Pattern Recognition 48 (2015) 3249–3257

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

Multi-task proximal support vector machine ☆

Ya Li [a,1], Xinmei Tian [a,*], Mingli Song [b], Dacheng Tao [c]

[a] University of Science and Technology of China, Hefei, Anhui 230026, PR China
[b] Zhejiang University, Hangzhou, Zhejiang, China
[c] University of Technology, Sydney, Australia

ARTICLE INFO

Article history:
Received 17 October 2014
Received in revised form
15 January 2015
Accepted 16 January 2015
Available online 3 February 2015

Keywords:
Multi-task learning
Support vector machines
Proximal classifiers

ABSTRACT

With the explosive growth of the use of imagery, visual recognition plays an important role in many applications and attracts increasing research attention. Given several related tasks, single-task learning learns each task separately and ignores the relationships among these tasks. Different from single-task learning, multi-task learning can explore more information to learn all tasks jointly by using relationships among these tasks. In this paper, we propose a novel multi-task learning model based on the proximal support vector machine. The proximal support vector machine uses the large-margin idea as does the standard support vector machines but with looser constraints and much lower computational cost. Our multi-task proximal support vector machine inherits the merits of the proximal support vector machine and achieves better performance compared with other popular multi-task learning models. Experiments are conducted on several multi-task learning datasets, including two classification datasets and one regression dataset. All results demonstrate the effectiveness and efficiency of our proposed multi-task proximal support vector machine.

1. Introduction

Given the explosive growth the use of imagery in the era of big data, visual recognition has become an important problem. Various image classification and recognition methods have been proposed and have achieved much success [1–9]. Some feature learning methods are also proposed to improve the performance of image classification and recognition [10–13]. When learning a visual recognition task, it can often be viewed as a combination of multiple correlated subtasks [14]. Considering multi-label image classification, for example, one particular image may contain multiple objects corresponding to different labels. Obviously, there are correlations among these labels. Traditional single-task learning methods, for example, SVMs and Bayesian models, learn to classify these labels separately and ignore correlations among them. It would be desirable to explore shared information across different subtasks and apply the information to learn all the subtasks jointly. Inspired by this idea, various methods are proposed to learn multiple tasks jointly rather than separately. This is often called the multi-task learning (MTL) [15], learning to learn [16] or inductive bias learning [17]. All these methods tend to learn multiple tasks together and improve the performance of single-task learning models.

The most important and difficult problem in multi-task learning is to discover the shared information among tasks and maintain the independence of each task. Considering the classification of vehicles (see Fig. 1), we have various types of vehicles, such as sports cars, family cars and buses corresponding to different classification tasks. These cars have shared features as well as unique characteristics. For example, all cars have four wheels and two headlights. However, sports cars usually have a lower and racing body, family cars often have medium size, and buses have a bigger body. Single-task learning only uses the information of the independent task, while multi-task learning will use all the information among the tasks. If a multi-task learning method can find the shared features of these vehicles and distinguish differences among the vehicles, each learning task will have much more additional information from other tasks. Conversely, noise will be added to the current learning task.

Existing multi-task learning methods mainly have two ways to discover relationships among different tasks. One way is to assume that different tasks share common parameters [18,14,19–23] such as a Bayesian model sharing a common prior [14] or a

☆This work is supported by the NSFC under the Contract nos. 61201413 and 61390514, the Fundamental Research Funds for the Central Universities Nos. WK2100060011 and WK2100100021, and the Specialized Research Fund for the Doctoral Program of Higher Education No. WJ2100060003. Australian Research Council Projects: DP-140102164, ARC FT-130101457, and ARC LP-140100569.

* Corresponding author. Tel.: +86 183 551 026 90; fax: +86 551 636 013 40
E-mail addresses: muziyiye@mail.ustc.edu.cn (Y. Li), xinmei@ustc.edu.cn (X. Tian), brooksong@ieee.org (M. Song), Dacheng.Tao@uts.edu.au (D. Tao).
[1] CAS Key Laboratory of Technology in Geo-spatial Information Processing and Application System, University of Science and Technology of China, China

http://dx.doi.org/10.1016/j.patcog.2015.01.014
0031-3203/

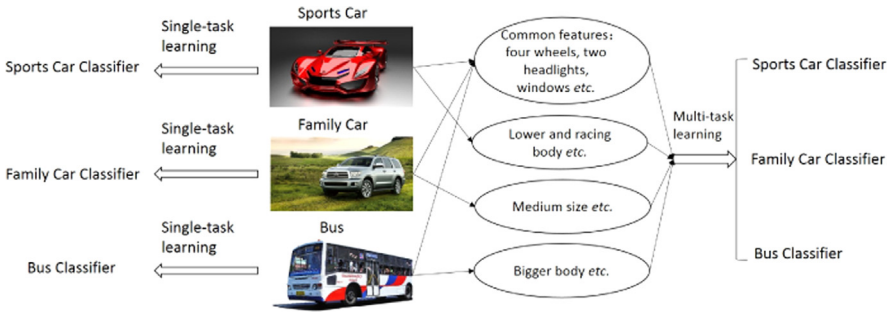

Fig. 1. An example of single-task learning comparing with multi-task learning.

large-margin model sharing a mean hyperplane [19]. The other way to learn the relatedness is to find latent feature representation among these tasks [24–26], for example, learning a sparse representation shared across tasks [25]. Existing multi-task learning methods mainly have two defects. First, some multi-task learning models have a complicated theoretical foundation, which leads to implementation difficulties. For example, a nonparametric Bayesian model usually has many assumptions and many parameters to select. Second, the efficiency is low, especially when the dataset has a large number of data points and a high dimensional feature. Our goal is to find an easily implemented multi-task learning method with high efficiency and comparable performance. In this paper, we propose a multi-task learning method based on the proximal support vector machine (PSVM) [27] and apply it to two classification datasets and one regression dataset. PSVM was proposed by Fung and Mangasarian and is different from the standard SVM [28]. PSVM also utilizes the large margin idea by assigning the data points to the closest of two disjoint hyperplanes, which are separated as far as possible. However, PSVM has looser constraints than does standard SVM, with comparable performance and much lower computational cost. Inspired by the idea of PSVM and the advantages of multi-task learning, we derive a multi-task proximal support vector machine (MTPSVM). All data examples of all tasks are needed to learn MTPSVM simultaneously. It will absolutely slow the computing process if the dataset is a large-scale one. In this paper, we develop a method to optimize the procedure of learning MTPSVM that greatly improves efficiency. Based on the idea of PSVM for unbalanced data, we also apply this to MTPSVM. Finally, we propose proximal support vector regression for regression problems, which is not discussed in PSVM [27], and extend it to multi-task problems.

MTPSVM has two primary merits compared with other multi-task learning methods. First, MTPSVM is easily implemented by just solving a quadratic optimization problem with equality constraints. Second, MTPSVM has much lower computational cost and can be applied to a large-scale dataset. We will demonstrate that the computational time of MTPSVM relies primarily on the feature dimension of the data rather than on the number of data points.

We organize the remainder of this paper as follows. Section 2 reviews previous works in multi-task learning. In Section 3, we first briefly introduce the proximal support vector machine and then give a specific derivation of the proposed multi-task proximal support vector machine. The derivation of multi-task proximal support vector regression will be presented in Section 4. In Section 5, experiments on several datasets are presented. Section 6 presents our study's conclusions.

