MTL-aLS-SVM I to train a common model, while smaller C_1 will make MTL-aLS-SVM I learn each task model independently. It is shown from (4) that N different tasks are trained simultaneously because of the connection of the public classification information.

The Lagrangian of the primal problem (4) is

$$\begin{aligned} & \mathcal{L}(\boldsymbol{\omega}_0, \boldsymbol{v}_k, b_k, \boldsymbol{\zeta}_k, \boldsymbol{\alpha}_k, \boldsymbol{\beta}_k) \\ & = \frac{1}{2}\|\boldsymbol{\omega}_0\|^2 + \frac{C_1}{2}\sum_{k=1}^{N}\|\boldsymbol{v}_k\|^2 + \frac{C_2}{2}\sum_{k=1}^{N}\|\boldsymbol{\zeta}_k\|^2 \\ & \quad - \sum_{k=1}^{N}\boldsymbol{\alpha}_k^T\left(\boldsymbol{Z}_k^T(\boldsymbol{\omega}_0+\boldsymbol{v}_k)+b_k\boldsymbol{y}_k-\boldsymbol{e}_{m_k}+\frac{1}{\rho}\boldsymbol{\zeta}_k\right) \\ & \quad + \sum_{k=1}^{N}\boldsymbol{\beta}_k^T\left(\boldsymbol{Z}_k^T(\boldsymbol{\omega}_0+\boldsymbol{v}_k)+b_k\boldsymbol{y}_k-\boldsymbol{e}_{m_k}-\frac{1}{1-\rho}\boldsymbol{\zeta}_k\right) \end{aligned} \tag{5}$$

where $\boldsymbol{\alpha}_k = (\alpha_{k1}, \alpha_{k2}, \cdots, \alpha_{km_k})^T$ and $\boldsymbol{\beta}_k = (\beta_{k1}, \beta_{k2}, \cdots, \beta_{km_k})^T$ are the nonnegative Lagrange multiplier vectors. By differentiating the Lagrangian with respect to $\boldsymbol{\omega}_0$, $\boldsymbol{v}_k$, $\boldsymbol{\zeta}_k$, b_k based on the Karush-Kuhn-Tucker (KKT) condition, we get the following equations:

$$\boldsymbol{\omega}_0 = \sum_{k=1}^{N}\boldsymbol{Z}_k(\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k) \tag{6}$$

$$\boldsymbol{v}_k = \frac{1}{C_1}\boldsymbol{Z}_k(\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k) \tag{7}$$

$$\boldsymbol{\zeta}_k = \frac{1}{C_2}\left(\frac{1}{\rho}\boldsymbol{\alpha}_k + \frac{1}{1-\rho}\boldsymbol{\beta}_k\right) \tag{8}$$

$$(\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k)^T\boldsymbol{y}_k = 0 \tag{9}$$

By (6) and (7), we have

$$\boldsymbol{\omega}_0 = C_1 \sum_{k=1}^{N} \boldsymbol{v}_k \tag{10}$$

which shows that $\boldsymbol{\omega}_0$ is a linear combination of $\boldsymbol{v}_k$. Since $\boldsymbol{\omega}_k = \boldsymbol{\omega}_0 + \boldsymbol{v}_k$, we further have

$$\boldsymbol{\omega}_0 = \frac{C_1}{1 + C_1 N} \sum_{k=1}^{N} \boldsymbol{\omega}_k \tag{11}$$

Substituting $\boldsymbol{\omega}_0$, $\boldsymbol{v}_k$ by $\boldsymbol{\omega}_k$, we get the following equivalent form of the objective function of the primal problem (4) (for the proof of (12), see the Appendix).

$$\frac{\tau_1}{2} \sum_{k=1}^{N} \|\boldsymbol{\omega}_k\|^2 + \frac{\tau_2}{2} \sum_{k=1}^{N} \|\boldsymbol{\omega}_k - \bar{\boldsymbol{\omega}}\|^2 + \frac{C_2}{2} \sum_{k=1}^{N} \|\boldsymbol{\zeta}_k\|^2 \tag{12}$$

where $\bar{\boldsymbol{\omega}} = \frac{1}{N} \sum_{k=1}^{N} \boldsymbol{\omega}_k$ is the mean vector of $\boldsymbol{\omega}_1, \cdots, \boldsymbol{\omega}_N$, $\tau_1 = \frac{C_1}{1+C_1 N}$, $\tau_2 = \frac{C_1^2 N}{1+C_1 N}$. It is shown by (12) and the constraints of (4) that the newly proposed MTL-aLS-SVM I seeks for a trade-off between the maximum expectile distance for each task model and the closeness of each task model to the averaged model.

Substituting (6)–(9) into the Lagrangian (5), we get the following dual form of the primal problem (4):

$$\begin{aligned}
\max_{\boldsymbol{\alpha},\boldsymbol{\beta}} \quad & -\frac{1}{2} \sum_{k,j=1}^{N} (\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k)^T \boldsymbol{Z}_k^T \boldsymbol{Z}_j (\boldsymbol{\alpha}_j - \boldsymbol{\beta}_j) \\
& - \frac{1}{2C_1} \sum_{k=1}^{N} (\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k)^T \boldsymbol{Z}_k^T \boldsymbol{Z}_k (\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k) \\
& - \frac{1}{2C_2} \sum_{k=1}^{N} \left(\frac{1}{\rho} \boldsymbol{\alpha}_k + \frac{1}{1-\rho} \boldsymbol{\beta}_k \right)^T \left(\frac{1}{\rho} \boldsymbol{\alpha}_k + \frac{1}{1-\rho} \boldsymbol{\beta}_k \right) \\
& + \sum_{k=1}^{N} (\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k)^T \boldsymbol{e}_{m_k} \\
\text{s.t.} \quad & (\boldsymbol{\alpha}_k - \boldsymbol{\beta}_k)^T \boldsymbol{y}_k = 0, \ k = 1, 2, \cdots, N \\
& \boldsymbol{\alpha}_k \geq \boldsymbol{0}, \ k = 1, 2, \cdots, N \\
& \boldsymbol{\beta}_k \geq \boldsymbol{0}, \ k = 1, 2, \cdots, N
\end{aligned} \tag{13}$$

where $\boldsymbol{\alpha} = (\boldsymbol{\alpha}_1^T, \boldsymbol{\alpha}_2^T, \cdots, \boldsymbol{\alpha}_N^T)^T$ and $\boldsymbol{\beta} = (\boldsymbol{\beta}_1^T, \boldsymbol{\beta}_2^T, \cdots, \boldsymbol{\beta}_N^T)^T$. By setting $\boldsymbol{\lambda}_k = \boldsymbol{\alpha}_k - \boldsymbol{\beta}_k$, we rewrite (13) as

$$\begin{aligned}
\min_{\boldsymbol{\lambda}_k,\boldsymbol{\beta}_k} \quad & \frac{1}{2} \sum_{k,j=1}^{N} {\boldsymbol{\lambda}_k}^T \boldsymbol{Z}_k^T \boldsymbol{Z}_j \boldsymbol{\lambda}_j + \frac{1}{2C_1} \sum_{k=1}^{N} {\boldsymbol{\lambda}_k}^T {\boldsymbol{Z}_k}^T \boldsymbol{Z}_k \boldsymbol{\lambda}_k \\
& + \frac{1}{2\rho^2 C_2} \sum_{k=1}^{N} \left(\boldsymbol{\lambda}_k + \frac{1}{1-\rho} \boldsymbol{\beta}_k \right)^T \left(\boldsymbol{\lambda}_k + \frac{1}{1-\rho} \boldsymbol{\beta}_k \right) \\
& - \sum_{k=1}^{N} {\boldsymbol{\lambda}_k}^T \boldsymbol{e}_{m_k} \\
\text{s.t.} \quad & {\boldsymbol{\lambda}_k}^T \boldsymbol{y}_k = 0, \ k = 1, 2, \cdots, N \\
& \boldsymbol{\lambda}_k + \boldsymbol{\beta}_k \geq 0, \ k = 1, 2, \cdots, N \\
& \boldsymbol{\beta}_k \geq 0, \ k = 1, 2, \cdots, N
\end{aligned} \tag{14}$$

where $\boldsymbol{\lambda}_k = (\lambda_{k1}, \lambda_{k2}, \cdots, \lambda_{km_k})^T$. Furthermore, the objective function of (14) can be rewritten as

$$\begin{aligned}
& \frac{1}{2} \sum_{k,j=1}^{N} \sum_{i=1}^{m_k} \sum_{r=1}^{m_j} \lambda_{ki} \lambda_{jr} y_{ki} y_{jr} \left(1 + \frac{\delta_{kj}}{C_1} \right) K(\boldsymbol{x}_{ki}, \boldsymbol{x}_{jr}) \\
& + \frac{1}{2\rho^2 C_2} \sum_{k=1}^{N} \left(\boldsymbol{\lambda}_k + \frac{1}{1-\rho} \boldsymbol{\beta}_k \right)^T \left(\boldsymbol{\lambda}_k + \frac{1}{1-\rho} \boldsymbol{\beta}_k \right) \\
& - \sum_{k=1}^{N} {\boldsymbol{\lambda}_k}^T \boldsymbol{e}_{m_k}
\end{aligned} \tag{15}$$

where

$$\delta_{kj} = \begin{cases} 1, k = j \\ 0, k \neq j \end{cases} \tag{16}$$

Denote $\boldsymbol{\lambda}_k^*$, $k = 1, \cdots, N$ as the optimal solutions of the above optimization problem. Then the decision function for task k can be obtained as

$$\begin{aligned}
f_k(\boldsymbol{x}) &= \operatorname{sign} \left(\phi(\boldsymbol{x})^T \left(\sum_{k=1}^{N} \boldsymbol{Z}_k \boldsymbol{\lambda}_k^* + \frac{1}{C_1} \boldsymbol{Z}_k \boldsymbol{\lambda}_k^* \right) + b_k^* \right) \\
&= \operatorname{sign} \left(\sum_{k=1}^{N} \sum_{i=1}^{m_k} \lambda_{ki}^* y_{ki} K(\boldsymbol{x}_{ki}, \boldsymbol{x}) \right. \\
& \qquad \left. + \frac{1}{C_1} \sum_{i=1}^{m_k} \lambda_{ki}^* y_{ki} K(\boldsymbol{x}_{ki}, \boldsymbol{x}) + b_k^* \right)
\end{aligned} \tag{17}$$

where $K(\cdot, \cdot)$ is a kernel function, and the optimal value b_k^* can be obtained by the following equations:

$$\boldsymbol{Z}_{ki}^T \boldsymbol{Z} \boldsymbol{\lambda} + \frac{1}{C_1} \boldsymbol{Z}_{ki}^T \boldsymbol{Z}_k \boldsymbol{\lambda}_k + y_{ki} b_k = 1 - \frac{1}{\rho} \zeta_{ki}, \forall ki : \alpha_{ki} > 0 \tag{18}$$

$$\boldsymbol{Z}_{ki}^T \boldsymbol{Z} \boldsymbol{\lambda} + \frac{1}{C_1} \boldsymbol{Z}_{ki}^T \boldsymbol{Z}_k \boldsymbol{\lambda}_k + y_{ki} b_k = 1 + \frac{1}{1-\rho} \zeta_{ki}, \forall ki : \beta_{ki} > 0 \tag{19}$$

where $\boldsymbol{\lambda} = \left(\boldsymbol{\lambda}_1^T, \boldsymbol{\lambda}_2^T, \cdots, \boldsymbol{\lambda}_N^T \right)^T$; $\boldsymbol{Z} = (\boldsymbol{Z}_1, \boldsymbol{Z}_2, \cdots, \boldsymbol{Z}_N) \in \mathbb{R}^{h \times m}$, and $\boldsymbol{Z}_{ki}$ is the ith column of $\boldsymbol{Z}_k$.

