Multi-task learning with one-class SVM

Xiyan He, Gilles Mourot, Didier Maquin, José Ragot, Pierre Beauseroy, André Smolarz, Edith Grall-Maës

Multi-task learning with one-class SVM

Xiyan He [a,*,1], Gilles Mourot [a], Didier Maquin [a], José Ragot [a], Pierre Beauseroy [b], André Smolarz [b], Edith Grall-Maës [b]

[a] Centre de Recherche en Automatique de Nancy, CNRS UMR 7039, Nancy-Université, 54500 Vandœuvre-lès-Nancy, France
[b] Institut Charles Delaunay, STMR CNRS UMR 6279, Université de Technologie de Troyes, 10000 Troyes, France

ARTICLE INFO

Article history:
Received 22 May 2013
Received in revised form
31 October 2013
Accepted 5 December 2013
Communicated by Shiliang Sun
Available online 10 January 2014

Keywords:
Machine learning
Multi-task learning
One-class SVM
Diagnostics
Multiple sources

ABSTRACT

Multi-task learning technologies have been developed to be an effective way to improve the generalization performance by training multiple related tasks simultaneously. The determination of the relatedness between tasks is usually the key to the formulation of a multi-task learning method. In this paper, we make the assumption that when tasks are related to each other, usually their models are close enough, that is, their models or their model parameters are close to a certain mean function. Following this task relatedness assumption, two multi-task learning formulations based on one-class support vector machines (one-class SVM) are presented. With the help of new kernel design, both multi-task learning methods can be solved by the optimization program of a single one-class SVM. Experiments conducted on both low-dimensional nonlinear toy dataset and high-dimensional textured images show that our approaches lead to very encouraging results.

1. Introduction

In recent years, multi-task learning has received significant attention in many research areas. Different from traditional single task learning methods, multiple related tasks are learned simultaneously with the objective to improve the generalization performance of each task [1–6]. The concept of multi-task learning is to share some useful information, for example, a common representation space or some model parameters that are close to each other, between related tasks [7–12]. Therefore, the determination of the relatedness between tasks is usually the key to the formulation of a multi-task learning method [13,14].

A broad community of multi-task learning has focused on support vector machines (SVM) [8,15–21], which have been extensively studied for single task learning. The SVM [22,23] was initially developed to solve the two-class classification problem. It looks for the hyperplane that separates two different classes with maximum margin. The method can be easily generalized to non-linearly separable cases by the well-known "kernel trick" [23]. We first map the original data to some higher-dimensional feature space and then solve a linear problem in that space. In the framework of multi-task learning, a general relatedness assumption for the SVM based method is that the model parameter values of different tasks are close to each other [16]. Following this assumption, numerous SVM based multi-task learning methods have been exploited, not only for binary data classification [20] but also for multi-class classification [21]. The good properties of kernel functions make support vector machines well-suited for multi-task learning [15]. In this paper, we focus on the problem of one-class classification in the framework of multi-task learning.

One class classification, also known as novelty or outlier detection, aims at detecting samples that do not resemble the majority of the dataset. Only information about one class, usually referred to as positive class, is available in the training set. For example, in many applications of fault detection and diagnosis, it is very difficult to collect samples corresponding to all the abnormal behaviors of the system. The insufficient knowledge on the negative class makes this kind of problem more difficult than traditional two-class or multi-class classification problems. Previous work has shown empirically as well as theoretically that the multi-task learning framework can lead to more intelligent learning models with better performance [7,13–15,24–26]. Therefore, it may be beneficial to introduce one class classification methods in the framework of multi-task learning.

The one-class support vector machines (one-class SVM), proposed by Schölkopf et al. [27], and equivalent to the support vector domain description [28], is a typical one-class classification

* Corresponding author.
E-mail addresses: xiyan.he@gipsa-lab.grenoble-inp.fr (X. He), gilles.mourot@ensem.inpl-nancy.fr (G. Mourot), didier.maquin@ensem.inpl-nancy.fr (D. Maquin), jose.ragot@ensem.inpl-nancy.fr (J. Ragot), pierre.beauseroy@utt.fr (P. Beauseroy), andre.smolarz@utt.fr (A. Smolarz), edith.grall@utt.fr (E. Grall-Maës).
[1] Now with Grenoble Institute of Technology (Grenoble INP), GIPSA-Lab, 38000 Grenoble, France.

method. Instead of the estimation of the probability density [29–32], it focuses on the estimation of a bounded area for samples from the target class, which provides significant advantages over other one-class classification methods. Recently, Yang et al. [18] proposed a new method based on one-class SVM that makes use of the advantages of multi-task learning when conducting one-class classification (hereafter denoted by MTL-OC). The basic idea is to constrain the solutions of related tasks close to each other by upper-bounding the L_2 difference between each pair of parameters from related tasks as in [33]. Within the multi-task learning framework, the main problem of this method is that the introduced constraints significantly increase the difficulty of the optimization problem. The authors solve the problem via conic programming, which is time consuming. This method will be later used as a comparison method in our experimental study.

In this paper, inspired by the work of Evgeniou and Pontil [16], we present two multi-task learning formulation approaches based on one-class SVM. The first one, named as MTL-OSVM I, has previously appeared in [34]. It makes the same assumption as in [16,20,21] that the normal vector of the task model can be represented by the sum of a mean vector and a specific vector corresponding to each task. We use the same non-linear feature mapping for all the task models. Some new content has been included in the current extended version. In particular, we propose a more general multi-task learning formulation, named by MTL-OSVM II, in which each task model is represented by the sum of a generic model and a specific model. According to this new formulation, we may define different non-linear feature mappings for different tasks. These task relatedness assumptions are reasonable due to the observation that when the tasks are similar to each other, usually their models are close enough. Following these assumptions, a number of one-class SVMs are learned simultaneously in the proposed methods. As demonstrated later in this paper, both multi-task learning approaches are easy to implement since they only require, with the help of kernel trick, a simple modification of the optimization problem in the single one-class SVM.

The remainder of this paper is organized as follows. In Section 2, we briefly describe the formulation of the one-class SVM algorithm in the framework of single task learning. The details of the two proposed multi-task learning methods based on one-class SVM (named by MTL-OSVM I and MTL-OSVM II) are then outlined in Section 3. Section 4 presents the experimental results. We draw conclusion in Section 5.

2. Single-task learning: preliminary

In a single-task one class classification scenario, we are given a set of m training samples of a single class $\mathcal{A}_m = \{\boldsymbol{x}_i\}$, $i = 1, \ldots, m$, where $\boldsymbol{x}_i$ is a sample in the space $\mathcal{X} \subseteq \mathbb{R}^d$ of dimension d. The goal is to learn a function that represents the best in the given data. In this section, we briefly review the basic setup of one-class SVM classifier for single-task learning. The one-class SVM [27] has been proposed based on support vector machines for solving the problem of one-class classification. Under the assumption that the origin in the feature space belongs to the negative or outlier class, the boundary region estimation is achieved by separating the target samples (in a higher-dimensional feature space for non-linearly separable cases) from the origin by a maximum-margin hyperplane which is as far away from the origin as possible. If a new test sample falls within this region, then the decision function

$$f(\boldsymbol{x}) = \text{sign}(\langle \mathbf{w}, \phi(\boldsymbol{x}_i)\rangle - \rho) \tag{1}$$

will be positive. The hyperplane is determined by solving the following optimization problem [27]:

$$\begin{cases} \min\limits_{\mathbf{w},\xi,\rho} & \frac{1}{2}\|\mathbf{w}\|^2 + \frac{1}{\nu m}\sum\limits_{i=1}^{m}\xi_i - \rho \\ \text{subject to :} & \langle \mathbf{w}, \phi(\boldsymbol{x}_i)\rangle \geq \rho - \xi_i, \quad \xi_i \geq 0 \end{cases} \tag{2}$$

where ϕ is the non-linear feature mapping and ξ_i is a slack variable for relaxing the optimality constraints for certain training samples. A specific parameter for one-class SVM is $\nu \in (0, 1]$. It is an upper-bound of the ratio of outliers among all the training samples as well as a lower-bound of the ratio of support vectors among all the samples.

As in the classical binary SVM case [23], the primal problem is solved by its Lagrange dual:

$$\begin{cases} \min\limits_{\boldsymbol{\alpha}} & \frac{1}{2}\sum\limits_{i,j=1}^{m}\alpha_i\alpha_j\langle\phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j)\rangle \\ \text{subject to :} & 0 \leq \alpha_i \leq \frac{1}{\nu m}, \quad \sum\limits_{i=1}^{m}\alpha_i = 1 \end{cases} \tag{3}$$

where α_i are the Lagrange multipliers. By defining the so-called kernel function [23] $\langle\phi(\boldsymbol{x}_i), \phi(\boldsymbol{x}_j)\rangle = k(\boldsymbol{x}_i, \boldsymbol{x}_j)$, the mapping ϕ is implicitly given. Examples of commonly used kernel functions are Gaussian kernel $(k_\sigma(\boldsymbol{x}_i, \boldsymbol{x}_j) = e^{(-\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2/2\sigma^2)})$ and polynomial kernel $(k_d(\boldsymbol{x}_i, \boldsymbol{x}_j) = \langle \boldsymbol{x}_i, \boldsymbol{x}_j\rangle^d)$. One of the advantages of the so-called *kernel trick* is that new kernels can be constructed based on simpler kernels. For a detailed description on the design of kernel functions, we refer interested readers to [35].