2. Related work

Multi-task learning has been proven more effective than single-task learning by many works via both theory analysis and extensive experiments. For example, Baxter proposed a novel model of inductive bias learning to learn multiple tasks together and derived explicit bounds which demonstrated that multi-task learning gave better generalization than single-task learning [17]. Another work conducted by Ben-David and Schuller developed a useful notion of task relatedness and better generalization of error bounds for learning multiple related tasks based on one special type of relatedness of tasks [29]. Both studies prove the merits of multi-task learning in theory. Various experiments also demonstrate that multi-task learning can achieve better performance than can single-task learning, e.g., experiments on School Dataset [19,30,25,31], Landmine Dataset [14,24]. Multi-task learning can achieve much better performance than single-task learning especially when the amount of training data is limited.

Due to the effectiveness of multi-task learning, many single-task learning methods are extended to multi-task learning ones, such as neural networks, nearest neighbor learners, Bayesian model and SVM. For example, multi-task learning methods are implemented by sharing hidden nodes in neural networks or using nearest neighbor learners [15,32]. Bayesian is another popular model for multitask learning. It assumes dependencies between various models and tasks [33,34]. Models can be learned by hierarchical Bayesian inference with shared parameters treated as hyperparameters at a higher level than the single-task model parameters. In recent years, nonparametric Bayesian models and infinite latent subspace learning have become popular in multi-task learning. Rai and Daume proposed an infinite latent feature model to automatically infer the dimensionality of the task subspace. They learned a multi-task learning model using the Indian Buffet Process as the nonparametric Bayesian prior [18]. Consider the success of SVM in single-task learning, support vector machines are popular in multi-task learning. Many multi-task learning methods are developed based on support vector machines with different assumptions or priors [35,19,30,24]. An infinite latent SVM for multi-task learning is derived using nonparametric Bayesian models with regularization on the desired posterior distributions [35]. Evgeniou and Pontil proposed a novel multi-task learning method based on the minimization of regularization functions, similar to support vector machines [19]. Based on the work of [19], a more specific and general derivation of kernel method was developed in [30]. Jebara proposed a maximum entropy discrimination method for multi-task learning based on the large-margin support vector machines [24]. It gives extensions of feature selection and kernel selection for multi-task learning. The idea of our multi-task learning method is similar to [19]. The difference is that our multi-task learning method is based on proximal support vector machine rather than on the standard support vector machines. This results in an easier implementation and lower computational cost.

As mentioned above, learning latent common features across tasks and sharing common parameters are two important ways to model the relatedness of multi-task learning. For learning latent common features, a framework was proposed to learn sparse

representations shared across multiple tasks [25]. It is based on a well-known single-task Ll-norm regularization and presents a novel non-convex regularizer that controls the number of learned features common across all tasks. A summary of feature selection and kernel selection was given by Jebara [24]. It combines feature selection and kernel selection via the support vector machines. Recently, Maksim et al. proposed a novel multi-task learning method to learn a low-dimensional representation jointly with corresponding classifiers. This scalable multi-task representation learning method is suitable for high-dimensional features [36]. Recent works point out that we need to consider whether all the tasks are related and share a common set of features. If not, learning jointly with outlier tasks will result in worse performance. Jalali et al. introduced an extra ℓ_1/ℓ_q-norm regularization term individually for feature selection [37]. Gong et al. applied a similar idea to learn more robust multi-task feature [38]. Another robust multi-task learning method is proposed to capture the task relationship using a low-rank structure and to identify the outlier tasks using group-sparse structure [39]. As for sharing the common parameters, Theodoros and Massimiliano applied multi-task learning to the support vector machines and assumed that related SVM classifiers share a common hyperplane [19]. The underlying assumption is that models of all tasks are close to one common model with a small offset. Rai and Daume assumed that task parameters shared a latent subspace, which was similar to factor analysis, to measure the relatedness of the tasks. For other works, [20,21] measured task relatedness through Frobenius norms of their difference and [14,22,23] learnt the correlation among tasks through a common prior.

3. Multi-task proximal support vector machine

In this section, we first give an overview of the proximal support vector machines and then introduce the detailed theoretical derivation of our proposed MTPSVM. Additionally, computing optimization details will be given in section 3.4.

3.1. Linear proximal support vector machine

Consider a classification problem, we have a dataset $\mathcal{D}$ including m data points in an n-dimensional real space $\mathbb{R}^n$. It can be represented by an $m \times n$ matrix A. Each data point, $x_i \in \mathbb{R}^n$, has a binary output $y_i \in \{\pm 1\}$ (considered as a binary classification problem), thus $\mathcal{D} = \{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\}$. We use an $m \times m$ diagonal matrix D with plus one or minus one along its diagonal to represent label information of the m data points, $D(i,i) = y_i$ and e is an $m \times 1$ vector of ones. To solve this problem, standard support vector machines with a linear kernel is given as the following quadratic program with penalty parameter υ with respect to ξ

$$\begin{aligned} \min_{(w,\gamma,\xi)\in\mathbb{R}^{(m+n+1)}} \quad & \upsilon e'\xi + \frac{1}{2}w'w \\ \text{s.t.} \quad & D(Aw - e\gamma) + \xi \geq e \\ & \xi \geq 0 \end{aligned} \tag{1}$$

The nonnegative variable ξ determines the error when the classes are linearly inseparable. Different from standard SVM, PSVM solves the following quadratic problem. It replaces the inequality constraints with an equality one and adds the penalty of γ

$$\begin{aligned} \min_{(w,\gamma,\xi)\in\mathbb{R}^{(m+n+1)}} \quad & \frac{1}{2}\upsilon\xi'\xi + \frac{1}{2}(w'w + \gamma^2) \\ \text{s.t.} \quad & D(Aw - e\gamma) + \xi = e \end{aligned} \tag{2}$$

Although the modification is very simple, it changes the optimization problem significantly. The planes $x'w - \gamma = \pm 1$ are not bounding planes anymore but can be regarded as proximal planes, around which the points of each class are clustered and which are pushed as far apart as possible by the term $\frac{1}{2}(w'w + \gamma)$ [27]. It is easy to derive an explicit solution for problem (2) while it is impossible to do so in the standard support vector machines problem. As a result, PSVM can greatly improve efficiency compared to standard SVMs. More details of proximal support vector machine can be found in [27].

3.2. Linear multi-task proximal support vector machine

Consider the following setup. We have T tasks and assume that the data of all the tasks are from the same space $X \times Y$. To keep the same setting as above, we assume that $X \in \mathbb{R}^n$ and $Y \in \mathbb{R}$ for regression or $Y \in \{\pm 1\}$ for classification. For task t we have m_t data points

$$\mathcal{D}_t = \{(x_{1t}, y_{1t}), (x_{2t}, y_{2t}), \ldots, (x_{m_t t}, y_{m_t t})\} \quad t = 1, 2, 3, \ldots, T.$$

D_t is sampled from a distribution $\mathcal{P}_t$, supposing $\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_T$ are related. For task t, the data are represented by $m_t \times n$ dimensional matrix A_t, and the label is represented by $m_t \times m_t$ diagonal matrix D_t with plus one or minus one along its diagonal. The goal of multi-task learning is to learn T different functions using the correlations among all the tasks, i.e., $f_t(x_{it}) = y_{it}$. In this paper, we will learn T different hyperplanes, i.e., $f_t(x_{it}) = x'_{it}w_t - \gamma_t$.