3.2 MTL-aLS-SVM II

Next, we present an other elegant formulation under the assumption that all tasks share a common model, and every task function f_k can be expressed as the sum of the common function h_0 and the private function h_k:

$$\begin{aligned}
f_k &= h_0 + h_k \\
&= \langle \boldsymbol{\omega}_0, \phi_0(\boldsymbol{x}) \rangle + \langle \boldsymbol{v}_k, \phi_k(\boldsymbol{x}) \rangle + b_k
\end{aligned}$$

where $\boldsymbol{\omega}_0$ and ϕ_0 are the normal vector and nonlinear feature mapping for the common model, respectively, and $\boldsymbol{v}_k$

and ϕ_k are those for the private model. We denote the offset $b_0 + b_k$ by b_k for simplicity. Obviously, ϕ_0 and ϕ_k for the different task k can be the different nonlinear mappings, and compared with MTL-aLS-SVM I in which only one nonlinear transformation is employed, MTL-aLS-SVM II is its extension. If $\phi_0 = \phi_k$, then MTL-aLS-SVM II reduces to MTL-aLS-SVM I.

We establish MTL-aLS-SVM II by solving the following optimization problem:

$$\begin{aligned}\min_{\boldsymbol{\omega}_0,\boldsymbol{v}_k,b_k,\boldsymbol{\zeta}_k}\ & \frac{1}{2}\|\boldsymbol{\omega}_0\|^2+\frac{C_1}{2}\sum_{k=1}^{N}\|\boldsymbol{v}_k\|^2+\frac{C_2}{2}\sum_{k=1}^{N}\|\boldsymbol{\zeta}_k\|^2\\ \text{s.t.}\ & \tilde{\boldsymbol{Z}}_k^T\boldsymbol{\omega}_0+\boldsymbol{A}_k^T\boldsymbol{v}_k+b_k\boldsymbol{y}_k\geq\boldsymbol{e}_{m_k}-\frac{1}{\rho}\boldsymbol{\zeta}_k,\\ & k=1,2,\cdots,N\\ & \tilde{\boldsymbol{Z}}_k^T\boldsymbol{\omega}_0+\boldsymbol{A}_k^T\boldsymbol{v}_k+b_k\boldsymbol{y}_k\leq\boldsymbol{e}_{m_k}+\frac{1}{1-\rho}\boldsymbol{\zeta}_k,\\ & k=1,2,\cdots,N\end{aligned}\tag{20}$$

where $\tilde{\boldsymbol{Z}}_k = (y_{k1}\phi_0(\boldsymbol{x}_{k1}), y_{k2}\phi_0(\boldsymbol{x}_{k2}), \cdots, y_{km_k}\phi_0 \cdots (\boldsymbol{x}_{km_k})) \in \mathbb{R}^{h\times m_k}$; $\boldsymbol{A}_k = (y_{k1}\phi_k(\boldsymbol{x}_{k1}), y_{k2}\phi_k(\boldsymbol{x}_{k2})\cdots, y_{km_k}\phi_k(\boldsymbol{x}_{km_k})) \in \mathbb{R}^{h\times m_k}$, $\boldsymbol{\zeta}_k$, $\boldsymbol{y}_k$, $\boldsymbol{e}_{m_k}$ C_1, and C_2 have the same meanings as in formula (4).

The Lagrangian function of the above optimization problem is

$$\begin{aligned}&\mathcal{L}(\boldsymbol{\omega}_0, \boldsymbol{v}_k, b_k, \boldsymbol{\zeta}_k, \boldsymbol{\alpha}_k, \boldsymbol{\beta}_k)\\ &=\frac{1}{2}\|\boldsymbol{\omega}_0\|^2+\frac{C_1}{2}\sum_{k=1}^{N}\|\boldsymbol{v}_k\|^2+\frac{C_2}{2}\sum_{k=1}^{N}\|\boldsymbol{\zeta}_k\|^2\\ &\quad-\sum_{k=1}^{N}\boldsymbol{\alpha}_k^T\left(\tilde{\boldsymbol{Z}}_k^T\boldsymbol{\omega}_0+\boldsymbol{A}_k^T\boldsymbol{v}_k+b_k\boldsymbol{y}_k-\boldsymbol{e}_{m_k}+\frac{1}{\rho}\boldsymbol{\zeta}_k\right)\\ &\quad+\sum_{k=1}^{N}\boldsymbol{\beta}_k^T\left(\tilde{\boldsymbol{Z}}_k^T\boldsymbol{\omega}_0+\boldsymbol{A}_k^T\boldsymbol{v}_k+b_k\boldsymbol{y}_k-\boldsymbol{e}_{m_k}-\frac{1}{1-\rho}\boldsymbol{\zeta}_k\right)\end{aligned}\tag{21}$$

where $\boldsymbol{\alpha}_k=(\alpha_{k1}, \alpha_{k2}, \cdots, \alpha_{km_k})^T$ and $\boldsymbol{\beta}_k=(\beta_{k1}, \beta_{k2}, \cdots, \beta_{km_k})^T$ are the nonnegative Lagrange multiplier vectors. According to the KKT condition, we get the following equations:

$$\boldsymbol{\omega}_0=\sum_{k=1}^{N}\tilde{\boldsymbol{Z}}_k(\boldsymbol{\alpha}_k-\boldsymbol{\beta}_k)\tag{22}$$

$$\boldsymbol{v}_k=\frac{1}{C_1}\boldsymbol{A}_k(\boldsymbol{\alpha}_k-\boldsymbol{\beta}_k)\tag{23}$$

$$\boldsymbol{\zeta}_k=\frac{1}{C_2}\left(\frac{1}{\rho}\boldsymbol{\alpha}_k+\frac{1}{1-\rho}\boldsymbol{\beta}_k\right)\tag{24}$$

$$(\boldsymbol{\alpha}_k-\boldsymbol{\beta}_k)^T\boldsymbol{y}_k=0\tag{25}$$

By substituting (22)–(25) into (21), we obtain the following dual program of (20):

$$\begin{aligned}\max_{\boldsymbol{\alpha},\boldsymbol{\beta}}\ & -\frac{1}{2}\sum_{k,j=1}^{N}(\boldsymbol{\alpha}_k-\boldsymbol{\beta}_k)^T\tilde{\boldsymbol{Z}}_k^T\tilde{\boldsymbol{Z}}_j(\boldsymbol{\alpha}_j-\boldsymbol{\beta}_j)\\ & -\frac{1}{2C_1}\sum_{k=1}^{N}(\boldsymbol{\alpha}_k-\boldsymbol{\beta}_k)^T\boldsymbol{A}_k^T\boldsymbol{A}_k(\boldsymbol{\alpha}_k-\boldsymbol{\beta}_k)\\ & -\frac{1}{2C_2}\sum_{k=1}^{N}\left(\frac{1}{\rho}\boldsymbol{\alpha}_k+\frac{1}{1-\rho}\boldsymbol{\beta}_k\right)^T\left(\frac{1}{\rho}\boldsymbol{\alpha}_k+\frac{1}{1-\rho}\boldsymbol{\beta}_k\right)\\ & +\sum_{k=1}^{N}(\boldsymbol{\alpha}_k-\boldsymbol{\beta}_k)^T\boldsymbol{e}_{m_k}\\ \text{s.t.}\ & (\boldsymbol{\alpha}_k-\boldsymbol{\beta}_k)^T\boldsymbol{y}_k=0,\ k=1,2,\cdots,N\\ & \boldsymbol{\alpha}_k\geq\mathbf{0},\ k=1,2,\cdots,N\\ & \boldsymbol{\beta}_k\geq\mathbf{0},\ k=1,2,\cdots,N\end{aligned}\tag{26}$$

where $\boldsymbol{\alpha} = (\boldsymbol{\alpha}_1^T, \boldsymbol{\alpha}_2^T, \cdots, \boldsymbol{\alpha}_N^T)^T$ and $\boldsymbol{\beta} = (\boldsymbol{\beta}_1^T, \boldsymbol{\beta}_2^T, \cdots, \boldsymbol{\beta}_N^T)^T$.