3. Multi-task learning: the proposed framework

In the context of multi-task learning, we have T learning tasks on the same space $\mathcal{X}$, with $\mathcal{X} \subseteq \mathbb{R}^d$. For each task t we have m_t samples $\{\boldsymbol{x}_{1t}, \boldsymbol{x}_{2t}, \ldots, \boldsymbol{x}_{m_t t}\}$. In this section, we use the standard one-class SVM method for the purpose of multi-task learning. Our objective is to learn a decision function (a hyperplane) $f_t(\boldsymbol{x}) = \text{sign}(\langle \mathbf{w}_t, \phi(\boldsymbol{x})\rangle - \rho_t)$ for each task t. Based on different task relatedness assumptions, two formulations (named as formulation I and formulation II) are proposed. In the following, only a brief review of formulation I is given as it has appeared in our previous work [34]. We will focus on formulation II, which introduces the one-class SVM into a more general multi-task learning framework.

3.1. Formulation I

3.1.1. Assumption I

Inspired by the method proposed by Evgeniou and Pontil [16], we make a first assumption that when the tasks are related to each other, the normal vector $\mathbf{w}_t$ of the task model can be represented by the sum of a mean vector $\mathbf{w}_0$ and a specific vector $\mathbf{v}_t$ corresponding to each task:

$$\mathbf{w}_t = \mathbf{w}_0 + \mathbf{v}_t. \tag{4}$$

Following the above assumption, we can generalize the one-class SVM method to the problem of multi-task learning.

3.1.2. Primal problem

The primal optimization problem can be written as follows:

$$\begin{cases} \min\limits_{\mathbf{w}_0,\mathbf{v}_t,\xi_{it},\rho_t} & \frac{1}{2}\sum\limits_{t=1}^{T}\|\mathbf{v}_t\|^2 + \frac{\mu}{2}\|\mathbf{w}_0\|^2 + \sum\limits_{t=1}^{T}\left(\frac{1}{\nu_t m_t}\sum\limits_{i=1}^{m_t}\xi_{it}\right) - \sum\limits_{t=1}^{T}\rho_t \\ \text{subject to :} & \langle(\mathbf{w}_0 + \mathbf{v}_t), \phi(\boldsymbol{x}_{it})\rangle \geq \rho_t - \xi_{it}, \quad \xi_{it} \geq 0 \end{cases} \tag{5}$$

for all $i \in \{1, 2, \ldots, m_t\}$ and $t \in \{1, 2, \ldots, T\}$, where ξ_{it} are the slack variables associated to each sample and $\nu_t \in (0, 1]$ is the special

parameter of one-class SVM for task t. In this formulation, the positive regularization parameter μ controls the similarity between tasks. A large value of μ tends to enforce the system to learn the T tasks independently, whereas a small value of μ leads the system to learn a common model for all tasks. By setting the partial derivatives of the Lagrangian to zero and replacing the vectors $\mathbf{v}_t$ and $\mathbf{w}_0$ by $\mathbf{w}_t$ in the primal optimization function (5), we can obtain an equivalent optimization function:

$$\min_{\mathbf{w}_t,\xi_{it},\rho_t} \frac{\lambda_1}{2}\sum_{t=1}^{T}\|\mathbf{w}_t\|^2+\frac{\lambda_2}{2}\sum_{t=1}^{T}\left\|\mathbf{w}_t-\frac{1}{T}\sum_{r=1}^{T}\mathbf{w}_r\right\|^2 + \sum_{t=1}^{T}\left(\frac{1}{\nu_t m_t}\sum_{i=1}^{m_t}\xi_{it}\right)-\sum_{t=1}^{T}\rho_t \tag{6}$$

with

$$\lambda_1=\frac{\mu}{\mu+T} \quad \text{and} \quad \lambda_2=\frac{T}{\mu+T}. \tag{7}$$

For a detailed proof we refer interested readers to [34]. We can see that the objective of the primal optimization problem (5) in the framework of multi-task learning is therefore to find a trade-off between the maximization of the margin for each one-class SVM model and the closeness of each one-class SVM model to the average model.

3.1.3. Dual problem

The Lagrangian dual form of problem (5) is given by

$$\begin{cases} \max\limits_{\alpha_{it}} & -\frac{1}{2}\sum\limits_{t=1}^{T}\sum\limits_{r=1}^{T}\sum\limits_{i=1}^{m_t}\sum\limits_{j=1}^{m_t}\alpha_{it}\alpha_{jr}\left(\frac{1}{\mu}+\delta_{rt}\right)\langle\phi(\mathbf{x}_{it}),\phi(\mathbf{x}_{jr})\rangle \\ \text{subject to :} & 0\le\alpha_{it}\le\frac{1}{\nu_t m_t},\quad \sum\limits_{i=1}^{m_t}\alpha_{it}=1, \end{cases} \tag{8}$$

for all $i\in\{1,2,\ldots,m_t\}$ and $t\in\{1,2,\ldots,T\}$, with δ_{rt} being the Kronecker delta kernel:

$$\delta_{rt}=\begin{cases}1 & \text{if } r=t,\\ 0 & \text{if } r\neq t.\end{cases}$$

Note that the main difference between the dual problem (8) and that in a single one-class SVM learning (3) is the new term $(1/\mu+\delta_{rt})$ in the multi-task learning framework.

Let us define the kernel function: $k(\mathbf{x}_{it},\mathbf{x}_{jr})=\langle\phi(\mathbf{x}_{it}),\phi(\mathbf{x}_{jr})\rangle$, where r and t are the task indices associated to each sample. Using the kernel property that the tensor product of two kernels (δ_{rt} and $k(\mathbf{x}_{it},\mathbf{x}_{jr})$ here) is a valid kernel [35, Proposition 13.6, p. 410], we can deduce that the following function:

$$G_{rt}^{(1)}(\mathbf{x}_{it},\mathbf{x}_{jr})=\left(\frac{1}{\mu}+\delta_{rt}\right)k(\mathbf{x}_{it},\mathbf{x}_{jr})=\frac{1}{\mu}k(\mathbf{x}_{it},\mathbf{x}_{jr})+\delta_{rt}k(\mathbf{x}_{it},\mathbf{x}_{jr}) \tag{9}$$

is a linear combination of two valid kernels with positive coefficients ($1/\mu$ and 1), and therefore is also a valid kernel [35, Proposition 13.1, p. 408]. The main advantage of using the new kernel function $G_{rt}^{(1)}(\mathbf{x}_{it},\mathbf{x}_{jr})$ is that we can solve the multi-task learning optimization problem (5) by means of solving a single one-class SVM problem. Accordingly, we obtain the decision function for each task:

$$f_t(\mathbf{x})=\operatorname{sign}\left(\sum_{r=1}^{T}\sum_{i=1}^{m_t}\alpha_{ir}G_{rt}^{(1)}(\mathbf{x}_{ir},\mathbf{x})-\rho_t\right). \tag{10}$$

3.2. Formulation II

3.2.1. Assumption II

In this section, we propose a more general multi-task learning formulation, still based on one-class SVM. Similar to Assumption I [16] expressed in Eq. (4), we may assume that all the task models are close to a certain mean model, that is, each task model f_t is represented by the sum of a generic model g_0 and a specific model g_t:

$$f_t=g_0+g_t \tag{11}$$

$$f_t=\langle\mathbf{w}_0,\phi_0(\mathbf{x})\rangle+\langle\mathbf{v}_t,\phi_t(\mathbf{x})\rangle-\rho_t, \tag{12}$$

where $\mathbf{w}_0$ and ϕ_0 are the normal vector and the non-linear transformation for the generic model, respectively, and $\mathbf{v}_t$ and ϕ_t are those for the specific model. For the sake of notation simplicity, we use $-\rho_t$ instead of $-\rho_0-\rho_t$ in the model. Note that here $\phi_0\neq\phi_t$ and we define different non-linear feature mapping ϕ_t for different task t, while in Formulation I of Section 3.1 we have used the same non-linear transformation $\phi(\mathbf{x})$ for all the tasks. In particular, Formulation I can be considered as a special case of Formulation II. If we define $\phi_0=\phi_t$ in Eq. (12), then Formulation II reduces to Formulation I.

3.2.2. Primal problem

In this case, we obtain the following primal optimization problem:

$$\begin{cases} \min\limits_{\mathbf{w}_0,\mathbf{v}_t,\xi_{it},\rho_t} & \frac{1}{2}\sum\limits_{t=1}^{T}\|\mathbf{v}_t\|^2+\frac{\mu}{2}\|\mathbf{w}_0\|^2+\sum\limits_{t=1}^{T}\left(\frac{1}{\nu_t m_t}\sum\limits_{i=1}^{m_t}\xi_{it}\right)-\sum\limits_{t=1}^{T}\rho_t \\ \text{subject to :} & \langle\mathbf{w}_0,\phi_0(\mathbf{x}_{it})\rangle+\langle\mathbf{v}_t,\phi_t(\mathbf{x}_{it})\rangle\ge\rho_t-\xi_{it},\quad \xi_{it}\ge 0 \end{cases} \tag{13}$$

for all $i\in\{1,2,\ldots,m_t\}$ and $t\in\{1,2,\ldots,T\}$.