In this paper, MTPSVM assumes all tasks share a common parameter to measure the relationship among the tasks as used in hierarchical Bayesian models [34,40,41] and regularized multi-task learning [19]. They all assume functions w_t share a mean function w_0 with an additional offset u_t. The hyperplane for task t can be formulated as follows:

$$w_t = w_0 + u_t$$

where w_0 is a shared or mean hyperplane among all the tasks and u_t is an offset of particular task t. With the above setup, we now give the primal problem of the proposed MTPSVM

$$\begin{aligned} \min_{(w_0,u_t,\gamma_t,\xi_t)} \quad & \frac{1}{2}\|w_0\|^2 + \frac{\upsilon}{2}\sum_{t=1}^{T}\xi'_t\xi_t + \frac{\lambda}{2T}\sum_{t=1}^{T}(u'_t u_t + \gamma_t^2) \\ \text{s.t.} \quad & D_t(A_t(w_0 + u_t) - e\gamma_t) + \xi_t = e_t \\ \text{for} \quad & t = 1, 2, 3, \ldots, T \end{aligned} \tag{3}$$

In the above multi-task problem, υ and λ are positive regularization parameters. Here, we give the same parameter constraint λ on offset u_t from the mean hyperplane w_0 and bias γ_t. υ is used to constrain the slack variables ξ_t. Different values of λ will determine the relationship of T tasks. Larger λ will make the T models more similar (u_t tends to be smaller) while smaller λ results in less similar ones. When learning a single-task proximal support vector machine, we need to solve T different problems separately as shown in problem (2). From problem (3), we can see that T different problems should be solved as a whole because of the connection of sharing parameter w_0. This is the key which allows our MTPSVM to learn share information among tasks.

The primal problem with equal constraints for MTPSVM is a convex problem, so Karush–Kuhn–Tucker (KKT) conditions are necessary and sufficient conditions for optimization. The Lagrangian is the following:

$$\begin{aligned} L(w_0, u_t, \gamma_t, \xi_t) = & \frac{1}{2}\|w_0\|^2 + \frac{\upsilon}{2}\sum_{t=1}^{T}\xi'_t\xi_t + \frac{\lambda}{2T}\sum_{t=1}^{T}(u'_t u_t + \gamma_t^2) \\ & - \sum_{t=1}^{T}\alpha'_t(D_t(A_t(w_0 + u_t) - e_t\gamma_t) + \xi_t - e_t), \end{aligned} \tag{4}$$

where α_t is the Lagrange multiplier associated with the equality constraint for task t. Giving the gradients of the Lagrangian with respect to $(w_0, u_t, \gamma_t, \xi_t)$ and setting them to zero, we obtain KKT

conditions as follows:

$$\begin{aligned} & w_0 - \sum_{t=1}^{T} A_t' D_t \alpha_t = 0 \\ & \frac{\lambda}{T} u_t - A_t' D_t \alpha_t = 0 \\ & \upsilon \xi_t - \alpha_t = 0 \\ & \frac{\lambda}{T} \gamma_t + e_t' D_t \alpha_t = 0 \\ & (D_t(A_t(w_0 + u_t) - e_t \gamma_t) + \xi_t - e_t) = 0 \end{aligned} \tag{5}$$

We then have the following equalities with respect to the primal problem variables $(w_0, u_t, \gamma_t, \xi_t)$ and the Lagrangian multiplier α_t

$$\begin{aligned} & w_0 = \sum_{t=1}^{T} A_t' D_t \alpha_t \\ & u_t = \frac{T}{\lambda} A_t' D_t \alpha_t \\ & \xi_t = \frac{\alpha_t}{\upsilon} \\ & \gamma_t = -\frac{T}{\lambda} e_t' D_t \alpha_t \end{aligned} \tag{6}$$

Calculating T different Lagrange multipliers α_t is the key to the solution of the problem. Replacing $(w_0, u_t, \gamma_t, \xi_t)$ with above equalities in the last equality in Eq. (5), we have

$$D_t\left(A_t\left(\sum_{t=1}^{T} A_t' D_t \alpha_t + \frac{T}{\lambda} A_t' D_t \alpha_t\right) + \frac{T}{\lambda} e_t e_t' D_t \alpha_t\right) + \frac{\alpha_t}{\upsilon} - e_t = 0. \tag{7}$$

To solve this problem, we need to simplify above equality. Let $A = (A_1', A_2', \ldots, A_T')$, $D = \mathrm{diag}(D_1, D_2, \ldots, D_T)$, $\alpha = (\alpha_1', \alpha_2', \ldots, \alpha_T')'$. Then we have $\sum_{t=1}^{T} A_t' D_t \alpha_t = AD\alpha$. Eq. (7) can be rewritten as

$$D_t A_t AD\alpha + \left(\frac{T}{\lambda} D_t(A_t A_t' + e_t e_t') D_t + \frac{I_t}{\upsilon}\right)\alpha_t = e_t \tag{8}$$

Let $P_t = (T/\lambda) D_t(A_t A_t' + e_t e_t') D_t + (I_t/\upsilon)$, $P = \mathrm{diag}(P_1, P_2, \ldots, P_T)$. Substituting P_t in the above equality and combining the T different equalities together, we have

$$\alpha = (DA'AD + P)^{-1} e \tag{9}$$

Finally, we obtain Lagrange multipliers from the above formulation. Compared with PSVM, we find that the solution form of the Lagrange multiplier is very similar to that of MTPSVM. We obtain $(w_0, u_t, \gamma_t, \xi_t)$ by substituting the solved Lagrange multipliers in equalities (6). The necessary calculation of the inverse of a large matrix $(DA'AD + P)$ is time consuming if we calculate it directly. We will discuss the optimization of calculating the Lagrange multipliers below.

3.3. Linear multi-task proximal support vector machine for unbalanced classifications

As mentioned in [42], we often encounter the situation where data points are unbalanced between positive and negative samples, especially for a binary classification problem. There are various methods for handling unbalanced data, such as upsampling and downsampling. Inspired by the method in [42], we propose the balanced MTPSVM (B_MTPSVM) to solve the unbalanced sample problem in MTPSVM. The main idea for B_MTPSVM is to penalize samples with different weights according to the number of data points in that class. Suppose there are m_{t1} positive data points and m_{t2} negative data points for task t. We define a diagonal matrix N_t with diagonal elements N_{tii} as follows:

$$N_{tii} = \begin{cases} \dfrac{1}{m_{t1}} & d_{ii} = 1 \\ \dfrac{1}{m_{t2}} & d_{ii} = -1 \end{cases} \tag{10}$$

where d_{ii} is the diagonal element of D_t. With the balancing matrix N_t, the balanced MTPSVM problem is formulated as the following:

$$\begin{aligned} \min_{(w_0, u_t, \gamma_t, \xi_t)} \quad & \frac{1}{2}\|w_0\|^2 + \frac{\upsilon}{2}\sum_{t=1}^{T} \xi_t' N_t \xi_t + \frac{\lambda}{2T}\sum_{t=1}^{T}(u_t' u_t + \gamma_t^2) \\ \text{s.t.} \quad & D_t(A_t(w_0 + u_t) - e_t\gamma_t) + \xi_t = e_t \\ \text{for} \quad & t = 1, 2, 3, \ldots, T \end{aligned} \tag{11}$$