Setting $\boldsymbol{\lambda}_k=\boldsymbol{\alpha}_k-\boldsymbol{\beta}_k$, we get the equivalent form of (26):

$$\begin{aligned}\min_{\boldsymbol{\lambda}_k,\boldsymbol{\beta}_k}\ & \frac{1}{2}\sum_{k,j=1}^{N}\sum_{i=1}^{m_k}\sum_{r=1}^{m_j}\lambda_{ki}\lambda_{jr}y_{ki}y_{jr}\\ & \times\left(K_0(\boldsymbol{x}_{ki},\boldsymbol{x}_{jr})+\frac{\delta_{kj}}{C_1}K_k(\boldsymbol{x}_{ki},\boldsymbol{x}_{jr})\right)\\ & +\frac{1}{2\rho^2C_2}\sum_{k=1}^{N}\left(\boldsymbol{\lambda}_k+\frac{1}{1-\rho}\boldsymbol{\beta}_k\right)^T\left(\boldsymbol{\lambda}_k+\frac{1}{1-\rho}\boldsymbol{\beta}_k\right)\\ & -\sum_{k=1}^{N}\boldsymbol{\lambda}_k{}^T\boldsymbol{e}_{m_k}\\ \text{s.t.}\ & \boldsymbol{\lambda}_k{}^T\boldsymbol{y}_k=0,\ k=1,2,\cdots,N\\ & \boldsymbol{\lambda}_k+\boldsymbol{\beta}_k\geq0,\ k=1,2,\cdots,N\\ & \boldsymbol{\beta}_k\geq0,\ k=1,2,\cdots,N\end{aligned}\tag{27}$$

where $K_0(\cdot,\cdot)$ and $K_k(\cdot,\cdot)(k = 1, 2, \cdots, N)$ are the kernel functions. It can be seen by comparing the program (27) with (14) (notice (15)) that MTL-aLS-SVM II and MTL-aLS-SVM I are equivalent if $K_0 = K_k$. Therefore, MTL-aLS-SVM II is an extension of MTL-aLS-SVM I.

Denote $\boldsymbol{\lambda}_k^*$, $k = 1, \cdots, N$ as the optimal solutions of the above optimization problem. Then the decision function for task k can be obtained as

$$\begin{aligned}f_k(\boldsymbol{x})&=\text{sign}\left(\phi_0(\boldsymbol{x})^T\sum_{k=1}^{N}\tilde{\boldsymbol{Z}}_k\boldsymbol{\lambda}_k^*+\phi_k(\boldsymbol{x})^T\frac{1}{C_1}\boldsymbol{A}_k\boldsymbol{\lambda}_k^*+b_k^*\right)\\ &=\text{sign}\left(\sum_{k=1}^{N}\sum_{i=1}^{m_k}\lambda_{ki}^*y_{ki}K_0(\boldsymbol{x}_{ki},\boldsymbol{x})\right.\\ &\qquad\left.+\frac{1}{C_1}\sum_{i=1}^{m_k}\lambda_{ki}^*y_{ki}K_k(\boldsymbol{x}_{ki},\boldsymbol{x})+b_k^*\right)\end{aligned}\tag{28}$$

where the optimal value b_k^* can be obtained by the following equations:

$$\sum_{j=1}^{N}\sum_{r=1}^{m_j}\lambda_{jr}y_{ki}y_{jr}\left(K_0(\boldsymbol{x}_{ki},\boldsymbol{x}_{jr})+\frac{\delta_{kj}}{C_1}K_k(\boldsymbol{x}_{ki},\boldsymbol{x}_{jr})\right)$$
$$+y_{ki}b_k=1-\frac{1}{\rho}\zeta_{ki},\ \forall ki:\alpha_{ki}>0$$

$$\sum_{j=1}^{N}\sum_{r=1}^{m_j}\lambda_{jr}y_{ki}y_{jr}\left(K_0(\boldsymbol{x}_{ki},\boldsymbol{x}_{jr})+\frac{\delta_{kj}}{C_1}K_k(\boldsymbol{x}_{ki},\boldsymbol{x}_{jr})\right)$$
$$+y_{ki}b_k=1+\frac{1}{1-\rho}\zeta_{ki},\ \forall ki:\beta_{ki}>0$$

3.3 The special cases

In this subsection, we develop two kinds of special cases of MTL-aLS-SVM I and MTL-aLS-SVM II for the multi-task learning. Recall that the sharp of the asymmetric squared loss function (1) is closely related to the value of ρ. When $\rho = 1$, the asymmetric squared loss (1) reduces to the squared hinge loss:

$$L_\rho(r)=\begin{cases} r^2, & r\geq 0\\ 0, & r<0\end{cases} \tag{29}$$

And accordingly, the MTL-aLS-SVM I and MTL-aLS-SVM II reduce to the L2-SVM based multi-task learning methods (denoted by MTL-L2-SVM I and MTL-L2-SVM II, respectively).

MTL-L2-SVM I:

$$\begin{aligned}\min_{\boldsymbol{\omega}_0,\boldsymbol{v}_k,b_k,\boldsymbol{\zeta}_k}\quad & \frac{1}{2}\|\boldsymbol{\omega}_0\|^2+\frac{C_1}{2}\sum_{k=1}^{N}\|\boldsymbol{v}_k\|^2+\frac{C_2}{2}\sum_{k=1}^{N}\|\boldsymbol{\zeta}_k\|^2\\ \text{s.t.}\quad & \boldsymbol{Z}_k^T(\boldsymbol{\omega}_0+\boldsymbol{v}_k)+b_k\boldsymbol{y}_k\geq \boldsymbol{e}_{m_k}-\boldsymbol{\zeta}_k,\\ & k=1,2,\cdots,N\end{aligned} \tag{30}$$

where $\boldsymbol{Z}_k$, $\boldsymbol{\zeta}_k$, C_1, C_2 and $\boldsymbol{e}_{m_k}$ have the same meanings as in formula (4). By the KKT condition, the dual problem of the above optimization problem can be obtained

$$\begin{aligned}\max_{\boldsymbol{\alpha}}\quad & -\frac{1}{2}\sum_{k,j=1}^{N}\boldsymbol{\alpha}_k{}^T\boldsymbol{Z}_k{}^T\boldsymbol{Z}_j\boldsymbol{\alpha}_j-\frac{1}{2C_1}\sum_{k=1}^{N}\boldsymbol{\alpha}_k{}^T\boldsymbol{Z}_k{}^T\boldsymbol{Z}_k\boldsymbol{\alpha}_k\\ & -\frac{1}{2C_2}\sum_{k=1}^{N}\boldsymbol{\alpha}_k{}^T\boldsymbol{\alpha}_k+\sum_{k=1}^{N}\boldsymbol{\alpha}_k{}^T\boldsymbol{e}_{m_k}\\ \text{s.t.}\quad & \boldsymbol{\alpha}_k{}^T\boldsymbol{y}_k=0,\ k=1,2,\cdots,N\\ & \boldsymbol{\alpha}_k\geq 0,\ k=1,2,\cdots,N\end{aligned} \tag{31}$$

MTL-L2-SVM II:

$$\begin{aligned}\min_{\boldsymbol{\omega}_0,\boldsymbol{v}_k,b_k,\boldsymbol{\zeta}_k}\quad & \frac{1}{2}\|\boldsymbol{\omega}_0\|^2+\frac{C_1}{2}\sum_{k=1}^{N}\|\boldsymbol{v}_k\|^2+\frac{C_2}{2}\sum_{k=1}^{N}\|\boldsymbol{\zeta}_k\|^2\\ \text{s.t.}\quad & \tilde{\boldsymbol{Z}}_k^T\boldsymbol{\omega}_0+\boldsymbol{A}_k^T\boldsymbol{v}_k+b_k\boldsymbol{y}_k\geq \boldsymbol{e}_{m_k}-\boldsymbol{\zeta}_k,\\ & k=1,2,\cdots,N\end{aligned} \tag{32}$$

where $\tilde{\boldsymbol{Z}}_k$, $\boldsymbol{A}_k$ have the same meanings as in (20). The dual form of the above optimization problem is

$$\begin{aligned}\min_{\boldsymbol{\alpha}_k}\quad & \frac{1}{2}\sum_{k,j=1}^{N}\sum_{i=1}^{m_k}\sum_{r=1}^{m_j}\alpha_{ki}\alpha_{jr}y_{ki}y_{jr}\Big(K_0(\boldsymbol{x}_{ki},\boldsymbol{x}_{jr})\\ & +\frac{\delta_{kj}}{C_1}K_k(\boldsymbol{x}_{ki},\boldsymbol{x}_{jr})\Big)+\frac{1}{2C_2}\sum_{k=1}^{N}\boldsymbol{\alpha}_k{}^T\boldsymbol{\alpha}_k-\sum_{k=1}^{N}\boldsymbol{\alpha}_k{}^T\boldsymbol{e}_{m_k}\\ \text{s.t.}\quad & \boldsymbol{\alpha}_k{}^T\boldsymbol{y}_k=0,\ k=1,2,\cdots,N\\ & \boldsymbol{\alpha}_k\geq 0,\ k=1,2,\cdots,N\end{aligned} \tag{33}$$

On the other hand, when $\rho = 1/2$, the asymmetric squared loss (1) reduces to the least squared loss $L_\rho(r)=\frac{1}{2}r^2$. Then the MTL-aLS-SVM I and MTL-aLS-SVM II accordingly turn to be the least squares SVM based multi-task learning methods (denoted by MTL-LS-SVM I and MTL-LS-SVM II, respectively).

MTL-LS-SVM I [22]:

$$\begin{aligned}\min_{\boldsymbol{\omega}_0,\boldsymbol{v}_k,b_k,\boldsymbol{\zeta}_k}\quad & \frac{1}{2}\|\boldsymbol{\omega}_0\|^2+\frac{C_1}{2}\sum_{k=1}^{N}\|\boldsymbol{v}_k\|^2+\frac{C_2}{2}\sum_{k=1}^{N}\|\boldsymbol{\zeta}_k\|^2\\ \text{s.t.}\quad & \boldsymbol{Z}_k^T(\boldsymbol{\omega}_0+\boldsymbol{v}_k)+b_k\boldsymbol{y}_k=\boldsymbol{e}_{m_k}-\boldsymbol{\zeta}_k,\\ & k=1,2,\cdots,N\end{aligned} \tag{34}$$

The optimization problem (34) can be solved by the following linear system:

$$\begin{bmatrix}\boldsymbol{0}_{N\times N} & \boldsymbol{D}^T\\ \boldsymbol{D} & \boldsymbol{H}\end{bmatrix}\begin{bmatrix}\boldsymbol{b}\\ \boldsymbol{\alpha}\end{bmatrix}=\begin{bmatrix}\boldsymbol{0}_N\\ \boldsymbol{e}_m\end{bmatrix} \tag{35}$$

where $\boldsymbol{D} = blockdiag(\boldsymbol{y}_1,\boldsymbol{y}_2,\cdots,\boldsymbol{y}_N)$, the positive definite matrix $\boldsymbol{H} = \boldsymbol{\Omega}+\frac{1}{C_2}\boldsymbol{I}_m+\frac{1}{C_1}\boldsymbol{B}\in\mathbb{R}^{m\times m}$, $\boldsymbol{\Omega}=\boldsymbol{Z}^T\boldsymbol{Z}\in\mathbb{R}^{m\times m}$ with $\boldsymbol{Z}=(\boldsymbol{Z}_1,\boldsymbol{Z}_2,\cdots,\boldsymbol{Z}_N)$, and $\boldsymbol{B}=blockdiag(\boldsymbol{\Omega}_1,\boldsymbol{\Omega}_2,\cdots,\boldsymbol{\Omega}_N)\in\mathbb{R}^{m\times m}$ with $\boldsymbol{\Omega}_k=\boldsymbol{Z}_k{}^T\boldsymbol{Z}_k\in\mathbb{R}^{m_k\times m_k}$.