With the introduction of Lagrange multipliers $\alpha_{it},\beta_{it}\ge 0$, the Lagrangian can be expressed as follows:

$$\begin{aligned} &L(\mathbf{w}_0,\mathbf{v}_t,\xi_{it},\rho_t,\alpha_{it},\beta_{it})\\ &=\frac{1}{2}\sum_{t=1}^{T}\|\mathbf{v}_t\|^2+\frac{\mu}{2}\|\mathbf{w}_0\|^2+\sum_{t=1}^{T}\left(\frac{1}{\nu_t m_t}\sum_{i=1}^{m_t}\xi_{it}\right)-\sum_{t=1}^{T}\rho_t\\ &-\sum_{t=1}^{T}\sum_{i=1}^{m_t}\alpha_{it}[\langle\mathbf{w}_0,\phi_0(\mathbf{x}_{it})\rangle+\langle\mathbf{v}_t,\phi_t(\mathbf{x}_{it})\rangle-\rho_t+\xi_{it}]-\sum_{t=1}^{T}\sum_{i=1}^{m_t}\beta_{it}\xi_{it} \end{aligned} \tag{14}$$

The partial derivatives of the Lagrangian are set to zero, which lead to the following equations:

$$\begin{aligned} &\text{(a)}\mathbf{w}_0=\frac{1}{\mu}\sum_{t=1}^{T}\sum_{i=1}^{m_t}\alpha_{it}\phi_0(\mathbf{x}_{it})\\ &\text{(b)}\mathbf{v}_t=\sum_{i=1}^{m_t}\alpha_{it}\phi_t(\mathbf{x}_{it})\\ &\text{(c)}\alpha_{it}=\frac{1}{\nu_t m_t}-\beta_{it}\\ &\text{(d)}\sum_{i=1}^{m_t}\alpha_{it}=1 \end{aligned} \tag{15}$$

3.2.3. Dual problem

From Eq. (15) we may obtain the dual form of the Lagrangian as follows:

$$\begin{cases} \max\limits_{\alpha_{it}} & -\frac{1}{2}\sum\limits_{t=1}^{T}\sum\limits_{r=1}^{T}\sum\limits_{i=1}^{m_t}\sum\limits_{j=1}^{m_t}\alpha_{it}\alpha_{jr}\left(\frac{1}{\mu}\langle\phi_0(\mathbf{x}_{it}),\phi_0(\mathbf{x}_{jr})\rangle+\delta_{rt}\langle\phi_t(\mathbf{x}_{it}),\phi_t(\mathbf{x}_{jr})\rangle\right) \\ \text{subject to :} & 0\le\alpha_{it}\le\frac{1}{\nu_t m_t},\quad \sum\limits_{i=1}^{m_t}\alpha_{it}=1 \end{cases} \tag{16}$$

for all $i\in\{1,2,\ldots,m_t\}$ and $t\in\{1,2,\ldots,T\}$, with δ_{rt} being the Kronecker delta kernel and r and t the task indices. Now we assume that k_0 is the kernel function for the generic model and k_t is that for the specific model:

$$k_0(\mathbf{x}_{it},\mathbf{x}_{jr})=\langle\phi_0(\mathbf{x}_{it}),\phi_0(\mathbf{x}_{jr})\rangle, \tag{17}$$

$$k_t(\boldsymbol{x}_{it}, \boldsymbol{x}_{jr}) = \langle \phi_t(\boldsymbol{x}_{it}), \phi_t(\boldsymbol{x}_{jr}) \rangle. \tag{18}$$

According to the kernel properties presented in [35, Chapter 13], we may construct a novel kernel function:

$$G_{rt}^{(2)}(\boldsymbol{x}_{it}, \boldsymbol{x}_{jr}) = \frac{1}{\mu} k_0(\boldsymbol{x}_{it}, \boldsymbol{x}_{jr}) + \delta_{rt} k_t(\boldsymbol{x}_{it}, \boldsymbol{x}_{jr}). \tag{19}$$

Once again, the multi-task learning optimization problem can be solved by the algorithm of a single one-class SVM with the above new kernel function $G_{rt}^{(2)}(\boldsymbol{x}_{it}, \boldsymbol{x}_{jr})$ and finally the decision function is written as follows:

$$f_t(\boldsymbol{x}) = \text{sign}\left(\sum_{r=1}^{T} \sum_{i=1}^{m_t} \alpha_{ir} G_{rt}^{(2)}(\boldsymbol{x}_{ir}, \boldsymbol{x}) - \rho_t \right). \tag{20}$$

3.3. Summary

Fig. 1 illustrates the flowchart of the two proposed multi-task learning methods, denoted by MTL-OSVM I for the Formulation I presented in Section 3.1 and MTL-OSVM II for the Formulation II presented in Section 3.2. The two methods are depicted in parallel within a single chart, in order to better present their similarity and difference. We can see that starting from different relatedness assumptions, both methods introduce a regularization parameter μ in the optimization process to control the trade-off between the maximization of the margin for each one-class SVM model and the closeness of each one-class SVM model to the average model. Note that the relatedness assumption for MTL-OSVM II is a more flexible one and we can thus employ different kernel functions for the generic model and the specific model in the formulation, as shown in Eqs. (17) and (18). On the contrary, based on the assumption I in Eq. (4), we have to use a common kernel for the whole model. Thanks to the new kernel design, both proposed methods can be implemented easily through the optimization of a single one-class SVM.

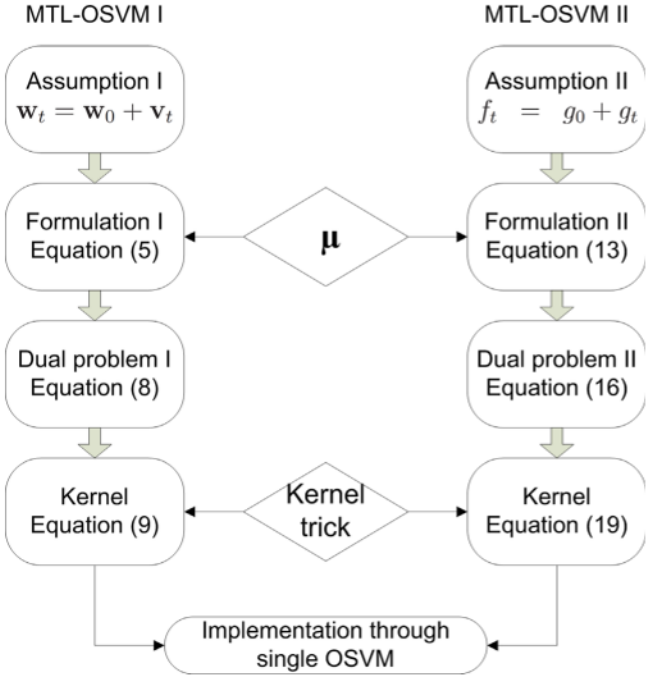


Fig. 1. Flowchart of the two proposed methods.

4. Experiments

We present in this section the experimental results of the proposed one-class SVM based multi-task learning approaches (MTL-OSVM I and MTL-OSVM II). Experiments have been conducted on both nonlinear toy data with low-dimensional feature space and textured image data with high-dimensional feature space. We address the problem of modeling the normal data (data of the target class) with the help of one-class SVM method, which is usually considered as an essential step for classification or outlier detection. In each experiment, related tasks are created in order to simulate applications of the modeling of normal data collected with different sensors or under different conditions (thus having different noises on the measurements). In order to evaluate the effectiveness of the proposed multi-task learning framework, we compare our approaches not only with a number of multi-task learning methods, but also with two other learning strategies: the traditional learning method that learns the T tasks independently each with a one-class SVM (denoted by T-OSVM) and the method that considers all the related tasks as one big task (denoted by 1-OSVM). For the sake of simplicity, we assume in our experiments that all the tasks have the same number of samples, so m_t is substituted by m. In order to ensure the reliability of the performance evaluation, all the results have been averaged over 20 trials each with random draws of training set.

4.1. Nonlinear toy data

4.1.1. Data description

We have firstly tested the proposed methods on four (T=4) related simple nonlinear classification tasks. The datasets are created according to the following steps. For the first task, each sample $\boldsymbol{x}_{i1} = [x_{i1}^{(1)}, x_{i1}^{(2)}, x_{i1}^{(3)}, x_{i1}^{(4)}]$ is composed of d=4 variables and generated by the following model:

$$\begin{aligned} & x_{i1}^{(j)} \sim \mathcal{U}(0,1), \quad j = 1,2,3 \\ & x_{i1}^{(4)} = x_{i1}^{(1)} + 2x_{i1}^{(2)} + (x_{i1}^{(3)})^2 \end{aligned} \tag{21}$$

where $\mathcal{U}(0,1)$ is the standard uniform distribution on the open interval $(0,1)$ and i is the index of sample. The datasets for the other three tasks are then created by adding Gaussian white noises with different amplitudes on the dataset of the first task. The added Gaussian noises are classified as low noise (for Task 2, with $\mathcal{N}(0, 0.01^2)$), medium noise (for Task 3, with $\mathcal{N}(0, 0.08^2)$) and high noise (for Task 4, with $\mathcal{N}(0, 0.15^2)$). In order to evaluate the false positive error rates, we have generated a set of negative samples that are composed of d=4 uniformly distributed variables on $(0,1)$. Therefore, the training set of each task contains only positive samples (m=200), whereas in the test procedure we use the test set of size 400 that contains both positive and negative samples (200 samples for each class).