The balanced MTPSVM problem is very similar to standard MTPSVM except that it penalizes positive and negative data points with different weights. Therefore the solution for balanced MTPSVM is similar to that for MTPSVM. We can derive the solution by making just a slight change on the solution of MTPSVM. Replacing $P_t = (T/\lambda) D_t(A_t A_t' + e_t e_t^T) D_t + (I_t/\upsilon)$ with $P_t = (T/\lambda) D_t(A_t A_t' + e_t e_t^T) D_t + (N_t/\upsilon)$, the Lagrange multiplier is still

$$\alpha = (DA'AD + P)^{-1} e \tag{12}$$

3.4. Calculating optimization for Lagrange multiplier

In a real word problem, the number of data points is often very large, such as large-scale image classification. The calculation of the inverse of matrix $(DA'AD + P)$ will be time consuming if the number of data points is very large. In this section, we propose a method to optimize the calculation of Lagrange multipliers. The total number of data points of all tasks is

$$M = m_1 + m_2 + \cdots + m_T.$$

Consider $\alpha = (DA'AD + P)^{-1} e$, we need to calculate the inverse of an $M \times M$ dimensional matrix. If we have tens of thousands of data points, the computational time required to compute the inverse of so large a matrix will be considerable. To solve this problem we can reformulate α as follows: let $H = DA'$ and use the Sherman–Morrison–Woodbury formula for matrix inversion [27,43]

$$\begin{aligned} \alpha &= (DA'AD + P)^{-1} e \\ &= \left(P^{-1} - P^{-1} H\left(I + H' P^{-1} H\right)^{-1} H' P^{-1}\right) e. \end{aligned} \tag{13}$$

Given this equality, we only need to compute the inverse of matrix P and the inverse of matrix $(I + H' P^{-1} H)$. It is easy to compute the inverse of matrix $(I + H' P^{-1} H)$, as it is an $n \times n$ dimensional matrix where n is the dimension of the data space. (If the dimension of the data is large, dimension reduction may be a good choice.) As for P, it is still a high dimensional matrix; however, it is a block diagonal matrix. The inverse of matrix P can be computed as follows:

$$\begin{aligned} & P^{-1} = \mathrm{diag}(P_1^{-1}, P_2^{-1}, \ldots, P_T^{-1}) \\ & P_t = \frac{T}{\lambda} D_t(A_t A_t' + e_t e_t^T) D_t + \frac{I_t}{\upsilon}. \end{aligned} \tag{14}$$

Next, the problem of computing P^{-1} converts to computing T different P_t^{-1}. Therefore, computing P_t^{-1} will be the key problem

$$P_t^{-1} = \left(\frac{T}{\lambda} D_t(A_t A_t' + e_t e_t^T) D_t + \frac{I_t}{\upsilon}\right)^{-1}. \tag{15}$$

Comparing with PSVM, we find that the above formula is similar to the one in PSVM [27], except that there is an additional coefficient T/λ for every P_t. Following the method in PSVM, we can reuse the Sherman–Morrison–Woodbury formula for matrix inversion. Let $F_t = D_t[A_t \quad e_t]$, then P_t^{-1} can be expressed as the

following:

$$P_t^{-1}=\left(\frac{T}{\lambda}F_tF_t'+\frac{I_t}{\upsilon}\right)^{-1}. \tag{16}$$

Employing Sherman–Morrison–Woodburry gives the following:

$$P_t^{-1}=\upsilon\left(I-F_t\left(\frac{\lambda I_t}{T\upsilon}+F_t'F_t\right)^{-1}F_t'\right). \tag{17}$$

This formula further reduces the computational time of matrix inversion as we just need to compute an $(n+1)\times(n+1)$ dimensional matrix $((\lambda I_t/T\upsilon)+F_t'F_t)$. From the above optimization, the inverse of an $M\times M$ matrix is converted to the inverse of an $n\times n$ matrix P and T different $(n+1)\times(n+1)$ dimensional matrixes $((\lambda I/T\upsilon)+F_t'F_t)$. This demonstrates that the computational time of MTPSVM relies mainly on the dimension of the data rather than the amount of the data and that it can be used for large-scale datasets.

4. Multi-task proximal support vector regression

Having determined the derivation of the multi-task proximal support vector machine, it is easy to convert the multi-task proximal support vector machine to multi-task proximal support vector regression. The problem of proximal support vector regression is not discussed in [27]. Therefore, we first show the primal problem of proximal support vector regression (PSVR) and then extend it to multi-task proximal support vector regression (MTPSVR). Suppose $\overline{Y}$ is an $m\times 1$ vector $(y_1,y_2,\ldots,y_m)',y_i\in\mathbb{R},i=1,2,\ldots,m$ and other settings are the same as used in PSVM

$$\begin{aligned}&\min_{(w,\gamma,\xi)\in\mathbb{R}(m+n+1)}\quad \frac{1}{2}\upsilon\xi'\xi+\frac{1}{2}(w'w+\gamma^2)\\&\text{s.t.}\quad \overline{Y}=Aw-e\gamma+\xi\end{aligned} \tag{18}$$

We hope to predict the output within ξ error with the above equation. With the same assumption of MTPSVM, we can extend PSVR to multi-task proximal support vector regression

$$\begin{aligned}&\min_{(w_0,u_t,\gamma_t,\xi_t)}\quad \frac{1}{2}\|w_0\|^2+\frac{\upsilon}{2}\sum_{t=1}^{T}\xi_t'\xi_t+\frac{\lambda}{2T}\sum_{t=1}^{T}(u_t'u_t+\gamma_t^2)\\&\text{s.t.}\quad Y_t=A_t(w_0+u_t)-e_t\gamma_t+\xi_t\\&\text{for}\quad t=1,2,3,\ldots,T\end{aligned} \tag{19}$$

The Lagrangian is

$$\begin{aligned}L(w_0,u_t,\gamma_t,\xi_t)=&\frac{1}{2}\|w_0\|+\frac{\upsilon}{2}\sum_{t=1}^{T}\xi_t'\xi_t+\frac{\lambda}{2T}\sum_{t=1}^{T}(u_t'u_t+\gamma_t^2)\\&-\sum_{t=1}^{T}\alpha_t'(A_t(w_0+u_t)-e_t\gamma_t)+\xi_t-Y_t).\end{aligned} \tag{20}$$

Problem (19) is still a convex optimization problem. The KKT conditions are necessary and sufficient for optimization of the problem (19). Giving the gradients of the Lagrangian with respect to (w_0,u_t,γ_t,ξ_t) and setting them to zero, we have the KKT conditions as follows:

$$\begin{aligned}&w_0-\sum_{t=1}^{T}A_t'\alpha_t=0\\&\frac{\lambda}{T}u_t-A_t'\alpha_t=0\\&\upsilon\xi_t-\alpha_t=0\\&\frac{\lambda}{T}\gamma_t+\alpha_t'e_t=0\\&A_t(w_0+u_t)-e_t\gamma_t)+\xi_t-Y_t=0\end{aligned} \tag{21}$$

Next, we have the following equalities:

$$\begin{aligned}&w_0=\sum_{t=1}^{T}A_t'\alpha_t\\&u_t=\frac{T}{\lambda}A_t'\alpha_t\\&\xi_t=\frac{\alpha_t}{\upsilon}\\&\gamma_t=-\frac{T}{\lambda}\alpha_t'e_t\end{aligned} \tag{22}$$

Replacing (w_0,u_t,γ_t,ξ_t) with above equalities in the last equality in Eq. (21), we have

$$A_t\left(\sum_{t=1}^{T}A_t'\alpha_t+\frac{T}{\lambda}A_t'\alpha_t\right)+\frac{T}{\lambda}e_te_t'\alpha_t+\frac{\alpha_t}{\upsilon}-Y_t=0. \tag{23}$$