The efficiency of MTL-LS-SVM I has been verified by comparing it with other several multi-task learning methods [22]. More details can be seen in [22].

MTL-LS-SVM II:

$$\begin{aligned}\min_{\boldsymbol{\omega}_0,\boldsymbol{v}_k,b_k,\boldsymbol{\zeta}_k}\quad & \frac{1}{2}\|\boldsymbol{\omega}_0\|^2+\frac{C_1}{2}\sum_{k=1}^{N}\|\boldsymbol{v}_k\|^2+\frac{C_2}{2}\sum_{k=1}^{N}\|\boldsymbol{\zeta}_k\|^2\\ \text{s.t.}\quad & \tilde{\boldsymbol{Z}}_k^T\boldsymbol{\omega}_0+\boldsymbol{A}_k^T\boldsymbol{v}_k+b_k\boldsymbol{y}_k=\boldsymbol{e}_{m_k}-\boldsymbol{\zeta}_k,\\ & k=1,2,\cdots,N\end{aligned} \tag{36}$$

The optimization problem (36) can be solved by the following linear system:

$$\begin{bmatrix} \mathbf{0}_{N\times N} & \boldsymbol{D}^T \\ \boldsymbol{D} & \tilde{\boldsymbol{H}} \end{bmatrix} \begin{bmatrix} \boldsymbol{b} \\ \boldsymbol{\alpha} \end{bmatrix} = \begin{bmatrix} \mathbf{0}_N \\ \boldsymbol{e}_m \end{bmatrix} \tag{37}$$

where $\boldsymbol{D} = blockdiag(\boldsymbol{y}_1, \boldsymbol{y}_2, \cdots, \boldsymbol{y}_N)$, the positive definite matrix $\boldsymbol{H} = \tilde{\boldsymbol{\Omega}} + \frac{1}{C_2}\boldsymbol{I}_m + \frac{1}{C_1}\tilde{\boldsymbol{B}} \in \mathbb{R}^{m\times m}$, $\tilde{\boldsymbol{\Omega}} = \tilde{\boldsymbol{Z}}^T\tilde{\boldsymbol{Z}} \in \mathbb{R}^{m\times m}$ with $\tilde{\boldsymbol{Z}} = (\tilde{\boldsymbol{Z}}_1, \tilde{\boldsymbol{Z}}_2, \cdots, \tilde{\boldsymbol{Z}}_N)$, and $\tilde{\boldsymbol{B}} = blockdiag(\boldsymbol{\Theta}_1, \boldsymbol{\Theta}_2, \cdots, \boldsymbol{\Theta}_N) \in \mathbb{R}^{m\times m}$ with $\boldsymbol{\Theta}_k = \boldsymbol{A}_k{}^T\boldsymbol{A}_k \in \mathbb{R}^{m_k\times m_k}$.

4 Experiments

To verify the effectiveness of the newly proposed multi-task learning methods, we conduct experiments to compare the newly proposed multi-task learning methods with the strategy that all of the N tasks are learned independently by employing aLS-SVM [31], L2-SVM [32], LS-SVM [25], and nonparallel least square SVM (NLSSVM) [29]. And the corresponding single task learning methods are denoted as N-aLS-SVM, N-L2-SVM, N-LS-SVM, and N-NLSSVM respectively. All the experiments are carried out in MATLAB R2014a on a personal computer (PC) with an Intel(R) Core(TM) i7 processor (3.40 GHz) and 4GB random access memory (RAM).

We test these methods on a collection of three benchmark datasets including Isolet, Monk, and Dermatology coming from the UCI Machine Learning Repository[1]. The Isolet dataset that is gathered from 150 subjects speaking 26 English letters twice consists of 7797 instances with 617 attributes (three instances had been historically lost). All of the speakers are divided into five equal number subsets known as Isolet1 to Isolet5, and each subset is treated as one classification task. On one hand, the five tasks have close relationship because they are gathered from the same utterances [11, 20]. On the other hand, the five tasks differ from each other because the speakers within diverse groups vary in the way of pronouncing the English letters. We classified three pairs of similar sounding letters including (B, D), (G, J) and (M, N) in our experiments. For (B, D) and (G, J) pairs, there are totally 600 instances in the five tasks of each pair; and for (M, N) pair, there are totally 599 instances in the five tasks. We employ principal component analysis (PCA) on the chosen datasets for removing the low variance noise. We reduce the attributes from 617 to 200, and 97.5% of the data variance is captured.

The Monk dataset with 432 instances is the basis of a first international comparison of learning algorithms. It is divided into 3 subsets based on the characteristic of 6 attributes. The subsets are referred to as Monk1, Monk2, and Monk3 which are corresponding to the three tasks.

The Dermatology dataset is a collection of 366 differential diagnosis including six kinds of dermatological diseases grounded on 33 clinicopathological characteristics. As in [8, 22], the problem can be converted into six binary one-versus-rest classification problems, and each one is regarded as a task. Therefore, we totally have six tasks.

In our experiments, the Gaussian kernel $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp(-\sigma\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2)$ is employed in the first multi-task learning method MTL-aLS-SVM I. For the second multi-task learning method MTL-aLS-SVM II, there exist two basic kernel functions: K_0 in the common model and K_k in the private model (27). We test our method with two different combinations: One is that $K_0(\boldsymbol{x}_{ki}, \boldsymbol{x}_{jr}) = \langle\boldsymbol{x}_{ki}, \boldsymbol{x}_{jr}\rangle$ is a linear kernel and $K_k(\boldsymbol{x}_{ki}, \boldsymbol{x}_{jr}) = \exp(-\sigma\|\boldsymbol{x}_{ki} - \boldsymbol{x}_{jr}\|^2)$ is a Gaussian kernel; The other combination is that $K_0(\boldsymbol{x}_{ki}, \boldsymbol{x}_{jr}) = \langle\boldsymbol{x}_{ki}, \boldsymbol{x}_{jr}\rangle$ is a linear kernel and $K_k(\boldsymbol{x}_{ki}, \boldsymbol{x}_{jr}) = (\langle\boldsymbol{x}_{ki}, \boldsymbol{x}_{jr}\rangle + 1)^d$ is a polynomial kernel with $d > 1$.

Generally speaking, the performance of the algorithms relies on the selections of parameters. There exist four (five) tuning parameters in MTL-aLS-SVM I (MTL-aLS-SVM II): C_1, C_2, σ, d (in MTL-aLS-SVM II), ρ. The first three (four) parameters are the same as those of MTL-L2-SVM I and MTL-LS-SVM I (MTL-L2-SVM II and MTL-LS-SVM II). The parameter ρ in MTL-aLS-SVM I and MTL-aLS-SVM II controls the sharp of the loss function. In our experiments, as in [31], we set $\rho = 0.99, 0.95, 0.83$. The parameter scopes are $C_1 \in \{2^{-7}, 2^{-5}, \cdots, 2^5\}$, $C_2 \in \{2^{-6}, 2^{-4}, \cdots, 2^8\}$, $\sigma \in \{2^{-7}, 2^{-5}, \cdots, 2^5\}$, and $d \in \{2, 3, \cdots, 9\}$. For the single task learning methods N-aLS-SVM, N-L2-SVM, and N-LS-SVM, except the kernel parameters σ, d, the optimal tuning parameter C is chosen from $\{2^{-6}, 2^{-4}, \cdots, 2^8\}$. For N-NLSSVM, except the kernel parameters σ, d, there are two tuning parameters c_1 and c_2 with the same ranges $\{2^{-7}, 2^{-6}, \cdots, 2^8\}$. For each dataset, the attributes are scaled in [-1,1]. About 55% of the instances are randomly chosen from the whole dataset to constitute the training set, and the rest is the testing set. The five-fold cross validation is used on the training dataset to find the optimal parameters, and then a classification accuracy on the testing set is obtained. Repeat the process ten times, and the "Accuracy" in the following tables is the mean value of ten times testing results.

In Tables 1, 2, 3, 4 and 5, "Accuracy±S" denotes the averaged classification accuracy plus or minus the standard deviation. "L", "G", and "P" represent Linear kernel, Gaussian kernel, and Polynomial kernel, respectively. In the second kind of multi-task learning methods, "L+G" represents that K_0 is the linear kernel function and K_k is the Gaussian kernel function; and "L+P" means that K_0 is the linear kernel function and K_k is the Polynomial kernel

[1] http://www.ics.uci.edu/~mlearn/MLRepository.html

Table 1 Experimental results on (B, D) pair of Isolet dataset

Single task learning method	Accuracy±S	Multi-task learning method	Accuracy±S
N-aLS-SVM ($\rho = 0.99$, L)	0.7674±0.0240	MTL-aLS-SVM II ($\rho = 0.99$, L+G)	0.9652±0.0070
N-aLS-SVM ($\rho = 0.99$, G)	0.7641±0.0230	MTL-aLS-SVM II ($\rho = 0.99$, L+P)	0.9652±0.0096
N-aLS-SVM ($\rho = 0.99$, P)	0.7148±0.0293	MTL-aLS-SVM I ($\rho = 0.99$, G)	0.9459±0.0129
N-aLS-SVM ($\rho = 0.95$, L)	0.7670±0.0251	MTL-aLS-SVM II ($\rho = 0.95$, L+G)	0.9644±0.0091
N-aLS-SVM ($\rho = 0.95$, G)	0.7648±0.0210	MTL-aLS-SVM II ($\rho = 0.95$, L+P)	**0.9656±0.0086**
N-aLS-SVM ($\rho = 0.95$, P)	0.7111±0.0313	MTL-aLS-SVM I ($\rho = 0.95$, G)	0.9633±0.0075
N-aLS-SVM ($\rho = 0.83$, L)	0.7663±0.0281	MTL-aLS-SVM II ($\rho = 0.83$, L+G)	0.9622±0.0085
N-aLS-SVM ($\rho = 0.83$, G)	0.7685±0.0191	MTL-aLS-SVM II ($\rho = 0.83$, L+P)	0.9644±0.0109
N-aLS-SVM ($\rho = 0.83$, P)	0.7070±0.0319	MTL-aLS-SVM I ($\rho = 0.83$, G)	0.9637±0.0109
N-L2-SVM (L)	0.7670±0.0250	MTL-L2-SVM II (L+G)	0.9581±0.0175
N-L2-SVM (G)	0.7663±0.0216	MTL-L2-SVM II (L+P)	0.9630±0.0105
N-L2-SVM (P)	0.7163±0.0331	MTL-L2-SVM I (G)	0.9622±0.0097
N-LS-SVM (L)	0.7681±0.0267	MTL-LS-SVM II (L+G)	0.9637±0.0147
N-LS-SVM (G)	0.7663±0.0216	MTL-LS-SVM II (L+P)	0.8211±0.0187
N-LS-SVM (P)	0.7156±0.0349	MTL-LS-SVM I (G)	0.9141±0.0130
N-NLSSVM (L)	0.5517±0.0513		
N-NLSSVM (G)	0.7596±0.0276		
N-NLSSVM (P)	0.6504±0.0475		

function. The best result among all the methods for each dataset is highlighted.