4.1.2. Evaluation of MTL-OSVM I

1. *Parameter setting*: In our experiments for the analysis of MTL-OSVM I, the kernel used in T-OSVM and 1-OSVM is a Gaussian kernel. For the proposed multi-task learning method MTL-OSVM I, the new kernel is thus constructed based on the Gaussian kernel as presented in Eq. (9). The optimum values for the two parameters ν and σ of the one-class SVM are determined through cross validation. For the sake of simplicity, we have used a common combination of their values (ν, σ) for all related tasks. The obtained optimum parameter values of one-class SVM are $(\nu, \sigma) = (0.01, 0.5)$ for this experiment. As the approaches are all one-class classification methods, the statistics of both false positive and false negative error rates are reported.

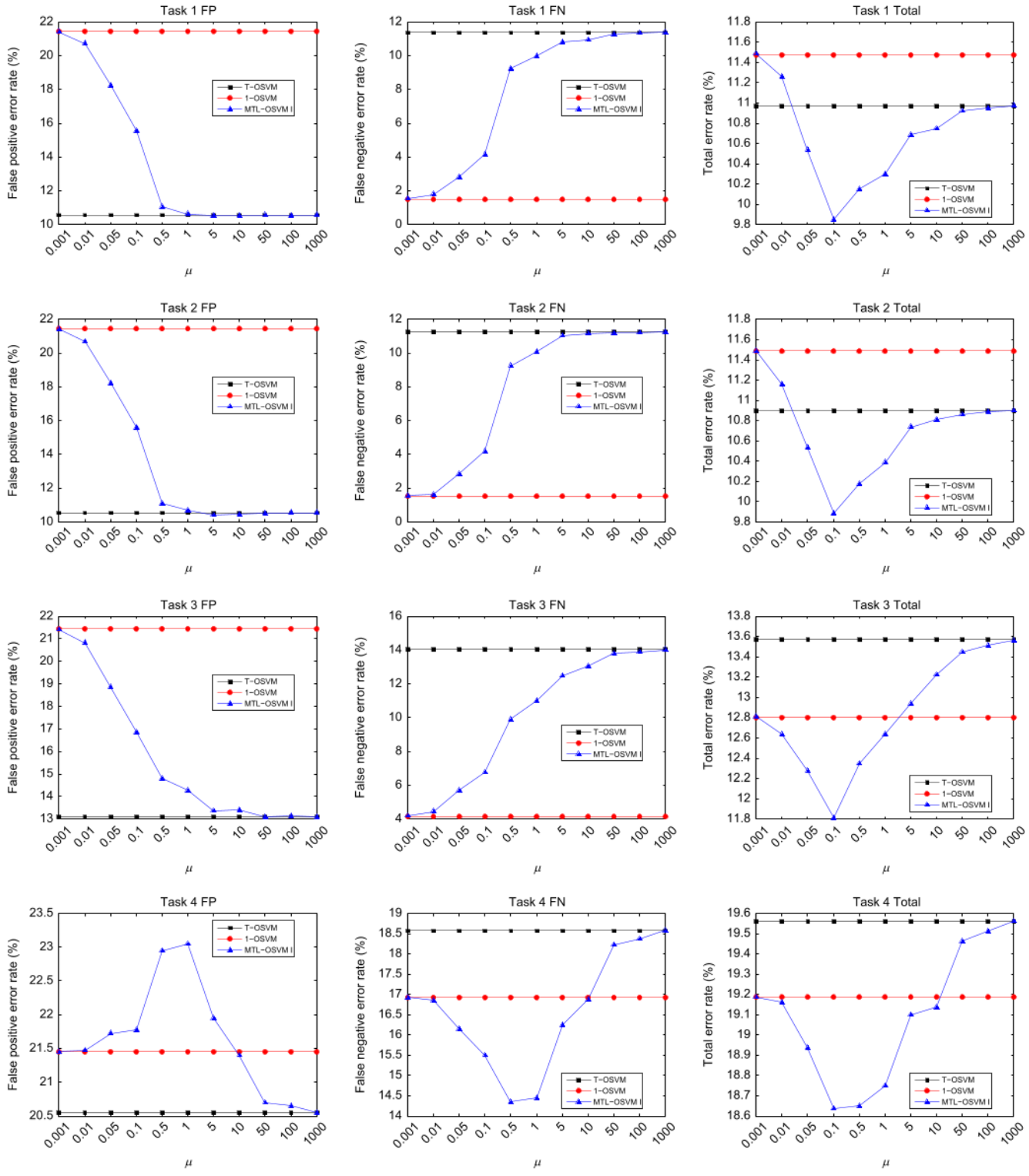


Fig. 2. The variation of the average false positive (FP), false negative (FN) and total error rates (Total) of different methods (*T*-OSVM, 1-OSVM and MTL-OSVM I) for each task (nonlinear toy data), along with the value change of the regularization parameter μ.

2. *Results*: Fig. 2 illustrates the variation of the average false positive, false negative and total error rates of our multi-task learning method MTL-OSVM I for each task, along with the value change of the regularization parameter μ. The error rates of *T*-OSVM and 1-OSVM are also presented. We can see that for a very small value of μ, the performance of MTL-OSVM I coincides with that of 1-OSVM as if all the tasks were considered as the same task. When the value of μ is very large, the performance of MTL-OSVM I is in accordance with that of the traditional independent learning method *T*-OSVM. With the increase of the value of μ, the behaviors of the first three tasks are similar. The false positive error rate of the MTL-OSVM I method tends to decrease, whereas its false negative error rate tends to increase. However, for the fourth task, the false positive (false negative) error rate first increases (decreases) and then decreases (increases) after it reaches the maximum (minimum) value. This behavior may be due to the very

high noise that we have added to the original dataset. With a good choice of μ (*e.g.* $\mu = 0.1$), the multi-task framework achieves a better performance in terms of the total error rate (see the third column in Fig. 2) when compared to the traditional learning methods.

4.1.3. Evaluation of MTL-OSVM II

1. *Parameter setting*: For the second multi-task learning approach MTL-OSVM II, two basic kernel functions: k_0 (Eq. (17)) in the generic model and k_t (Eq. (18)) in the specific model, need to be defined. We may use the Gaussian kernel with different values of σ or polynomial kernels with different degrees. In this section we present results with $k_0(\boldsymbol{x}_{it}, \boldsymbol{x}_{jr}) = \langle \boldsymbol{x}_{it}, \boldsymbol{x}_{jr} \rangle^{d_0}$ a polynomial function of degree d_0 and $k_t(\boldsymbol{x}_{it}, \boldsymbol{x}_{jr}) = \langle \boldsymbol{x}_{it}, \boldsymbol{x}_{jr} \rangle^{d_t}$ a polynomial function of degree $d_t > d_0$. It is reasonable to choose a higher degree for the specific model than for the generic one since the specific model for each task is generally more complicated than the generic one. Note that, in this section, the kernel of the one-class SVM used for *T*-OSVM, 1-OSVM and MTL-OSVM I is also a polynomial kernel with degree d_0.

In order to determine the optimum degree value of each polynomial function for the kernel $G^{(2)}$, we have evaluated the performance of MTL-OSVM II for different combinations of d_0 and d_t. To do this, a series of experiments have been conducted with d_0 fixed ($d_0 \in \{1, 2, \ldots, 25\}$) and d_t varied from d_0+1 to 26. The optimum values of d_0 and d_t are determined as the one that gives the minimum total error rate. Fig. 3 shows the total error rate for task 1 of different methods (*T*-OSVM, 1-OSVM, MTL-OSVM I and MTL-OSVM II) with $d_0 = 1$ and $d_t \in \{2, 3, \ldots, 26\}$. Note that for the two proposed methods, MTL-OSVM I and MTL-OSVM II, only the error rates obtained with the optimum value of μ are presented. We can see that as the value of d_t increases, the total error rate of MTL-OSVM II first decreases and then increases showing a minimum at $d_t = 7$. This observation suggests that effect of overfitting occurs when the degree of polynomial is too high. This behavior is common for other fixed values of d_0. We can thus obtain a series of optimum combinations of d_0 and d_t.

2. *Results*

Influence of μ: We have first tested the effect of the regularization parameter μ in MTL-OSVM II. Fig. 4 illustrates results of different methods (*T*-OSVM, 1-OSVM and MTL-OSVM II) for each task with different values of μ. In this figure, we have $d_0 = 1, d_t = 2$.

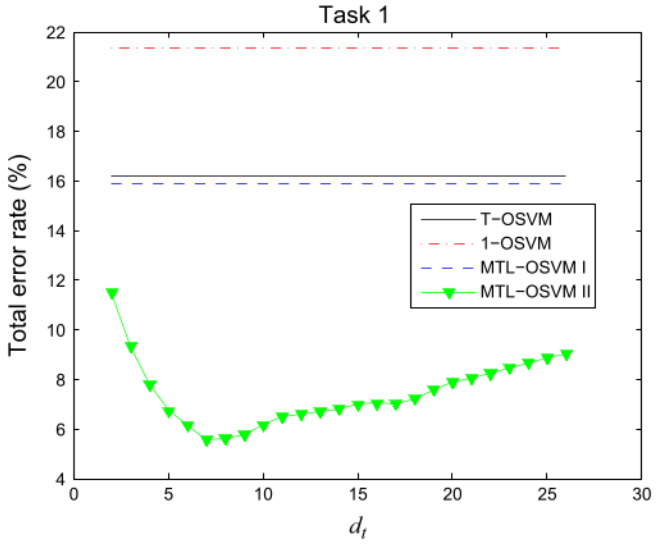


Fig. 3. The average total error rate value of MTL-OSVM II for task 1 (nonlinear toy data) in function of d_t, with $d_0 = 1$ and optimal value of the regularization parameter μ.