The method to solve the above problem is similar to the one solving MTPSVM. Let $A=(A_1',A_2',\ldots,A_T')$, $\alpha=(\alpha_1',\alpha_2',\ldots,\alpha_T')'$, then we have $\sum_{t=1}^{T}A_t'\alpha_t=A\alpha$

$$A_tA\alpha+\frac{T}{\lambda}\left(A_tA_t'+e_te_t'+\frac{\lambda}{T\upsilon}\right)\alpha_t=Y_t. \tag{24}$$

Let $P_t=A_tA_t'+e_te_t'+(\lambda/T\upsilon)$, $P=\mathrm{diag}(P_1,P_2,\ldots,P_T)$, $Y=(Y_1',Y_2',\ldots,Y_T')'$. Substituting P_t in the above equality and combining the T different equalities together, we have

$$\alpha=\left(A'A+\frac{T}{\lambda}P\right)^{-1}Y. \tag{25}$$

To solve the above problem, we still need to compute the inverse of an $M\times M$ matrix, where M is the total amount of training data. We need to convert the computing of the inverse of a large matrix to computing smaller ones. As done in MTPSVM, we have the following equality when using the Sherman–Morrison–Woodbury formula:

$$\begin{aligned}\alpha&=\left(A'A+\frac{T}{\lambda}P\right)^{-1}Y\\&=\left(\frac{\lambda}{T}P^{-1}-\frac{\lambda^2}{T^2}P^{-1}A'\left(I+\frac{\lambda}{T}AP^{-1}A'\right)^{-1}AP^{-1}\right)Y.\end{aligned} \tag{26}$$

In the above equation, the inverse of matrix P can be reformulated as $P^{-1}=\mathrm{diag}(P_1^{-1},P_2^{-1},P_3^{-1},\ldots,P_T^{-1})$. Thus the inverse of large matrix $(A'A+(T/\lambda P))$ is converted to the inverse of smaller matrix $(I+(\lambda/T)AP^{-1}A')$ and P_t, whose size is only related to the dimension of the data.

5. Experiments

We show empirical results of our proposed multi-task models on three real world datasets including two classification datasets and one regression dataset. The regression dataset is the school dataset, which is developed and used to evaluate the performance of multi-task learning in many works [31,25,19,30]. We will test the performance of MTPSVR on this dataset. The two classification datasets are the landmine dataset [24,14] and a multi-task image classification dataset using images from Pascal, Caltech, Flickr and ImageNet. The landmine dataset is very suitable for multi-task learning because it contains 39 different tasks related to each other. Multi-task image classification is more complex than the tasks of the landmine or school datasets. First, multi-task learning methods should have the ability to distinguish low-level features of the image content. Second, the features have a high dimension. Experiments are performed on a desktop with Intel Core i3-2130 CPU and 4G RAM. All experiment results demonstrate the merits of our MTPSVM.

5.1. Landmine dataset

In this section, we show experiments on the landmine dataset. The landmine dataset consists of 29 binary classification tasks collected from different landmine fields. The number of data samples varies from 445 to 690 with nine dimensions. Both single-task PSVM and MTPSVM are evaluated in this dataset. Additionally, we compare our MTPSVM with three other multi-task learning methods, including multi-task sparsity via maximum entropy discrimination (MED) [24], multi-task feature learning (MTL-FEAT) [25] and group multi-task feature learning (GMTL-FEAT) [26]. MED and our MTPSVM have something in common, as they are both based on the support vector machines. GMTL-FEAT achieves remarkable success with group multi-task learning methods. MTL-FEAT is a popular framework for multi-task feature learning and it is the basis of many other multi-task learning methods. The performance of balanced MTPSVM and balanced PSVM will be given compared with standard MTPSVM and PSVM without balancing. All methods use a linear kernel to run the experiments. Both AUC (area under the curve) and running time are evaluated by training a varied number of training data exam ples: $20, 40, \ldots, 160$.

Values υ for PSVM and (υ, λ) for MTPSVM are chosen through a validation set as follows. We validate PSVM by setting $\upsilon = 2^i$, where $i = -2, -1, 0, 1, 2, \ldots, 9$. For MTPSVM, we used the same setting $\upsilon = 2^i$ and $\lambda = 2^j$, where $i = -2, -1, 0, 1, 2, \ldots, 9$ and $j = -2, -1, 0, 1, 2, \ldots, 9$. Values υ and (υ, λ) are chosen to give the best performance on the validation set. Next, we use the chosen parameter to train PSVM and MPSVM on the training set and test it on the test set. The method for choosing parameters for the other three multi-task learning methods is similar to PSVM and MTPSVM. We find the best parameters on the validation set according to the methods used in related papers. We run all the experiments five times to avoid randomness and report average performance.

We first compare our proposed MTPSVM with PSVM in both balanced and unbalanced cases. Fig. 2 shows that balanced methods improve the performance by approximately 1% ~ 2% compared with unbalanced methods. This shows that the balanced methods have better performance than unbalanced methods on unbalanced datasets. Therefore, the following experiments will use the balanced methods. From Fig. 2, it is also obvious that the multi-task learning method outperforms single-task learning on the landmine dataset. It is interesting to find that when the amount of training data is small, MTPSVM has a much better performance than PSVM. As the amount of training data increases, they have almost the same performance. This is because small amount of training data provide less information for single-task learning, while multi-task learning can find the correlation among different tasks leading to more information. It makes sense that multi-task learning gains better performance with more information about the data set. However, when the amount of training data increases, the single-task learning model can learn enough information from its own training data, and the multi-task learning method learns little new information from other tasks. From this point of view, multi-task learning and single-task learning will have comparable performance.

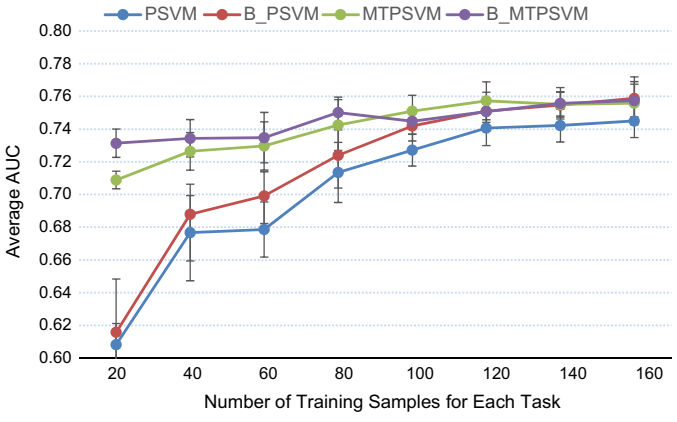

Fig. 2. Comparison of PSVM, balanced PSVM (B_PSVM), MTPSVM and balanced MTPSVM (B_MTPSVM) on Landmine dataset.

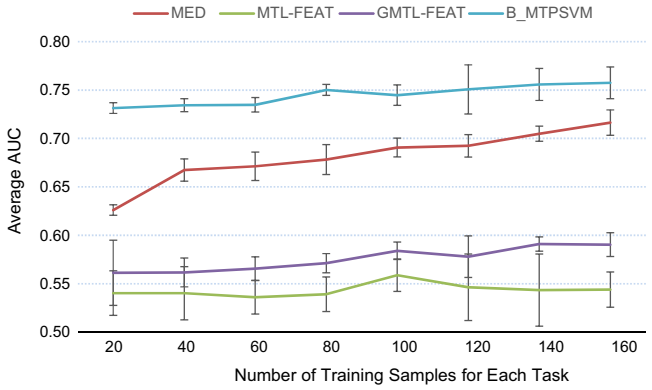

Fig. 3. Performance comparison between our multi-task learning methods and three other methods on Landmine dataset.