As can be seen from Tables 1, 2 and 3, for Isolet dataset, the accuracies of the multi-task learning methods are much higher than the single task learning methods in general. Specifically, for (B, D) pair, the highest accuracy was created by MTL-aLS-SVM II using the combination of the linear kernel and Polynomial kernel. For (G, J) pair, the multi-task learning method MTL-L2-SVM II with the linear kernel and Gaussian kernel combination achieved the best

Table 2 Experimental results on (G, J) pair of Isolet dataset

Single task learning method	Accuracy±S	Multi-task learning method	Accuracy±S
N-aLS-SVM ($\rho = 0.99$, L)	0.8578±0.0126	MTL-aLS-SVM II ($\rho = 0.99$, L+G)	0.9791±0.0069
N-aLS-SVM ($\rho = 0.99$, G)	0.8556±0.0136	MTL-aLS-SVM II ($\rho = 0.99$, L+P)	0.9637±0.0083
N-aLS-SVM ($\rho = 0.99$, P)	0.7567±0.0443	MTL-aLS-SVM I ($\rho = 0.99$, G)	0.9559±0.0082
N-aLS-SVM ($\rho = 0.95$, L)	0.8578±0.0139	MTL-aLS-SVM II ($\rho = 0.95$, L+G)	0.9819±0.0054
N-aLS-SVM ($\rho = 0.95$, G)	0.8548±0.0141	MTL-aLS-SVM II ($\rho = 0.95$, L+P)	0.9641±0.0087
N-aLS-SVM ($\rho = 0.95$, P)	0.7530±0.0328	MTL-aLS-SVM I ($\rho = 0.95$, G)	0.9793±0.0060
N-aLS-SVM ($\rho = 0.83$, L)	0.8544±0.0134	MTL-aLS-SVM II ($\rho = 0.83$, L+G)	0.9815±0.0068
N-aLS-SVM ($\rho = 0.83$, G)	0.8574±0.0127	MTL-aLS-SVM II ($\rho = 0.83$, L+P)	0.9637±0.0098
N-aLS-SVM ($\rho = 0.83$, P)	0.7504±0.0437	MTL-aLS-SVM I ($\rho = 0.83$, G)	0.9796±0.0058
N-L2-SVM (L)	0.8589±0.0120	MTL-L2-SVM II (L+G)	**0.9841±0.0058**
N-L2-SVM (G)	0.8557±0.0143	MTL-L2-SVM II (L+P)	0.9789±0.0080
N-L2-SVM (P)	0.7611±0.0508	MTL-L2-SVM I (G)	0.9778±0.0074
N-LS-SVM (L)	0.8574±0.0132	MTL-LS-SVM II (L+G)	0.9770±0.0072
N-LS-SVM (G)	0.8566±0.0147	MTL-LS-SVM II (L+P)	0.9007±0.0193
N-LS-SVM (P)	0.5622±0.0692	MTL-LS-SVM I (G)	0.9430±0.0273
N-NLSSVM (L)	0.5200±0.0205		
N-NLSSVM (G)	0.6229±0.0115		
N-NLSSVM (P)	0.5633±0.0189		

Table 3 Experimental results on (M,N) pair of Isolet dataset

Single task learning method	Accuracy±S	Multi-task learning method	Accuracy±S
N-aLS-SVM ($\rho = 0.99$, L)	0.6885±0.0291	MTL-aLS-SVM II ($\rho = 0.99$, L+G)	0.8669±0.0172
N-aLS-SVM ($\rho = 0.99$,G)	0.6788±0.0331	MTL-aLS-SVM II ($\rho = 0.99$, L+P)	0.8227±0.0305
N-aLS-SVM ($\rho = 0.99$, P)	0.6022±0.0138	MTL-aLS-SVM I ($\rho = 0.99$, G)	0.8468±0.0248
N-aLS-SVM ($\rho = 0.95$, L)	0.6881±0.0296	MTL-aLS-SVM II ($\rho = 0.95$, L+G)	0.8677±0.0209
N-aLS-SVM ($\rho = 0.95$, G)	0.6788±0.0294	MTL-aLS-SVM II ($\rho = 0.95$, L+P)	0.8234±0.0299
N-aLS-SVM ($\rho = 0.95$, P)	0.5955±0.0195	MTL-aLS-SVM I ($\rho = 0.95$, G)	0.8717±0.0142
N-aLS-SVM ($\rho = 0.83$, L)	0.6840±0.0295	MTL-aLS-SVM II ($\rho = 0.83$, L+G)	0.8725±0.0132
N-aLS-SVM ($\rho = 0.83$, G)	0.6762±0.0324	MTL-aLS-SVM II ($\rho = 0.83$, L+P)	0.8208±0.0284
N-aLS-SVM ($\rho = 0.83$, P)	0.5937±0.0228	MTL-aLS-SVM I ($\rho = 0.83$, G)	0.8725±0.0138
N-L2-SVM (L)	0.6855±0.0296	MTL-L2-SVM II (L+G)	0.8699±0.0185
N-L2-SVM (G)	0.6792±0.0337	MTL-L2-SVM II (L+P)	**0.8747±0.0148**
N-L2-SVM (P)	0.6033±0.0186	MTL-L2-SVM I (G)	0.8729±0.0168
N-LS-SVM (L)	0.6907±0.0267	MTL-LS-SVM II (L+G)	0.8706±0.0189
N-LS-SVM (G)	0.6810±0.0328	MTL-LS-SVM II (L+P)	0.7004±0.0207
N-LS-SVM (P)	0.5364±0.0254	MTL-LS-SVM I (G)	0.8138±0.0488
N-NLSSVM (L)	0.5067±0.0086		
N-NLSSVM (G)	0.7350±0.0126		
N-NLSSVM (P)	0.6000±0.0539		

accuracy. For (M, N) pair, the best accuracy was obtained by the multi-task learning method MTL-L2-SVM II with the linear kernel and Polynomial kernel combination.

For the Monk dataset, it is shown by Table 4 that the MTL-aLS-SVM I and MTL-aLS-SVM II achieve better performance than the other multi-task learning methods and the single task learning methods. And MTL-aLS-SVM II obtains the best accuracy among all of the multi-task and single task learning methods. In addition, it can be found that the performance of the single task learning methods has

Table 4 Experimental results on Monk dataset

Single task learning method	Accuracy±S	Multi-task learning method	Accuracy±S
N-aLS-SVM ($\rho = 0.99$, L)	0.7050±0.0133	MTL-aLS-SVM II ($\rho = 0.99$, L+G)	**0.8992±0.0177**
N-aLS-SVM ($\rho = 0.99$, G)	0.8725±0.0077	MTL-aLS-SVM II ($\rho = 0.99$, L+P)	0.8773±0.0294
N-aLS-SVM ($\rho = 0.99$, P)	0.8268±0.0142	MTL-aLS-SVM I ($\rho = 0.99$, G)	0.8896±0.0277
N-aLS-SVM ($\rho = 0.95$, L)	0.7060±0.0117	MTL-aLS-SVM II ($\rho = 0.95$, L+G)	0.8963±0.0141
N-aLS-SVM ($\rho = 0.95$, G)	0.8785±0.0114	MTL-aLS-SVM II ($\rho = 0.95$, L+P)	0.8830±0.0396
N-aLS-SVM ($\rho = 0.95$, P)	0.8237±0.0161	MTL-aLS-SVM I ($\rho = 0.95$, G)	0.8930±0.0124
N-aLS-SVM ($\rho = 0.83$, L)	0.7067±0.0128	MTL-aLS-SVM II ($\rho = 0.83$, L+G)	0.8942±0.0143
N-aLS-SVM ($\rho = 0.83$, G)	0.8887±0.0108	MTL-aLS-SVM II ($\rho = 0.83$, L+P)	0.8858±0.0227
N-aLS-SVM ($\rho = 0.83$, P)	0.8142±0.0137	MTL-aLS-SVM I ($\rho = 0.83$, G)	0.8935±0.0153
N-L2-SVM (L)	0.7058±0.0118	MTL-L2-SVM II (L+G)	0.8672±0.0347
N-L2-SVM (G)	0.8681±0.0200	MTL-L2-SVM II (L+P)	0.8613±0.0470
N-L2-SVM (P)	0.8560±0.0157	MTL-L2-SVM I (G)	0.8605±0.0385
N-LS-SVM (L)	0.7069±0.0114	MTL-LS-SVM II (L+G)	0.8022±0.0165
N-LS-SVM (G)	0.8851±0.0174	MTL-LS-SVM II (L+P)	0.8085±0.0210
N-LS-SVM (P)	0.8495±0.0206	MTL-LS-SVM I (G)	0.7925±0.0139
N-NLSSVM (L)	0.5818± 0.1113		
N-NLSSVM (G)	0.6815±0.0254		
N-NLSSVM (P)	0.6025±0.1426		