The behaviors are common for all the other combinations of d_0 and d_t. Similar to the results of MTL-OSVM I (see Fig. 2), we can see that with a good choice of μ (*e.g.* $\mu = 0.1$), the multi-task learning method MTL-OSVM II shows significant improvement in terms of total error rate over the traditional learning methods. For the first three tasks when μ is very small (respectively large), the error rates of MTL-OSVM II have the tendency to coincide with those of 1-OSVM (respectively those of *T*-OSVM). The fourth task, which has high noise on the dataset, has a slightly different behavior. For the first three tasks the variations of the false positive and false negative error rates of MTL-OSVM II are no longer monotone and the values of these two error rates are not completely bounded between those of *T*-OSVM and 1-OSVM. It appears that the regularization parameter μ has a smaller influence in MTL-OSVM II than in MTL-OSVM I. This can be explained by the fact that MTL-OSVM II uses two different kernels (k_0 and k_t) for constructing $G^{(2)}$, whereas MTL-OSVM I employs a single kernel (k) for $G^{(1)}$.

Influence of d_0 *and* d_t: Fig. 5 presents the variation of the best average total error rate of different methods, along with the value change of d_0 for the four tasks. When d_0 is small, we observe that MTL-OSVM II outperforms significantly the other three methods and 1-OSVM performs the worst. With increasing degree d_0, the performance of MTL-OSVM II does not change greatly. However, the error rates of all the other three methods decrease sharply at first ($1 \le d_0 \le 7$) and then increase slightly after they reach the minimum value. For $d_0 \ge 13$, it appears that MTL-OSVM I tends to outperform MTL-OSVM II. This may be explained by the fact that when d_0 is large, even with a larger value of d_t, we may encounter the effect of overfitting. Besides, a higher degree in the polynomial kernel usually increases the computational cost. A lower degree value is thus preferred. We may always fix $d_0 = 1$ and then find an appropriate value of d_t with validation set. As illustrated in Fig. 3, for this dataset, we may simply choose the combination $(d_0, d_t) = (1, 7)$.

4.2. Textured image data

The two proposed methods were also tested on several textured gray-scale images that contain textures generated by using Markov chain models [36]. Many examples in textile quality control can be found where multi-task learning can be beneficial. For example, it is usually difficult or even impossible to obtain a perfect training set of texture images due to the presence of sensor noise, incorrect set of sensor parameters or oblique position of sensors. We may consider the learning of data that were collected from different sensors as a different task. The objective of this experiment is to show that within the multi-task learning framework, imperfect datasets can be helpful to improve the performance of outlier detection.

According to the nature of a texture, we suppose that the useful information for texture characterization is included in an isotropic neighborhood of each pixel. In our experiments we use the gray levels of a local $d = 5 \times 5$ squared window centered to each pixel as its feature vector. Similar to the previous experiment in Section 4.1, four related tasks are created. The dataset for Task 1 contains samples with high feature dimension ($d = 5 \times 5 = 25$) that are selected randomly from the original single texture source image. The samples for the other three tasks are selected from textured images of the same source as Task 1, but contaminated by low Gaussian noise (Task 2, with $\mathcal{N}(0, 5^2)$), medium Gaussian noise (Task 3, with $\mathcal{N}(0, 15^2)$) and high Gaussian noise (for Task 4, with $\mathcal{N}(0, 35^2)$). Negative samples used in the test set are generated by using a different single texture source image. Fig. 6 illustrates the single texture source images used for generating the datasets. In each trial, the training set of each task contains $m = 200$ positive samples and the test set is composed of 200 positive and 200

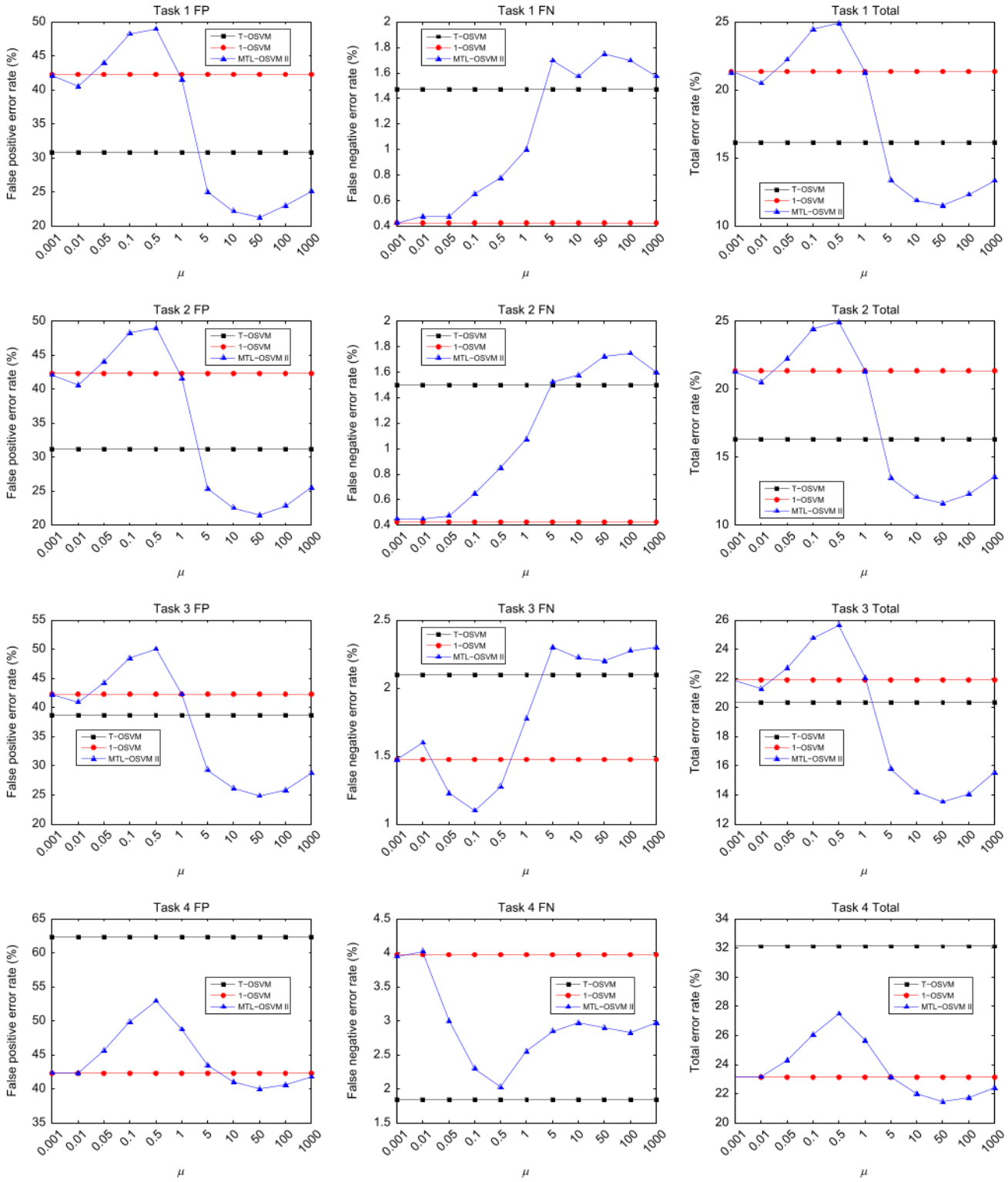


Fig. 4. The variation of the average false positive (FP), false negative (FN) and total error rates (Total) of different methods (*T*-OSVM, 1-OSVM and MTL-OSVM II) for each task (nonlinear toy data), along with the value change of μ, with $d_0 = 1, d_t = 2$.

negative samples. The common parameter value of one-class SVM used in this experiment is $\nu = 0.01$.