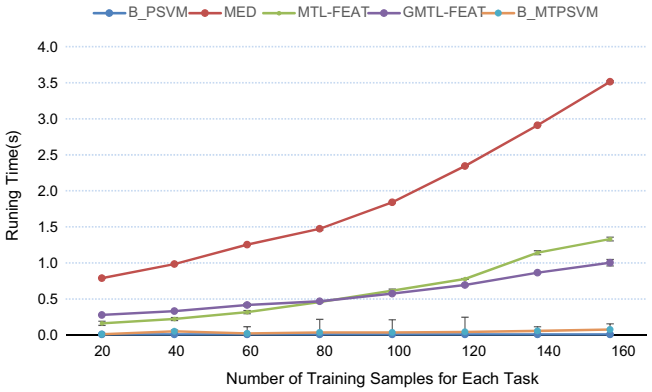

Fig. 4. Running time comparison of B_PSVM, B_MTPSVM and three other multi-task learning methods on Landmine dataset.

Fig. 3 compares our MTPSVM with three other multi-task learning methods. It is obvious that our MTPSVM outperforms the other three multi-task learning methods significantly. In this figure, we can see that the performance of MTPSVM reaches a high level when the training number is only 20 and improves consistently as the amount of training data increases from 20 to 160. This shows that MTPSVM can learn much more information than the other three multi-task learning methods when the amount of training data is small. The reason MTL-FEAT and GMTL-FEAT perform badly maybe that MTL-FEAT and GMTL-FEAT learn a sparse latent feature. However, the landmine data has a low dimension. Sparse representations may ignore some information and not be suitable for such a dataset. Our MTPSVM does not have such a problem, as we need not learn a sparse shared feature to measure the relationship among these tasks.

Fig. 4 shows the efficiency of MTPSVM. MTPSVM has comparable speed with PSVM on the landmine dataset and is much faster than the other three multi-task learning methods. For example, PSVM uses approximately 0.008 s, MTPSVM uses 0.009 s, MTL-FEAT uses 0.162 s, MED uses 0.789 s and GMTL-FEAT uses 0.278 s when running the experiments with the number of training

samples for each task equal to 20. The speed of MTPSVM is about two orders of magnitude faster than the three other multi-task learning methods. As the amount of training data increases, the running time of the other three multi-task methods increases greatly. However, the running time of MTPSVM increases only slightly. This also demonstrates that the running time of MTPSVM is determined by the dimension of the data rather than the amount of training data, as mentioned in Section 3.

5.2. Multi-task image classification

In this section, we conduct experiments on one multi-task image classification dataset to demonstrate the effectiveness of our MTPSVM. The multi-task image classification dataset includes four sub-datasets: Flickr, Caltech, ImageNet, and Pascal. Each sub-dataset has 10 classes: "airplane", "bicycle", "bus", "car", "cat", "chair", "dog", "horse", "motorbike", and "sheep". The classification of each sub-dataset is treated as one task related to the other three tasks. Thus, there are four subtasks for each class, and each subtask is regarded as a binary classification. Take the "airplane" category for example, single-task learning methods learn four classifiers independently for the four sub-datasets using just the images from that sub-dataset. Multi-task learning methods learn four classifiers jointly for the four sub-datasets using all the images. For the Flickr dataset, we searched and downloaded images from the Flickr website. The number of images in different classes ranges from 56 to 300. We chose images for Caltech from Caltech 101 and Caltech 256 by using all images in the relevant category. As for Pascal, we chose images from the Pascal 2007 dataset with all the training and test images related to the 10 different classes. We download the 10 different categories for the ImageNet dataset from the ImageNet website. The number of images in each category from ImageNet ranges from 910 to 1603. We give some example images of the dataset in Fig. 5. From these examples, we can see that images from different datasets vary in different aspects. For example, in the car category, images of cars in the Caltech dataset are all pictures of sides of cars, while car images in the other three datasets have more variety.

We randomly select 10 examples from every class as a training set, and the rest are split into two halves for validation and test. In other words, we use 400 examples as a training set. As for the features, we use dense sift and quantize them into 600 visual words with codebooks computed using bag-of-word models. We first map the 600 dimensional features into a higher-dimensional linear space, 1800 dimensions in this paper, using feature map [44]. Then we apply linear MTPSVM and linear PSVM on these higher dimensional features instead of using non-linear kernel on the original 600 dimensional features. This results in higher efficiency of experiments. We use the standard procedures for selecting the parameters as mentioned in experiments on land-mine dataset.

For this real-world image classification task, we also compare MTPSVM with MTL-FEAT, MED and GMTL-FEAT. Additionally, the performance of pooling PSVM and pooling LIBSVM is shown. "Pooling" refers to training just one classifier using all of the data for each class. We use average precision to evaluate the performance of the classifiers. The results are shown in Table 1. It is clear that MTPSVM outperforms pooling PSVM, pooling LIBSVM and the other three multi-task learning methods. The reason that the classification performance of pooling PSVM and pooling LIBSVM is worse than that of multi-task learning methods is that they do not consider correlation among tasks and just learn one classifier for all the tasks. Data from other tasks can be observed as a type of noise adding to the current task leading to worse performance. Across the 10 classes, MTPSVM outperforms all the baseline methods on eight classes. MED performs best on category chair and pooling PSVM performs best on category dog. The last row of Table 1 shows the mean average precision of the 10 classes. The mean average precision of MTPSVM is 48.58%, while GMTL-FEAT performs the worst. We find that GMTL-FEAT may not be suitable for high dimension features, which leads to bad performance. Therefore, in this experiment, we first reduce high-dimension feature to a low feature space for GMTL-FEAT. Our MTPSVM consistently outperforms other methods on such a multi-task image classification dataset.

5.3. School dataset

In this section, we will test our multi-task regression method on the school dataset, which is from the Inner London Education Authority. This dataset is publicly available and has been used for evaluating many multi-task learning methods, for example [31,25]. It consists of examination scores of 15,362 students from 139 secondary schools in 1985, 1986 and 1987. There are 139 different tasks corresponding to predicting the examination scores in that school. The input feature includes the year of the examination, four school-dependent features and three student-dependent

Table 1
Performance comparison between MTPSVM and baseline methods on each of the 10 classes as well as the MAPs over all classes.

Category	Pooling PSVM (%)	Pooling LIBSVM (%)	MED (%)	MTL-FEAT (%)	GMTL-FEAT (%)	MTPSVM (%)
Airplane	79.01	78.25	79.77	79.58	80.33	**80.92**
Bicycle	41.69	41.12	42.65	43.03	41.07	**44.63**
Bus	62.91	60.87	65.07	64.41	60.04	**67.37**
Car	49.04	52.44	54.08	52.81	48.03	**54.31**
Cat	36.75	35.75	37.77	37.43	37.29	**38.77**
Chair	41.75	44.10	**47.07**	44.74	43.32	46.54
Dog	**35.32**	31.78	30.14	32.89	30.04	34.32
Horse	30.57	29.32	29.75	31.04	28.58	**31.74**
Motorbike	42.87	42.50	43.26	43.15	38.96	**44.03**
Sheep	39.47	38.52	40.33	41.14	41.00	**43.14**
MAP	45.94	45.46	46.99	47.02	44.87	**48.58**

Fig. 5. Example images in Caltech, Pascal, ImageNet and Flickr for multi-task image classification.