Table 5 Experimental results on Dermatology dataset

Single task learning method	Accuracy±S	Multi-task learning method	Accuracy±S
N-aLS-SVM ($\rho = 0.99$, L)	0.9676±0.0104	MTL-aLS-SVM II ($\rho = 0.99$, L+G)	0.9723±0.0061
N-aLS-SVM ($\rho = 0.99$, G)	0.9827±0.0044	MTL-aLS-SVM II $\rho = 0.99$, L+P)	0.9707±0.0043
N-aLS-SVM ($\rho = 0.99$, P)	0.8166±0.0165	MTL-aLS-SVM I ($\rho = 0.99$, G)	0.9720±0.0060
N-aLS-SVM ($\rho = 0.95$, L)	0.9744±0.0064	MTL-aLS-SVM II ($\rho = 0.95$, L+G)	0.9734±0.0066
N-aLS-SVM ($\rho = 0.95$, G)	0.9819±0.0039	MTL-aLS-SVM II ($\rho = 0.95$, L+P)	0.9664±0.0132
N-aLS-SVM ($\rho = 0.95$, P)	0.8727±0.0153	MTL-aLS-SVM I ($\rho = 0.95$, G)	0.9743±0.0065
N-aLS-SVM ($\rho = 0.83$, L)	0.9792±0.0052	MTL-aLS-SVM II ($\rho = 0.83$, L+G)	0.9774±0.0057
N-aLS-SVM ($\rho = 0.83$, G)	0.9804±0.0029	MTL-aLS-SVM II ($\rho = 0.83$, L+P)	0.9679±0.0088
N-aLS-SVM ($\rho = 0.83$, P)	0.9545±0.0144	MTL-aLS-SVM I ($\rho = 0.83$, G)	0.9772±0.0053
N-L2-SVM (L)	0.9805±0.0042	MTL-L2-SVM II (L+G)	0.9760±0.0081
N-L2-SVM (G)	**0.9833±0.0023**	MTL-L2-SVM II (L+P)	0.9558±0.0298
N-L2-SVM (P)	0.9815±0.0036	MTL-L2-SVM I (G)	0.9765±0.0086
N-LS-SVM (L)	0.9575±0.0065	MTL-LS-SVM II (L+G)	0.9643±0.0096
N-LS-SVM (G)	0.9818±0.0043	MTL-LS-SVM II (L+P)	0.7272±0.0869
N-LS-SVM (P)	0.9752±0.0045	MTL-LS-SVM I (G)	0.9527±0.0101
N-NLSSVM (L)	0.8343±0.0759		
N-NLSSVM (G)	0.8533±0.0657		
N-NLSSVM (P)	0.8424±0.0448		

much to do with the different choices of the kernel functions. However, the multi-task learning methods are less sensitive to kernel functions.

For the Dermatology dataset, it can be seen from Table 5 that the accuracies obtained by MTL-aLS-SVM I and MTL-aLS-SVM II are sightly lower than the highest accuracy obtained by the single task learning method L2-SVM. The same phenomenon occurs in [8] for MTL-FEAT (RBF) and independent (RBF). Argyrious et al. reinforce their conjecture by the numerical experiments that the relation among these tasks is weak or not [8]. As in [8, 22], the results in Table 5 indicate that the newly proposed multi-task learning methods can also achieve good performance even in such case.

In addition, it has been shown by Tables 1, 2, 3, 4 and 5 that the results obtained by our proposed multi-task learning methods are better than those reported by the multi-task RMM algorithm in [24].

Further, we employ the non-parametric Friedman test with its corresponding Nemenyi post-hoc test [36] to perform a more fair comparison of all the involved algorithms on the employed UCI datasets. For simplicity, only the best accuracy of each involved algorithm is under consideration. Table 6 reports the ranks of "Accuracy" of all the involved algorithms on the employed UCI datasets, where each algorithm is represented by its abbreviation, for example, "MTL-aL I" denotes "MTL-aLS-SVM I".

Let R_i denotes the average rank of the ith algorithm in Table 6, the Friedman statistic which is distributed according to $\mathcal{X}_F^2$ with $(K - 1)$ degrees of freedom and the $\mathcal{F}_F$ which is distributed according to the $\mathcal{F}$-distribution with $(K-1)$ and $(K-1)(N-1)$ degrees of freedom can be calculated as $\mathcal{X}_F^2 = \frac{12N}{K(K+1)}\left[\sum_{i=1}^{K} R_i^2 - \frac{K(K+1)^2}{4}\right] = 21.1418$ and $\mathcal{F}_F = \frac{(N-1)\mathcal{X}_F^2}{N(K-1)-\mathcal{X}_F^2} = 3.5446$, where $N = 5$, K=10. According to the table of critical values, it is easy to know that $\mathcal{F}_{\alpha=0.1}(10, 5) = 1.811 < 3.5446$, so we reject the null hypothesis. For further pairwise comparison, we resort to

Table 6 The ranks of the involved algorithms in the Friedman test on the employed UCI datasets

Dataset	N-aL	N-L2	N-LS	N-NL	MTL-L2 I	MTL-LS I	MTL-aL I	MTL-aL II	MTL-L2 II	MTL-LS II
(B, D)	7	9	8	10	5	6	2.5	1	4	2.5
(G, J)	8	7	9	10	4	6	3	2	1	5
(M,N)	9	10	8	7	2	6	3.5	3.5	1	5
Monk	3	5	4	10	7	9	2	1	6	8
Dermatology	2	1	3	10	6	9	5	4	7	8
Average ranks	5.8	6.4	6.4	9.4	4.8	7.2	3.2	2.3	3.8	5.7

the Nemenyi post-hoc test. For $\alpha = 0.1$, the critical difference $CD = \mathcal{F}_{\alpha=0.1}(10, 5) * \sqrt{\frac{K(K+1)}{6N}} = 3.4678$. It is well known that the performance of two algorithms is significantly different if their average ranks differ by at least the critical difference. Based on Table 6, the differences between MTL-aL II and other algorithms can be calculated as follows:

$$d(\text{N-aL}) - d(\text{MTL-aL II}) = 5.8 - 2.3 = 3.5 > 3.4678$$
$$d(\text{N-L2}) - d(\text{MTL-aL II}) = 6.4 - 2.3 = 4.1 > 3.4678$$
$$d(\text{N-LS}) - d(\text{MTL-aL II}) = 6.4 - 2.3 = 4.1 > 3.4678$$
$$d(\text{N-NL}) - d(\text{MTL-aL II}) = 9.4 - 2.3 = 7.1 > 3.4678$$
$$d(\text{MTL-L2 I}) - d(\text{MTL-aL II}) = 4.8 - 2.3 = 2.5 < 3.4678$$
$$d(\text{MTL-LS I}) - d(\text{MTL-aL II}) = 7.2 - 2.3 = 4.9 > 3.4678$$
$$d(\text{MTL-aL I}) - d(\text{MTL-aL II}) = 3.2 - 2.3 = 0.9 < 3.4678$$
$$d(\text{MTL-L2 II}) - d(\text{ MTL-aL II}) = 3.8 - 2.3 = 1.5 < 3.4678$$
$$d(\text{MTL-LS II}) - d(\text{MTL-aL II}) = 5.7 - 2.3 = 3.4 < 3.4678$$

where $d(\text{a} - \text{b})$ denotes the differences between a and b. Then we obtain the following conclusion: on the employed UCI datasets, MTL-aLS-SVM II performs significantly better than all the single task learning methods including N-aLS-SVM, N-L2-SVM, N-LS-SVM, N-NLSSVM and the multi-task learning method MTL-LS-SVM I, and there is no significant differences between MTL-aLS-SVM II and MTL-aLS-SVM I, MTL-L2-SVM I, MTL-L2-SVM II, MTL-LS-SVM II.

In the next part of the experiments, we demonstrate the influence of the parameter C_1 in multi-task learning methods MTL-aLS-SVM I and MTL-aLS-SVM II (formulations (4) and (20)) which trades off the public classification information and the dissimilarity between tasks. For this purpose, we contradistinguish MTL-aLS-SVM I, MTL-aLS-SVM II including MTL-aLS-SVM II (L+G) and MTL-aLS-SVM II (L+P), N-aLS-SVM, and 1-aLS-SVM (the method that employs one aLS-SVM for all tasks when all tasks are

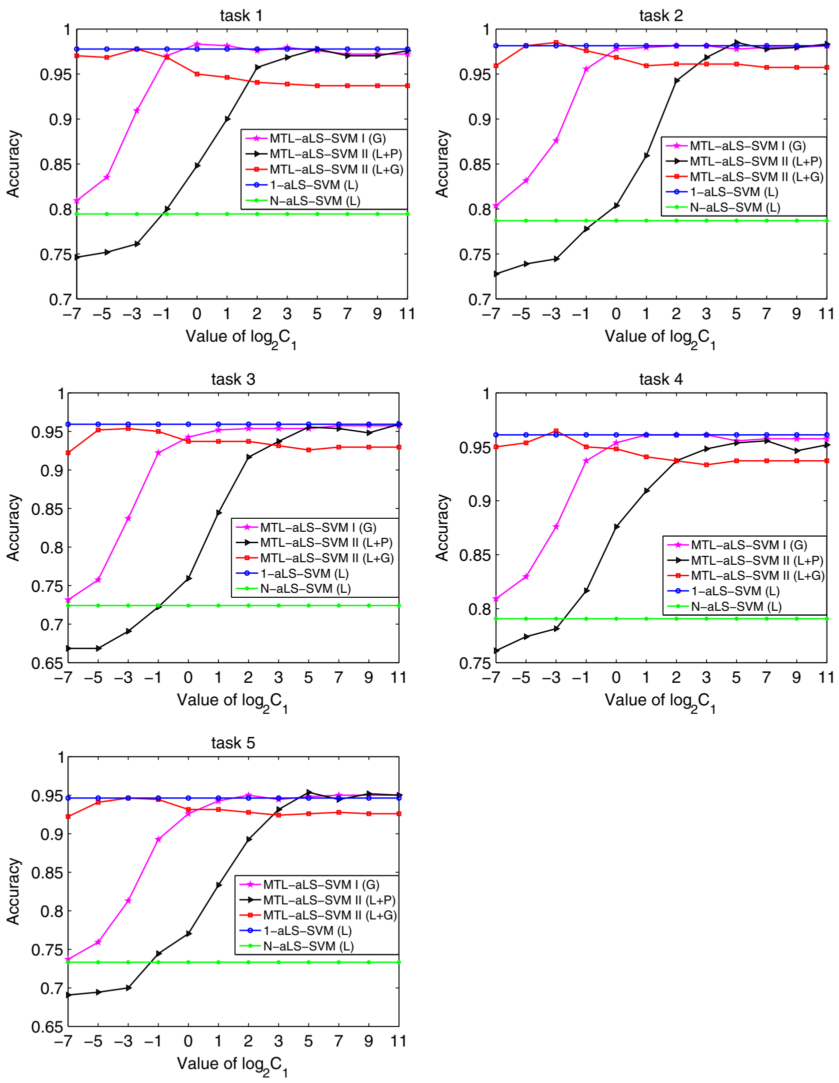


Fig. 1 Accuracy variations of MTL-aLS-SVM I and MTL-aLS-SVM II along with C_1; and the comparison algorithms N-aLS-SVM and 1-aLS-SVM use the Linear kernel