Besides *T*-OSVM and 1-OSVM, we also compare the two proposed methods with MTL-OC of Yang et al. [18] and MTL-FEAT of Argyriou et al. [9]. Readers could refer to Section 1 for brief descriptions of MTL-OC. The MTL-FEAT method has been developed for learning a common sparse representation for related tasks based on an optimization problem with a mixed $(2,1)$-norm regularizer. MTL-FEAT was initially proposed for regression problems. We may extend the method to one-class classification by introducing a threshold interval. That is, when the regression function value is in the predefined threshold interval, the query sample belongs to the normal class. In order to conduct fair reasonable comparisons with our one-class classification methods, we report the results of MTL-FEAT by setting an appropriate threshold interval that results in the same false positive (or false

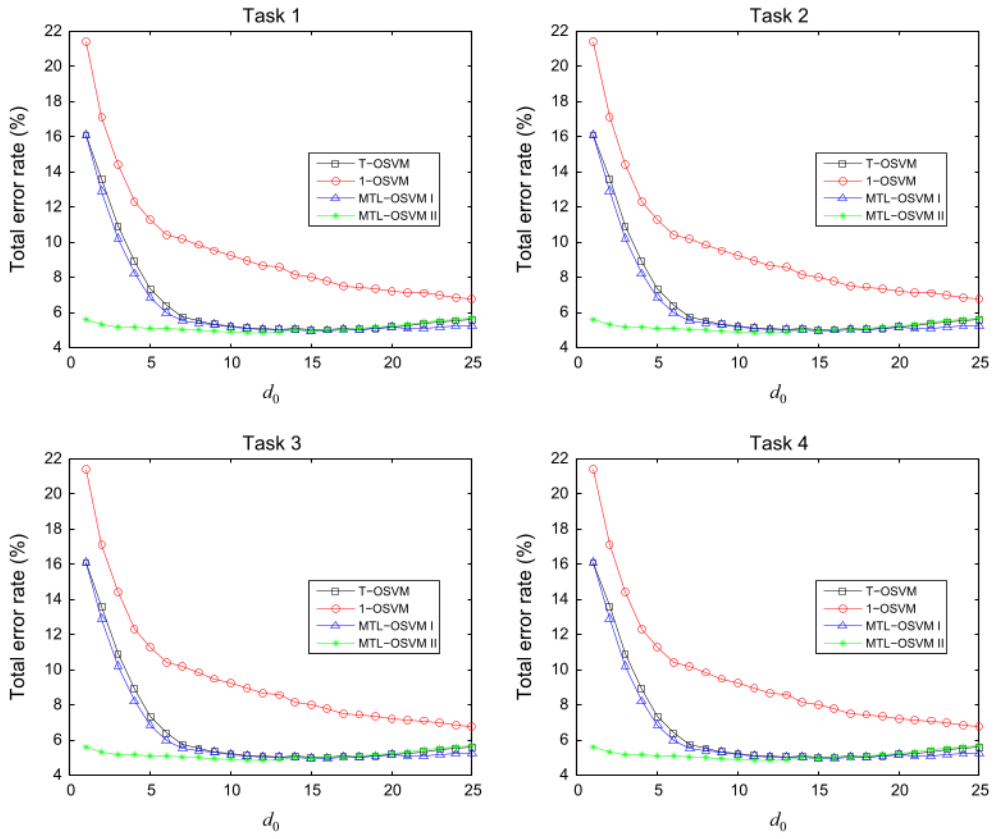


Fig. 5. The variation of the best average total error rate of different methods, along with the value change of d_0 for each task.

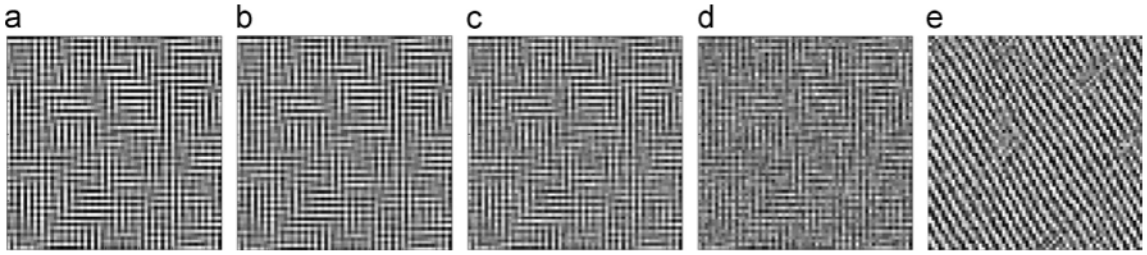


Fig. 6. Single texture source images used for generating the training and testing datasets. (a) Original texture image for Task 1. (b) Texture image of (a) with low noise, for Task 2. (c) Texture image of (a) with medium noise, for Task 3. (d) Texture image of (a) with high noise, for Task 4. (e) Original texture image for generating negative samples.

Table 1
Parameter settings and training time of different methods.

Methods	Kernel	Kernel parameter	Time (s)
T-OSVM	Gaussian	$\sigma=300$	0.0175
1-OSVM	Gaussian	$\sigma=300$	0.0159
MTL-OSVM I	Gaussian	$\sigma=300$	0.0192
MTL-OSVM II	Polynomial	$(d_0, d_t)=(1,6)$	0.0426
MTL-OC	Gaussian	$\sigma=300$	636
MTL-FEAT	Linear	–	0.0813

negative) value as the method under comparison once the regression function is learned.

In order to determine the appropriate kernel (we tested both Gaussian and polynomial kernels) and parameter values for each method, a series of experiments similar to those in Section 4.1 have been conducted on a validation set. The resulted optimum settings, which minimize the average total error rate of the four tasks on the validation set, are presented in Table 1. The Gaussian kernel is therefore employed for *T*-OSVM, 1-OSVM, MTL-OSVM I and MTL-OC, whereas the polynomial kernel provides the best results for MTL-OSVM II. For the MTL-FEAT method, a linear kernel is used. The last column of Table 1 lists the average training time of each method with the fixed parameter values. Note that here we only present a comparison with MTL-OC by using $m=50$ training samples per task due to the overwhelming time and space consumption of the MTL-OC method when $m>50$. All experiments were conducted under the Matlab software on a PC with an Intel i7 2 GHz processor. We can see that 1-OSVM is the fastest among all the methods. Taking the training time of 1-OSVM as a unit measure, the differences between 1-OSVM, *T*-OSVM and MTL-OSVM I are small. MTL-OSVM II needs a little more time for training, about 2.68 times of 1-OSVM. MTL-FEAT carries out the common feature selection in addition to the training, so its training time is longer, about five times of 1-OSVM. Only MTL-OC stays far behind due to the complexity of the optimization procedure. It needs about 40,000 times of that for 1-OSVM, making this method impractical for real applications with large training sets.

Table 2 shows the best obtained results (both the means and the standard deviations of the obtained error rates) of all the methods under comparison with $m=50$ training samples. The best

Table 2
Statistics of the error rate (%) of different methods for all tasks on texture data with 50 training samples per task. FP: false positive error rate, FN: false negative error rate, Total: total error rate. The FN of MTL−FEAT$_{(FN)}$ is shown in italic, which has been manually tuned to be close to the smallest FN of methods under comparison.

Methods	FP	FN	Total	μ
Task 1				
T-OSVM	1.62 ± 0.92	50.4 ± 5.3	26.0 ± 2.7	–
1-OSVM	14.1 ± 3.4	**23.7** ± 6.9	18.9 ± 4.0	–
MTL-OSVM I	14.0 ± 0.8	23.8 ± 5.1	18.9 ± 2.7	0.001
MTL-OSVM II	**0.375** ± 0.393	50.7 ± 5.5	25.5 ± 2.8	10^4
MTL-OC	1.70 ± 0.85	49.8 ± 5.2	25.7 ± 2.8	–
MTL−FEAT$_{(FN)}$	11.3 ± 3.9	*23.8* ± 11.2	**17.6** ± 5.1	–
Task 2				
T-OSVM	1.72 ± 0.89	51.5 ± 5.7	26.4 ± 2.9	–
1-OSVM	14.1 ± 3.4	**24.3** ± 7.4	**19.2** ± 4.2	–
MTL-OSVM I	14.0 ± 0.8	24.4 ± 5.3	**19.2** ± 2.8	0.001
MTL-OSVM II	**0.425** ± 0.373	51.3 ± 6.4	25.9 ± 3.2	10^4
MTL-OC	1.73 ± 0.85	50.2 ± 5.4	26.0 ± 2.8	–
MTL−FEAT$_{(FN)}$	17.6 ± 4.7	*24.5* ± 9.8	21.1 ± 5.0	–
Task 3				
T-OSVM	2.62 ± 1.38	54.3 ± 6.2	28.4 ± 3.3	–
1-OSVM	14.1 ± 3.4	**29.8** ± 8.2	**21.9** ± 4.2	–
MTL-OSVM I	14.0 ± 1.4	30.0 ± 6.2	**21.9** ± 3.3	0.001
MTL-OSVM II	**0.675** ± 0.467	53.1 ± 7.3	26.9 ± 3.7	10^4
MTL-OC	3.05 ± 1.56	51.5 ± 6.6	27.3 ± 3.5	–
MTL−FEAT$_{(FN)}$	23.6 ± 5.7	*29.6* ± 11.9	26.6 ± 5.9	–
Task 4				
T-OSVM	14.0 ± 3.8	63.5 ± 8.2	38.8 ± 3.8	–
1-OSVM	14.1 ± 3.4	62.8 ± 8.0	38.4 ± 3.8	–
MTL-OSVM I	16.3 ± 3.3	56.7 ± 7.9	36.5 ± 3.7	1
MTL-OSVM II	**6.42** ± 3.12	61.3 ± 9.8	**33.8** ± 4.6	10^4
MTL-OC	19.1 ± 4.9	**51.6** ± 8.9	35.3 ± 4.6	–
MTL−FEAT$_{(FN)}$	37.1 ± 7.3	*51.5* ± 16.6	44.3 ± 7.8	–

results are labeled in bold. Note that μ is a special regularization parameter of our proposed methods, the symbol "–" in Table 2 means that the corresponding method in the same line has no such parameter. According to this table, we can see that in general *T*-OSVM and MTL-OSVM II have the lowest false positive error rate but a higher false negative error rate. On the contrary, 1-OSVM and MTL-OSVM I achieve the lowest false negative rate at the expense of a higher false positive rate. From the table we can see that a relative small false negative usually leads to a better performance (*i.e.* small total error). Therefore, for the MTL-FEAT method, we tune manually the threshold interval providing false negative that is close to the smallest one for each task and we note in the table as MTL−FEAT$_{(FN)}$ (for MTL-FEAT with fixed FN values). The obtained results show that MTL−FEAT$_{(FN)}$ achieves the best performance for task 1 but the worst for task 4. Meanwhile, a large variance is observed for MTL−FEAT$_{(FN)}$. For the first three tasks, *T*-OSVM that learns each task individually performs the worst. 1-OSVM and MTL-OSVM I with a small value of μ have very close results. Both of them perform significantly better than the other methods. This implies that we may learn all related tasks as a single task in order to improve the system's performance when only a few training samples are available (in our case, the number of samples is 50). For the fourth task that differs a lot from the other tasks, MTL-OSVM II performs the best and 1-OSVM turns to be a bad choice.