Table 2
Performance comparison of our methods and methods proposed in [25] on school dataset (training size 75%).

Method	Explained variance (%)
Aggregate	22.7 ± 1.3
Independent	23.8 ± 2.1
MTL-FEAT(variable selection)	24.8 ± 2.0
MTL-FEAT(linear kernel)	26.7 ± 2.0
MTL-FEAT(Gaussian kernel)	26.4 ± 1.9
PSVR(linear kernel)	22.6 ± 1.6
MTPSVR(linear kernel)	**28.0 ± 1.3**

Table 3
Experimental performance and running time by MTPSVM with varying training sizes.

Training size (%)	Explained variance (%)	Running time (s)
20	25.35 ± 0.53	0.039 ± 0.002
30	26.81 ± 0.59	0.048 ± 0.005
40	27.58 ± 0.54	0.057 ± 0.007
50	27.74 ± 0.51	0.061 ± 0.004
60	27.97 ± 0.89	0.072 ± 0.001
70	28.07 ± 1.14	0.082 ± 0.003

features. The features relevant to school are percentage of students eligible for free school meals, percentage of students in VR band one (highest band in a verbal reasoning test), school gender and school denomination. Four student-dependent features are gender, VR band (taking the value 1, 2 or 3) and ethnic group. To compare with other methods, we follow the same setup as other multi-task learning methods do by creating binary variable for each possible attribute value. Finally, there are 19 student-dependent features and eight school-dependent features.

We use the same 10 random splits of the dataset as used in [25]. Seventy-five percent of the examples from each school are as training data and 25% as test data. The number of examples differs from school to school and on average the training set includes 80 examples per school and the test set includes 30 examples per school. To compare with other methods, we use the measure of percentage of explained variance used in [31], which is defined as the total variance of the data minus the sum-squared error on the test set as a percentage of the total variance. We select the parameter for PSVR and MTPSVR through validation as done in experiments on landmine dataset.

Table 2 shows the performance comparison between our method and multi-task feature learning (MTL-FEAT) [25]. MTL-FEAT is a popular framework to learn shared features and often compared as a baseline in multi-task learning [26,20]. An "Independent" result is achieved by training 139 ridge regressions. The "Aggregate" result is obtained by training just one ridge regression on the entire dataset. MTL-FEAT of variable selection is one special case for variable selection using multi-task feature learning. MTL-FEAT of linear kernel and Gaussian kernel refers to multi-task feature learning using linear kernel and Gaussian kernel. From Table 2, we can see that our MTPSVM outperforms PSVR and other methods. Additionally, MTPSVM outperforms PSVR approximately 7.5% and MTL-FEAT just improves the performance of single-task learning (Independent) approximately 3%. This indicates that MTPSVM has greater potential than MTL-FEAT to improve performance for single-task learning.

Table 3 shows the performance and running time of MTPSVM with various training sizes. We can see the high efficiency of MTPSVM from the table. The running time just reaches approximately 0.082 s when using 70% of the 15,362 examples as training set. The performance can reach a high level of 25.35% using just 20% of the examples as a training set. This is comparable to the performance of MTL-FEAT(variable selection) when using 75% of the examples as training set.

6. Conclusion and future work

In this paper, we propose a novel multi-task learning method based on PSVM. We give a detailed derivation of our MTPSVM and extend it for unbalanced data (B_MTPSVM). Considering the efficiency problem, the calculating procedure of MTPSVM is optimized, which leads to high efficiency. Experiments are conducted on three datasets: the landmine dataset, the school dataset and one multi-task image classification dataset. We compare both the performance and the running time of MTPSVM, PSVM and three other popular multi-task learning methods. All results demonstrate that MTPSVM has better performance and much shorter running time. Additionally, MTPSVM performs quite well especially when the amount of training data is small.

In the future, we will extend our multi-task learning algorithm to more-general settings. In this paper, we make an assumption that all tasks share a mean hyperplane to measure the relationship among all the tasks. Although it is suitable for some cases, data in real life is more complicated and may not be suitable for such an assumption. It is difficult to derive the real relationship of the parameters among all the tasks without sufficient prior information. We will try to combine latent feature learning with parameter sharing for multi-task learning to handle this problem. Our main idea is that the original feature space may not be suitable for such an assumption. However, we can consider learning a shared latent future subspace that is suitable for our assumption.

References

[1] W. Hu, R. Hu, N. Xie, H. Ling, S. Maybank, Image classification using multiscale information fusion based on saliency driven nonlinear diffusion filtering, IEEE Trans. Image Process.: Publ. IEEE Signal Process. Soc. 23 (4) (2014) 1513–1526.
[2] L. Zhang, X. Zhen, L. Shao, Learning object-to-class kernels for scene classification, IEEE Trans. Image Process. 23 (8) (2014) 3241–3253.
[3] F. Zhu, Z. Jiang, L. Shao, Submodular object recognition, in: IEEE Conference on Computer Vision and Pattern Recognition, CVPR, 2014, pp. 2457–2464 http://dx.doi.org/10.1109/CVPR.2014.315.
[4] Q. Qiu, V.M. Patel, P. Turaga, R. Chellappa, Domain adaptive dictionary learning, in: Computer Vision—ECCV, 2012, pp. 631–645.
[5] P. Turaga, A. Veeraraghavan, A. Srivastava, R. Chellappa, Statistical computations on Grassmann and Stiefel manifolds for image and video-based recognition, IEEE Trans. Pattern Anal. Mach. Intell. 33 (11) (2011) 2273–2286.
[6] X. Shen, Z. Lin, J. Brandt, Y. Wu, Spatially-constrained similarity measure for large-scale object retrieval, IEEE Trans. Pattern Anal. Mach. Intell. 36 (6) (2014) 1229–1241. http://dx.doi.org/10.1109/TPAMI.2013.237.
[7] A.J. Joshi, F. Porikli, N. Papanikolopoulos, Multi-class active learning for image classification, in: IEEE Conference on Computer Vision and Pattern Recognition, 2009, CVPR 2009, 2009, pp. 2372–2379.
[8] Y.-H. Shao, W.-J. Chen, J.-J. Zhang, Z. Wang, N.-Y. Deng, An efficient weighted lagrangian twin support vector machine for imbalanced data classification, Pattern Recognit. 47 (9) (2014) 3158–3167.
[9] A. Tayal, T.F. Coleman, Y. Li, Primal explicit max margin feature selection for nonlinear support vector machines, Pattern Recognit. 47 (6) (2014) 2153–2164.
[10] Z. Jiang, G. Zhang, L.S. Davis, Submodular dictionary learning for sparse coding, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, Washington, DC, USA, 2012, pp. 3418–3425.
[11] F. Zhu, L. Shao, Weakly-supervised cross-domain dictionary learning for visual recognition, Int. J. Comput. Vis. 109 (1-2) (2014) 42–59.
[12] L. Shao, L. Liu, X. Li, Feature learning for image classification via multiobjective genetic programming, IEEE Trans. Neural Netw. Learn. Syst. 25 (7) (2014) 1359–1371.
[13] L. Shao, D. Wu, X. Li, Learning deep and wide: a spectral method for learning deep networks, IEEE Trans. Neural Netw. Learn. Syst. 25 (12) (2014) 2303–2308.
[14] Y. Xue, X. Liao, L. Carlin, B. Krishnapuram, Multi-task learning for classification with Dirichlet process priors, J. Mach. Learn. Res. 8 (2007) 35–63.
[15] R. Caruanal, Multitask learning, Mach. Learn. 28 (1) (1997) 41–75. http://dx.doi.org/10.1023/A:1007379606734.
[16] S. Thrun, Learning to learn: introduction, in: Learning To Learn.