regarded as one big task). We take $\rho = 0.95$ as an example, and conduct the experiments on the (B, D) pair of Isolet dataset. The variations of the averaged accuracy of the multi-task learning methods on each task along with the values of C_1 are illustrated in the Figs. 1, 2 and 3. As comparisons, the averaged accuracy obtained by the single learning methods N-aLS-SVM and 1-aLS-SVM with linear kernel, polynomial kernel, and Gaussian kernel are also illustrated in the Figs. 1, 2 and 3, respectively. Note that the N-aLS-SVM and 1-aLS-SVM models do not contain parameter C_1, the averaged accuracy of these two models are not affected by the variation of parameter C_1. The "Accuracy" in the three figures denotes the averaged accuracy.

It is shown by the three figures that when the values of C_1 are small, the accuracies of MTL-aLS-SVM I and MTL-aLS-SVM II (L+P) are close to those of the conventional independent learning strategy N-aLS-SVM. When the values of C_1 are large, the performance of MTL-aLS-SVM I and MTL-aLS-SVM II (L+P) is in line with that of 1-aLS-SVM. However, the variation of the accuracies of MTL-aLS-SVM II (L+G) is not distinct, and MTL-aLS-SVM II (L+G) keeps the good performance along with the values of C_1.

In addition, it is interesting to see that the averaged accuracies of N-aLS-SVM are always lower than those of 1-aLS-SVM. The reason for this is that a small number of training data provides less information for N-aLS-SVM. And 1-aLS-SVM cannot deal with the label-incompatible dataset (for example, Monk dataset and Dermatology dataset). However, MTL-aLS-SVM I and MTL-aLS-SVM II can obtain the good performance with the proper values of C_1

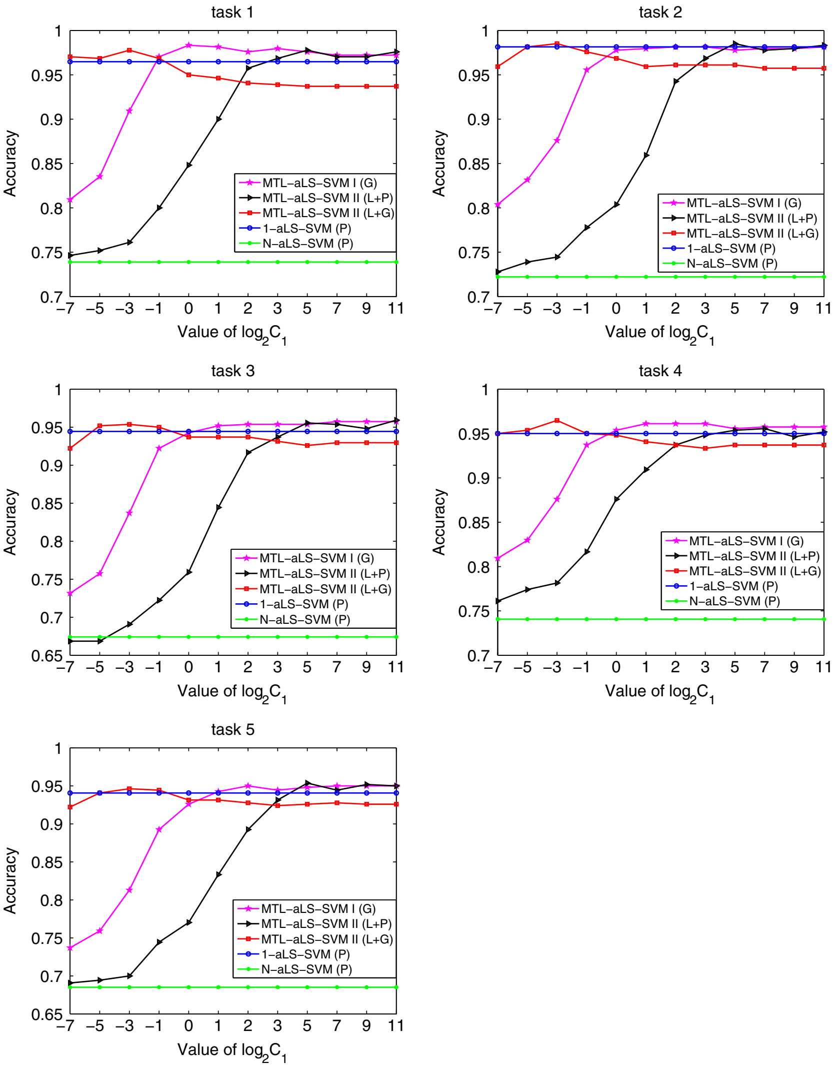
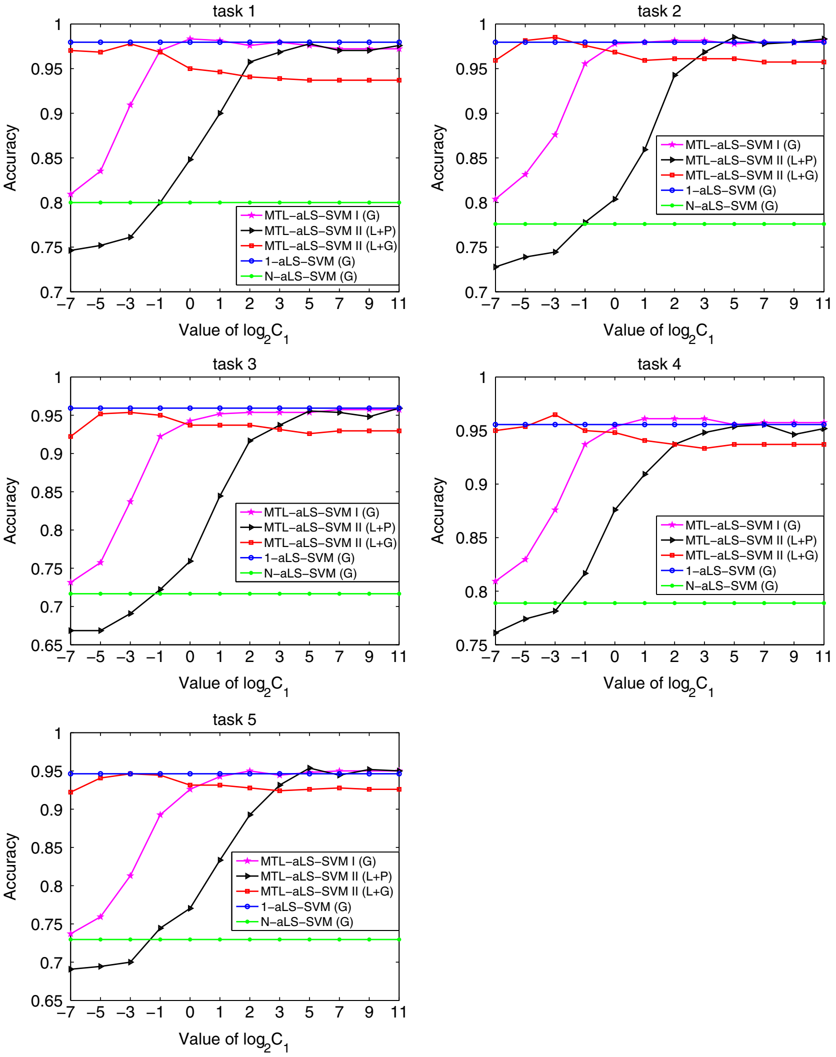


Fig. 2 Accuracy variations of MTL-aLS-SVM I and MTL-aLS-SVM II along with C_1; and the comparison algorithms N-aLS-SVM and 1-aLS-SVM use the Polynomial kernel

Fig. 3 Accuracy variations of MTL-aLS-SVM I and MTL-aLS-SVM II along with C_1; and the comparison algorithms N-aLS-SVM and 1-aLS-SVM use the Gaussian kernel



since they can potentially learn correlation between tasks leading to more information.

5 Conclusion

In this paper, we have proposed the multi-task learning methods MTL-aLS-SVM I, MTL-aLS-SVM II, and their special cases for binary classification. MTL-aLS-SVM I combines the advantages of multi-task learning and the asymmetric least squares support vector machines. MTL-aLS-SVM II is an extension of the MTL-aLS-SVM I which adopt the assumption that the models of related tasks share a common model. A regularization parameter C_1 is introduced in MTL-aLS-SVM I and MTL-aLS-SVM II to seek for a trade-off between the public information and the private information dedicated to some specific task. In addition, the special cases MTL-L2-SVM II and MTL-LS-SVM II are also the newly proposed multi-task learning methods, which exhibit good performance. We have conducted comprehensive experiments to test the performance of the newly proposed methods and the influence of the regularization parameter C_1. Experimental results have shown that our methods are more effective than the corresponding single task learning methods. Additionally, our methods are flexible due to the introduction of parameter C_1. When there exists relatedness among the tasks, a proper value of C_1 can be selected to make the methods achieve good performance. On the other hand, if the tasks are independent, a small value of C_1 will make the methods learn tasks independently.

The multi-task learning is mainly designed to explore the latent information by learning all tasks jointly. As for exploiting the underlying information to improve the traditional inductive learning, an other renewed interest approach is Learning Using Privileged Information (LUPI) [37, 38]. Our future work is to extend our proposed multi-task learning methods to the LUPI learning paradigm.

Acknowledgments The work is supported by the National Natural Science Foundation of China (No.11171346) and Chinese Universities Scientific Fund No. 2017LX003.