Fig. 7 shows the variation of the average total error rate for the proposed methods when we change the number of training samples ($m = 50, 100, 150, 200$). We can see that *T*-OSVM performs always the worst. With the increase of the number of training samples, the error rate of all the methods decreases except for 1-OSVM. This observation indicates that considering all related tasks as a single task (as 1-OSVM) may result in deterioration of

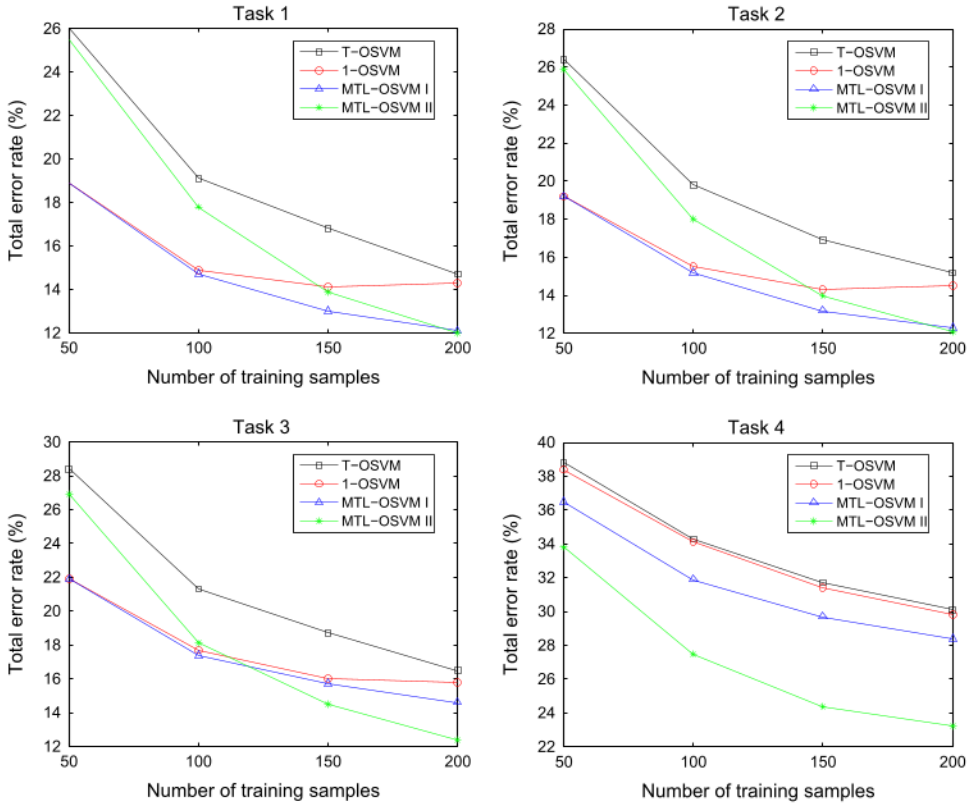


Fig. 7. Total error rate for different methods *versus* number of training samples.

performance when we have enough training samples. When many training samples are available, say $m \geq 100$, our methods lead to more interesting results than 1-OSVM which learns a single model for all tasks. The MTL-OSVM II method that uses a combination of two polynomial functions has the best overall performance. Under task 1 and task 2, the difference between MTL-OSVM I and MTL-OSVM II is small. For task 3 and task 4 MTL-OSVM II performs much better. Another point is that when the number of training samples is small (*e.g.* $m=50$, see the last column of Table 2), we have to use different values of μ for different tasks (especially for task 4) in order to obtain the best performance for our methods. However, with $m \geq 200$ training samples, the best results of MTL-OSVM I and MTL-OSVM II are obtained with a fixed value of μ for all the tasks (*e.g.* $m=200$, $\mu = 0.1$ for MTL-OSVM I and $\mu = 10^4$ for MTL-OSVM II). This implies that with the help of multi-task learning framework, significant improvement of the system performance not only can be obtained with small sample size compared to single-task learning *T*-OSVM, but also can be guaranteed when more samples are available.

5. Conclusion

We have presented two multi-task learning methods that learn a pool of related one-class SVM classifiers, each for a task, simultaneously. The first one is based upon the assumption that the model parameters of different one-class SVMs are close to a certain mean value. In the second method, we assume that the models of related one-class SVMs are close to a certain mean function, which leads to a more general assumption than in the first one. We show that with the help of kernel design techniques, both methods can be reformulated to a single one-class SVM optimization problem and therefore are easy to implement. We have reported experiments on a set of one-class classification problems, including toy dataset and textured images. We have shown that even with a common setting of model and kernel parameters, the proposed multi-task learning frameworks are usually more effective than a single-task learning strategy. In addition, the proposed multi-task learning methods are dynamic. That is, when tasks are related to each other, an appropriate value of the regularization parameter μ in the optimization problem may lead to improvements of the system performance. On the contrary, if tasks are independent, then a large value of μ will enforce the system to learn tasks independently, and can thus preserve the system performance. Future work includes different parameter settings for different tasks. Automatic determination of the regularization parameter μ in the formulation remains another open question.

It is possible to extend the proposed methods to the nonlinear multi-output regression problems, for example, we may extend our methods to the SVM multiregression approach in multiple-input multiple-output systems (proposed by [37]). It would be an interesting idea to study the nonlinear multi-output regression problems in the framework of multi-task learning. We may consider the learning of different channels of the multi-output regression problem as different tasks. The only issue is whether these tasks are related. If the tasks are related to each other, then using the proposed multi-task learning methods may explore the relatedness between tasks and thus improve the performance. On the contrary, if the tasks are independent, then by setting an appropriate value of the regularization parameter (the parameter μ) in the formulation, each task (*i.e.* each channel) can be trained independently, which can at least preserve the system performance. Again, this requires the automatic determination of the regularization parameter μ in the formulation.

Acknowledgments

This research is supported by GIS 3SGS (Groupement d'Intérêt Scientifique "Surveillance, Sûreté, Sécurité des Grands Systèmes"). The authors would like to thank Dr. Haiqin Yang for kindly providing us the Matlab code of MTL-OC method.

References

[1] J. Bi, T. Xiong, S. Yu, M. Dundar, R.B. Rao, An improved multi-task learning approach with applications in medical diagnosis, in: Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases – Part I, Antwerp, Belgium, 2008, pp. 117–132.

[2] V.W. Zheng, S.J. Pan, Q. Yang, J.J. Pan, Transferring multi-device localization models using latent multi-task learning, in: Proceedings of the Twenty-third National Conference on Artificial Intelligence, Chicago, IL, USA, 2008, pp. 1427–1432.

[3] S.J. Pan, Q. Yang, A survey on transfer learning, IEEE Trans. Knowl. Data Eng. 22 (10) (2010) 1345–1359.

[4] S.R. Ozawa, D.A. Roussinov, A multitask learning model for online pattern recognition, IEEE Trans. Neural Netw. 20 (3) (2009) 430–445.

[5] A. Birlutiu, P. Groot, T. Heskes, Multi-task preference learning with an application to hearing aid personalization, Neurocomputing 73 (2010) 1177–1185.

[6] L. Du, P. Wang, H. Liu, M. Pan, F. Chen, Z. Bao, Bayesian spatiotemporal multitask learning for radar HRRP target recognition, IEEE Trans. Signal Process. 59 (7) (2011) 3182–3196.

[7] R. Caruana, Multitask learning, Mach. Learn. 28 (1997) 41–75.

[8] T. Jebara, Multi-task feature and kernel selection for SVMS, in: Proceedings of the Twenty-first International Conference on Machine Learning, Banff, Alberta, Canada, 2004.

[9] A. Argyriou, T. Evgeniou, M. Pontil, Convex multi-task feature learning, Mach. Learn. 73 (2008) 243–272.

[10] Q. Gu, J. Zhou, Learning the shared subspace for multi-task clustering and transductive transfer classification, in: Proceedings of the Ninth IEEE International Conference on Data Mining, Miami, FL, USA, 2009, pp. 159–168.

[11] G. Obozinski, B. Taskar, M.I. Jordan, Joint covariate selection and joint subspace selection for multiple classification problems, Stat. Comput. 20 (2010) 231–252.

[12] H. Yang, I. King, M.R. Lyu, Online learning for multi-task feature selection, in: Proceedings of the Nineteenth International Conference on Information and Knowledge Management, Toronto, Canada, 2010, pp. 1693–1696.

[13] S. Ben-David, R. Schuller, Exploiting task relatedness for multiple task learning, in: Proceedings of the Sixteenth Annual Conference on Computational Learning Theory and the Seventh Kernel Workshop, Washington DC, USA, 2003, pp. 567–580.

[14] S. Ben-David, R.S. Borbely, A notion of task relatedness yielding provable multiple-task learning guarantees, Mach. Learn. 73 (2008) 273–287.