[17] J. Baxter, A model of inductive bias learning, J. Artif. Intell. Res. 12(1-C12) (2000) 149–198.
[18] P. Rai, H. Daume, Infinite predictor subspace models for multitask learning, in: International Conference on Artificial Intelligence and Statistics, 2010, pp. 613–620.
[19] T. Evgeniou, M. Pontil, Regularized multi–task learning, in: Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2004, pp. 109–117.
[20] Y. Zhang, D.-Y. Yeung, A Convex Formulation for Learning Task Relationships in Multi-task Learning, arXiv:1203.3536.
[21] S. Parameswaran, K.Q. Weinberger, Large margin multi-task metric learning, Adv. Neural Inf. Process. Syst. (2010) 1867–1875.
[22] K. Yu, V. Tresp, A. Schwaighofer, Learning gaussian processes from multiple tasks, in: Proceedings of the 22nd International Conference on Machine Learning, 2005, pp. 1012–1019.
[23] S.-I. Lee, V. Chatalbashev, D. Vickrey, D. Koller, Learning a meta-level prior for feature relevance from multiple related tasks, in: Proceedings of the 24th International Conference on Machine Learning, 2007, pp. 489–496.
[24] T. Jebara, Multitask sparsity via maximum entropy discrimination, J. Mach. Learn. Res. 12 (2011) 75–110.
[25] A. Argyriou, T. Evgeniou, M. Pontil, Convex multi-task feature learning, Mach. Learn. 73 (3) (2008) 243–272.
[26] Z. Kang, K. Grauman, F. Sha, Learning with whom to share in multi-task feature learning, in: Proceedings of the 28th International Conference on Machine Learning (ICML-11), 2011, pp. 521–528.
[27] G. Fung, O. L. Mangasarian, Proximal support vector machine classifiers, in: Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2001, pp. 77–86.
[28] V. Vapnik, The Nature of Statistical Learning Theory, Springer, 2000.
[29] S. Ben-David, R. Schuller, Exploiting task relatedness for multiple task learning, in: Learning Theory and Kernel Machines, 2003, pp. 567–580.
[30] T. Evgeniou, C.A. Micchelli, M. Pontil, Learning multiple tasks with kernel methods, J. Mach. Learn. Res. (2005) 615–637.
[31] B. Bakker, T. Heskes, Task clustering and gating for Bayesian multitask learning, J. Mach. Learn. Res. 4 (2003) 83–99.
[32] J. Baxter, Learning internal representations, in: Proceedings of the Eighth Annual Conference on Computational Learning Theory, 1995, pp. 311–320.
[33] T. Heskes, Solving a huge number of similar tasks: a combination of multi-task learning and a hierarchical Bayesian approach, ICML (1998) 233–241.
[34] T. Heskes, Empirical Bayes for learning to learn, ICML (2000) 367–374.
[35] J. Zhu, N. Chen, E. P. Xing, Infinite svm: a Dirichlet process mixture of large-margin kernel machines, in: Proceedings of the 28th International Conference on Machine Learning (ICML-11), 2011, pp. 617–624.
[36] M. Lapin, B. Schiele, M. Hein, Scalable multitask representation learning for scene classification, in: 2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 23–28, 2014, pp. 1434–1441, http://dx.doi.org/10.1109/CVPR.2014.186.
[37] A. Jalali, P.D. Ravikumar, S. Sanghavi, A dirty model for multiple sparse regression, IEEE Trans. Inf. Theory 59 (12) (2013) 7947–7968. http://dx.doi.org/10.1109/TIT.2013.2280272.
[38] P. Gong, J. Ye, C. Zhang, Robust multi-task feature learning, Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining (2012) 895–903.
[39] J. Chen, J. Zhou, J. Ye, Integrating low-rank and group-sparse structures for robust multi-task learning, in: Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2011, pp. 42–50.
[40] G.M. Allenby, P.E. Rossi, Marketing models of consumer heterogeneity, J. Economet. 89 (1) (1998) 57–78.
[41] N. Arora, G.M. Allenby, J.L. Ginter, A hierarchical Bayes model of primary and secondary demand, Market. Sci. 17 (1) (1998) 29–44.
[42] G.M. Fung, O.L. Mangasarian, Multicategory proximal support vector machine classifiers, Mach. Learn. 59 (1-C2) (2005) 77–97.
[43] M.C. Ferris, T.S. Munson, Interior-point methods for massive support vector machines, SIAM J. Optim. 13 (3) (2002) 783–804.
[44] A. Vedaldi, A. Zisserman, Efficient additive kernels via explicit feature maps, IEEE Trans. Pattern Anal. Mach. Intell. 34 (3) (2012) 480–492.

Ya Li received his B.S. degree in 2013 from the Department of Electronic Engineering and Information Science in the University of Science and Technology of China (USTC). He is now pursuing his Ph.D. degree in USTC and his research interest is machine learning.

Xinmei Tian is an associate professor in the Department of Electronic Engineering and Information Science, University of Science and Technology of China. She received the Ph.D. degree from the University of Science and Technology of China in 2010. Her current research interests include multimedia information retrieval and machine learning. She received the Excellent Doctoral Dissertation of Chinese Academy of Sciences award in 2012 and the Nomination of National Excellent Doctoral Dissertation award in 2013.

Mingli Song a professor of Computer Science with the College of Computer Science and Technology, Zhejiang University. He received his Ph.D degree in Computer Science and Technology from College of Computer Science, Zhejiang University, and B. Eng. Degree from Northwestern Polytechnical University. He was awarded Microsoft Research Fellowship in 2004. His research interests include Pattern Classification, Weakly Superivsed Clustering, Color and Texture Analysis, Object Recognition, and Reconstruction. He has authored and co-authored more than 70 scientific articles at top venues including IEEE T-PAMI, IEEE T-IP, T-MM, T-SMCB, Information Sciences, Pattern Recognition, CVPR, ECCV and ACM MM. He has served with more than 10 major international conferences including ICDM, ACM Multimedia, ICIP, ICASSP, ICME, PCM, PSIVT and CAIP, and more than 10 prestigious international journals including T-IP, T-VCG, T-KDE, T-MM, T-CSVT, T-NNLS and TSMCB. He is a Senior Member of IEEE, and Professional Member of ACM.

Dacheng Tao is a professor of Computer Science with the Centre for Quantum Computation & Intelligent Systems, and the Faculty of Engineering and Information Technology in the University of Technology, Sydney. He mainly applies statistics and mathematics to data analytics and his research interests spread across computer vision, data science, image processing, machine learning, neural networks and video surveillance. His research results have expounded in one monograph and 100+ publications at prestigious journals and prominent conferences, such as IEEE T-PAMI, T-NNLS, T-IP, T-CYB, JMLR, IJCV, NIPS, ICML, CVPR, ICCV, ECCV, AISTATS, ICDM; and ACM SIGKDD, with several best paper awards, such as the Best Theory/Algorithm Paper Runner Up Award in IEEE ICDM'07, the Best Student Paper Award in IEEE ICDM'13, and the 2014 ICDM 10 Year Highest-Impact Paper Award.