Appendix: The proof of (12)

Substituting (11) into the objective function of (4), we have

$$
\begin{aligned}
&\frac{1}{2}\|\boldsymbol{\omega}_0\|^2+\frac{C_1}{2}\sum_{k=1}^{N}\|\boldsymbol{v}_k\|^2+\frac{C_2}{2}\sum_{k=1}^{N}\|\boldsymbol{\zeta}_k\|^2\\
&=\frac{1}{2}\|\boldsymbol{\omega}_0\|^2+\frac{C_1}{2}\sum_{k=1}^{N}\|\boldsymbol{\omega}_k-\boldsymbol{\omega}_0\|^2+\frac{C_2}{2}\sum_{k=1}^{N}\|\boldsymbol{\zeta}_k\|^2\\
&=\frac{1}{2}\left\|\frac{C_1N}{1+C_1N}\cdot\frac{1}{N}\sum_{k=1}^{N}\boldsymbol{\omega}_k\right\|^2+\frac{C_1}{2}\sum_{k=1}^{N}\Bigg\|\boldsymbol{\omega}_k\\
&\quad-\frac{C_1N}{1+C_1N}\cdot\frac{1}{N}\sum_{t=1}^{N}\boldsymbol{\omega}_t\Bigg\|^2+\frac{C_2}{2}\sum_{k=1}^{N}\|\boldsymbol{\zeta}_k\|^2\\
&=\frac{\tau_1^2N^2}{2}\|\bar{\boldsymbol{\omega}}\|^2+\frac{C_1}{2}\sum_{k=1}^{N}\|\boldsymbol{\omega}_k-\tau_1N\bar{\boldsymbol{\omega}}\|^2+\frac{C_2}{2}\sum_{k=1}^{N}\|\boldsymbol{\zeta}_k\|^2
\end{aligned}
$$

where $\bar{\boldsymbol{\omega}}=\frac{1}{N}\sum_{t=1}^{N}\boldsymbol{\omega}_t$, $\tau_1=\frac{C_1}{1+C_1N}$, $\tau_2=\frac{C_1^2N}{1+C_1N}$. Noticing that $\tau_1+\tau_2=C_1$, $\tau_2=\tau_1C_1N$, and $\tau_2=(1+C_1N)\tau_1^2N$, the above equation can be calculated as follows.

$$
\begin{aligned}
&\frac{\tau_1^2N^2}{2}\|\bar{\boldsymbol{\omega}}\|^2+\frac{C_1}{2}\sum_{k=1}^{N}\|\boldsymbol{\omega}_k-\tau_1N\bar{\boldsymbol{\omega}}\|^2+\frac{C_2}{2}\sum_{k=1}^{N}\|\boldsymbol{\zeta}_k\|^2\\
&=\frac{C_1}{2}\sum_{k=1}^{N}\|\boldsymbol{\omega}_k\|^2-\tau_1C_1N\sum_{k=1}^{N}{\boldsymbol{\omega}_k}^T\bar{\boldsymbol{\omega}}\\
&\quad+\frac{(1+C_1N)\tau_1^2N^2}{2}\|\bar{\boldsymbol{\omega}}\|^2+\frac{C_2}{2}\sum_{k=1}^{N}\|\boldsymbol{\zeta}_k\|^2\\
&=\frac{1}{2}\left((\tau_1+\tau_2)\sum_{k=1}^{N}\|\boldsymbol{\omega}_k\|^2-2\tau_2\sum_{k=1}^{N}\boldsymbol{\omega}_k^T\bar{\boldsymbol{\omega}}+\tau_2N\|\bar{\boldsymbol{\omega}}\|\right)\\
&\quad+\frac{C_2}{2}\sum_{k=1}^{N}\|\boldsymbol{\zeta}_k\|^2\\
&=\frac{\tau_1}{2}\sum_{k=1}^{N}\|\boldsymbol{\omega}_k\|^2+\frac{\tau_2}{2}\sum_{k=1}^{N}\|\boldsymbol{\omega}_k-\bar{\boldsymbol{\omega}}\|^2+\frac{C_2}{2}\sum_{k=1}^{N}\|\boldsymbol{\zeta}_k\|^2
\end{aligned}
$$

Therefore, the proof of (12) is completed.

References

1. Bakker B, Heskes T (2003) Task clustering and gating for Bayesian multitask learning. J Mach Learn Res 4:83–99
2. Ando RK, Zhang T (2005) A framework for learning predictive structures from multiple tasks and unlabeled data. J Mach Learn Res 6:1817–1953
3. Bi J, Xiong T, Yu S, Dundar M, Rao RB (2008) An improved multi-task learning approach with applications in medical diagnosis. In: Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Datasets–Part I, Antwerp, Belgium, pp 117–132
4. Birlutiu A, Groot P, Heskes T (2010) Multi-task preference learning with an application to hearing aid personalization. Neurocomputing 73:1177–1185
5. Chapelle O, Shivaswamy P, Vadrevu S, Weinberger K, Zhang Y, Tseng B (2010) Multi-task learning for boosting with application to web search ranking. In: Proceedings of the 16th ACM SIGKDD international conference on knowledge discovery and data mining. ACM, New York, pp 1189–1198
6. Ren Y, Xu B, Zhu P (2016) A multiCell visual tracking algorithm using multi-task paticle swarm optimization for low-constrast image seqences. Appl Intell 45(4):1129–1147
7. Caruana R (1997) Multitask learning. Mach Learn 28(1):41–75
8. Argyriou A, Evgeniou T, Pontil M (2008) Convex multi-task feature learning. Mach Learn 73(3):243–272
9. Ben-David S, Schuller R (2003) Exploiting task relatedness for multiple task learning. In: Proceedings of the 16th Annual Conference on Computational Learning Theory and the 7th Kernel Workshop, Washington DC, pp 567–580
10. Ben-David S, Borbely RS (2008) A notion of task relatedness yielding provable multiple-task learning guarantees. Mach Learn 73(3):273–287
11. Parameswaran S, Weinberger KQ (2000) Large margin multi-task metric learning. Adv Neural Inf Process Syst 23:1867–1875
12. Evgeniou T, Pontil M (2004) Regularized multi-task learning. In: Tenth ACM SIGKDD International Conference on Knowledge discovery and data mining, Seattle, pp 109–117
13. Evgeniou T, Micchelli CA, Pontil M (2005) Learning multiple tasks with kernel methods. J Mach Learn Res 6:615–637
14. Li X, Zhao L, Wei L, Yang MH, Wu F, Zhuang Y, Ling H, Wang J (2016) DeepSaliency: Multi-task deep neural network model for salient object detection. IEEE Trans Image Process Publ IEEE Signal Process Soc 25(8):3919–3930
15. Yan Y, Ricci E, Subramanian R, Liu G, Lanz O, Sebe N (2016) A multi-task learning framework for head pose estimation under target motion. IEEE Trans Pattern Anal Mach Intell 38(6):1070–1083
16. Kato T, Kashima H, Sugiyama M, Asai K (2008) Multi-task learning via conic programming. In: Advances in Neural Information Processing Systems 20. MIT Press, Cambridge, pp 737–744
17. Yang H, King I, Lyu MR (2010) Multi-task learning for one-class classification. In: Proceedings of the International Joint Conference on Neural Networks, Barcelona, pp 1–8
18. He X, Mourot G, Maquin D, Ragot J, Beauseroy P, Smolarz A, Grall-Maes E (2011) One-class SVM in multi-task learning. In: Advances in Safety, Reliability and Risk Management. ESREL 2011, Troyes, pp 486–494
19. He X, Mourot G, Maquin D, Ragot J, Beauseroy P, Smolarz A, Grall-Maes E (2014) Multi-task learning with one-class SVM. Neurocomputing 133:416–426
20. Ji Y, Sun S (2013) Multitask multiclass support vector machines: model and experiments. Pattern Recogn 46(3):914–924
21. Ji Y, Sun S, Lu Y (2012) Multitask multiclass privileged information support vector machines. In: Proceedings of the twenty-first international conference on pattern recognition, pp 2323–2326
22. Xu S, An X, Qiao X, Zhu L (2014) Multi-task least-squares support vector machines. Multimed Tools Appl 71(2):699–715

23. Li Y, Tian X, Song M, Song MG, Tao DC (2015) Multi-task proximal support vector machine. Pattern Recogn 48(10):3249–3257
24. Song YY, Zhu WX (2016) Multi-task support vector machine for data classification. Image Process Pattern Recogn 9(7):341–350
25. Suykens JAK, Van Gestel T, De Brabanter J, De Moor B, Vandewalle J (2002) Least squares support vector machines. World Scientific, Singapore
26. Maldonado S, López J (2017) Robust kernel-based multiclass support vector machines via second-order cone programming. Appl Intell 46:983–992
27. Le Thi HA, Pham Dinh T, Thiao M (2016) Efficient approaches for l_2-l_0 regularization and applications to feature selection in SVM. Appl Intell 45:549–565
28. Li C, Zhang Y, Lu L (2015) An MIMLSVM algorithm based on ECC. Appl Intell 42:537–543
29. Zhao J, Yang Z, Xu Y (2016) Nonparallel least square support vector machine for classification. Appl Intell 45:1119–1128
30. Huang X, Shi L, Suykens JAK (2014) Support vector machine classifier with pinball loss. IEEE Trans Pattern Anal Mach Intell 36(5):984–997
31. Huang X, Shi L, Suykens JAK (2014) Asymmetric least squares support vector machine classifiers. Comput Stat Data Anal 70: 395–405
32. Vapnik V (1995) The nature of statistical learning theory. Springer-Verlag, New York
33. Wang KN, Zhu WX, Zhong P (2015) Robust support vector regression with generalized Loss Function and Applications. Neural Process Lett 41:89–106
34. Wang KN, Zhong P (2014) Robust non-convex least squares loss function for regression with outliers. Knowl-Based Syst 71:290–302
35. Zhong P (2012) Training robust support vector regression with smooth non-convex loss function. Optim Methods Softw 27(6):1039–1058
36. Demsar J (2006) Statistical comparisons of classifiers over multiple data sets. J Mach Learn Res 7:1–30
37. Vapnik V, Vashist A (2009) A new learning paradigm: learning using privileged information. Neural Netw 22(5):544–557
38. Zhu WX, Zhong P (2014) A new one-class SVM based on hidden information. Knowl-Based Syst 60:35–43