[15] T. Evgeniou, C.A. Micchelli, M. Pontil, Learning multiple tasks with kernel methods, J. Mach. Learn. Res. 6 (2005) 615–637.

[16] T. Evgeniou, M. Pontil, Regularized multi-task learning, in: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Seattle, WA, USA, 2004, pp. 109–117.

[17] C. Widmer, N. Toussaint, Y. Altun, G. Ratsch, Inferring latent task structure for multitask learning by multiple kernel learning, BMC Bioinformat. 11 (Suppl. 8) (2010) S5.

[18] H. Yang, I. King, M.R. Lyu, Multi-task learning for one-class classification, in: Proceedings of the International Joint Conference on Neural Networks, Barcelona, Spain, 2010, pp. 1–8.

[19] Y. Ji, S. Sun, Y. Lu, Multitask multiclass privileged information support vector machines, in: Proceedings of the Twenty-first International Conference on Pattern Recognition, 2012, pp. 2323–2326.

[20] X. Xie, S. Sun, Multitask twin support vector machines, in: Neural Information Processing, Lecture Notes in Computer Science, vol. 7664, Springer, Berlin Heidelberg, 2012, pp. 341–348.

[21] Y. Ji, S. Sun, Multitask multiclass support vector machines: model and experiments, Pattern Recognit. 46 (3) (2013) 914–924.

[22] B.E. Boser, I.M. Guyon, V.N. Vapnik, A training algorithm for optimal margin classifiers, in: Proceedings of the Fifth Annual Workshop on Computational Learning Theory, Pittsburgh, PA, USA, 1992, pp. 144–152.

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

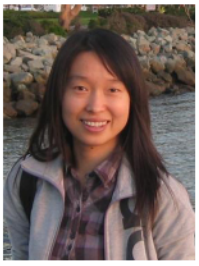
[24] K.M. Chai, Generalization errors and learning curves for regression with multi-task Gaussian processes, in: Advances in Neural Information Processing Systems 22, 2009, pp. 279–287.

[25] T. Heskes, Empirical Bayes for learning to learn, in: Proceedings of the Seventeenth International Conference on Machine Learning, San Francisco, CA, USA, 2000, pp. 367–374.

[26] G. Skolidis, G. Sanguinetti, Bayesian multitask classification with Gaussian process priors, IEEE Trans. Neural Netw. 22 (12) (2011) 2011–2021.

[27] B. Schölkopf, J.C. Platt, J. Shawe-Taylor, A.J. Smola, R.C. Williamson, Estimating the support of a high-dimensional distribution, Neural Comput. 13 (7) (2001) 1443–1471.

[28] D.M.J. Tax, R.P.W. Duin, Support vector domain description, Pattern Recognit. Lett. 20 (11–13) (1999) 1191–1199.
[29] L. Tarassenko, P. Hayton, N. Cerneaz, M. Brady, Novelty detection for the identification of masses in mammograms, in: Proceedings of the Fourth IEEE International Conference on Artificial Neural Networks, Cambridge, UK, 1995, pp. 442–447.
[30] G. Ritter, M.T. Gallegos, Outliers in statistical pattern recognition and an application to automatic chromosome classification, Pattern Recognit. Lett. 18 (1997) 525–539.
[31] E. Eskin, Anomaly detection over noisy data using learned probability distributions, in: Proceedings of the Seventeenth International Conference on Machine Learning, San Francisco, CA, USA, 2000, pp. 255–262.
[32] S. Singh, M. Markou, An approach to novelty detection applied to the classification of image regions, IEEE Trans. Knowl. Data Eng. 16 (4) (2004) 396–407.
[33] T. Kato, H. Kashima, M. Sugiyama, K. Asai, Multi-task learning via conic programming, in: Advances in Neural Information Processing Systems, vol. 20, 2008, pp. 737–744.
[34] X. He, G. Mourot, D. Maquin, J. Ragot, P. Beauseroy, A. Smolarz, E. Grall-Maës, One-class svm in multi-task learning, in: Advances in Safety, Reliability and Risk Management, ESREL 2011, Troyes, France, 2011, pp. 486–494.
[35] B. Schölkopf, A.J. Smola, Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond, MIT Press, Cambridge, MA, USA, 2001.
[36] A. Smolarz, Etude qualitative du modèle auto-binomial appliqué à la synthèse de texture, in: XXIXèmes Journées de Statistique, Carcassonne, France, 1997, pp. 712–715.
[37] M. Sánchez-Fernández, M. de Prado-Cumplido, J. Arenas-García, F. Pérez-Cruz, SVM multiregression for nonlinear channel estimation in multiple-input multiple-output systems, IEEE Trans. Signal Process. 52 (8) (2004) 2298–2307.



Xiyan He received in 2006 the Generalist Engineer degree from Ecole Centrale Paris, France, and the M.E. degree in Pattern Recognition and Intelligent System from Xi'an Jiaotong University, China. She received her Ph.D. degree in Computer Science in 2009, from University of Technology of Troyes, France. She was a Teaching Assistant in University of Technology of Troyes in 2009, a Post-doctoral Research Fellow in Research Centre for Automatic Control of Nancy in 2010, and a Teaching Assistant in University of Pierre-Mendès-France, Grenoble in 2011. Since 2012, she has been a Post-doctoral Research Fellow in Grenoble Laboratory of Image, Speech, Signal and Automatics. Her main research interests include machine learning, pattern recognition and data fusion, with special focus on applications to remote sensed images.

Gilles Mourot received his Ph.D. in Electrical Engineering in 1993 from Institut National Polytechnique de Lorraine, Nancy, France. His research activities, within Automatic Control Research Center of Nancy (CRAN), include data-driven fault diagnosis based on machine learning and multivariate statistical analysis methods.

Didier Maquin is a Professor of Automatic Control at the University of Lorraine, France. He teaches Automatic Control and Applied Mathematics in various engineering schools. He is a member of the Research Center for Automatic Control of Nancy (CRAN). From January 2005 to December 2012, he was the Scientific Leader of one of the five CRAN research groups entitled Dependability and system diagnosis which gathered about 20 teachers and researchers and as many Ph.D. students. From a national point of view, he is involved in the French Automatic Control Research Group. He has been on its directorate staff since 2001 and currently serves as the head of the theme Dependability, Supervision and Maintenance. Didier Maquin has co-authored around 50 journal articles and 140 conference communications.

José Ragot received the "Diplome d'Ingénieur" (Engineer's Degree) with specialization in control from the Ecole Centrale de Nantes, France, in 1969. Then, he joined the University of Nancy, France, where he received the "Diplôme d'Etudes Approfonfies" (Masters' Degree) in 1970. In 1973 he obtained the "Diplôme de Doctorat" (Ph.D. degree) and in 1980 the "Diplôme de Doctorat-es-Science". Since 1985, José Ragot has been a full Professor at the Institut National Polytechnique de Lorraine (INPL, National Polytechnic Institute of Lorraine) and a researcher in the Centre de Recherche en Automatique de Nancy (CRAN). where he was the head of the group diagnosis during 12 years. His major research fields include data validation and reconciliation, data-based diagnosis, model-based diagnosis, fault detection and isolation, fault tolerant control.

He has successfully advised 70 Ph.D. and published about 600 refereed technical communications (including 130 papers in international journals, 370 communications in international conferences, 100 communications in national conferences), 4 books, 15 chapters in collective volumes. A list of these publications can be founded at http://perso.ensem.inpl-nancy.fr/Jose.Ragot/. Applications have been in various fields such as mineral and metallurgical processing, chemical engineering, water treatment, aerospace, environmental processes (air pollution, water purification, water distribution), nuclear powerplants. Since 1980, José RAGOT participated or was the conductor of about 60 industrial research projects (with EDF, ELF, ARCELOR MITAL, AIRLOR, PECHINEY, ALSTOM, SATL, MINEMET, SNECMA, LAFARGE, DELPHI, RHÔNE POULENC, PSA, AIRBUS, THALES, etc.).

Pierre Beauseroy received the Engineer degree and a Master degree in 1988. He also received a Ph.D. from the Université de Technologie de Compiègne in 1992. In 2007, he obtained a HDR from the Université de Technologie de Compiègne.

Between 1993 and 2010 he was Associate Professor at the Université de Technologie de Troyes, France. He is actually a Full Professor at the same university. Since 2011, he is the head of the Laboratory of Systems Modelling and Dependability (LM2S). He is a partner of CNRS projects, ANR projects, GIS 3SGS projects and investment for the future program on smart grids. His research interests include machine learning, pattern recognition, classification, health monitoring, data analysis and nonstationary signal analysis.

André Smolarz graduated in mechanical engineering in 1978, from the Université de Technologie de Compiègne (UTC), where he also received a Ph.D. degree in Statistical Pattern Recognition in 1982. He is currently an Associate Professor at the Université de Technologie de Troyes (UTT). His research interests include statistical pattern recognition, ensemble methods, statistical learning, probabilistic modeling of joint binary decision rules.

Edith Grall-Maës is an Associate Professor at Troyes University of Technology. She received the Ingenieur degree from Compiègne University of Technology, France, in 1989, and the Doctorat Sciences Techniques from Federal Polytechnic School of Lausanne, Switzerland, in 1993. Her research interests include signal and data analysis, supervised and unsupervised learning decision methods, with applications mainly to prognostic and health management.