Multi-task Learning for One-class Classification

Haiqin Yang, and Irwin King, *Senior Member, IEEE,* and Michael R. Lyu, *Fellow, IEEE*

***Abstract*— In this paper, we address the problem of one-class classification. Taking into account the fact that in some applications, the given training samples are rather limited, we attempt to utilize the advantages of *Multi-task Learning* (MTL), where the data of related tasks may share similar structure and helpful information. We then propose an MTL framework for one-class classification. The framework derives from the one-class ν-SVM and makes use of related tasks by constraining them to have similar solutions. This formulation can be cast into a *second-order cone program*, which achieves a global solution and is solved efficiently. Further, the framework also maintains the favorable property of the ν parameter in the ν-SVM, which can control the fraction of outliers and support vectors, in one-class classification. This framework also connects with several existing models. Experimental results on both synthetic and real-world datasets demonstrate the properties and advantages of our proposed model.**

I. Introduction

Multi-task learning (MTL, also known as *inductive transfer* or *learning to learn*) has become a research topic of renewed interest in machine learning, see [2], [4], [9], [10], [11], [13], [15], [16] and references therein. One main insight of multi-task learning techniques is that related tasks share similar structure and information which may be useful for improving the performance of these tasks [3], [14], [17], [21]. This is especially beneficial when the number of samples in a specific task is limited. Incorporating other related useful information and utilizing those "background data" efficiently will actually help for the task-of-interest.

Currently, nearly all multi-task learning models focus on supervised learning tasks [2], [5], [4], [10], [11], while only few focus on semi-supervised learning task [18]. However, there is no research touching on the employment of the MTL in one-class classification problems. The problem of one-class classification can be regarded as a special type of classification problems. Usually, in solving one-class classification problems, researchers are dealing with what is really a two-class classification problem, where the two classes are called the *target* class and the *outlier* class, respectively [24], [25]. This problem is common in applications such as machine diagnostics, novelty detection, outlier detection, disease detection, etc. [22], [24], [25]. In early studies, a typical solution for this problem was to estimate the probability density of the target data, then to assign an object as an outlier when the object falls into a region with density lower than a certain threshold [6]. Later on, researchers developed one-class classification models based on the Support Vector Machine (SVM), such as Support Vector Domain Description [25] and one-class ν-SVM [24], [22]. These approaches follow Vapnik's principle, one of the key concepts in learning theory: never try to solve a problem which is more general than the one that is actually interested in [27].

Haiqin Yang, Irwin King and Michael R. Lyu are with the Department of Computer Sciences and Engineering, The Chinese University of Hong Kong, Shatin, N.T., Hong Kong (email: {hqyang, king, lyu}@cse.cuhk.edu.hk).

For real applications, a very common problem is that the labeled training samples are usually too few for a specific task. This is ubiquitous in applications such as bioinformatics, or related diagnostics tasks. There are several possible solutions. For example, one may solve this problem by restricting the function complexity using prior knowledge, or by collecting more data. However, prior knowledge may not exist or may be insufficient, while getting new data may be too expensive or there may not exist further representative samples for a solo task. However, it is often possible to exploit relevant data from other related tasks. How to use the partially representative data from relevant tasks is a key issue.

In this paper, we aim to utilize the advantages provided by the MTL and focus on the one-class classification problem. By upper-bounding the distance between solutions of related paired-tasks, we derive a ν-SVM style MTL framework for one-class classification. The proposed model can be transformed and solved by a *second-order cone program* (SOCP). Further, its corresponding kernelized version can be solved in a *matrix-fractional program* (MFP) [19], [7], which is also an SOCP. Hence, the proposed model can attain a global solution and can be solved efficiently. The MTL framework not only takes the one-class ν-SVM as one special case, maintaining the favorable property of the ν parameter, but also connects to other related models. Experimental comparisons on toy data and real-world datasets demonstrate the validity and promise of the proposed MTL for one-class classification in enhancing current existing one-class SVMs.

The paper is organized as follows: Section II reviews some current work on one-class support vector machines. Section III defines and formulates the MTL framework for one-class classification. Section IV derives its kernelized version and a corresponding solution. Section V discusses the properties of the MTL framework in one-class classification. Section VI details our experiment and the results. Finally, the paper is concluded in Section VII.

II. Related Work

In this section, we introduce current related work for one-class classification.

In one-class classification, the only given information are N samples of the same class in a data set $\{\mathbf{x}_i\} \subseteq \mathcal{X}$, with $\mathcal{X} \subseteq \mathbb{R}^d$, the data space, from a certain distribution. The task is to find a separating boundary between the data set and the rest of the feature space by utilizing the provided one kind of labeled data only.

Following Vapnik's principle mentioned in Section I, there are two kinds of SVM derivatives to solve the one-class classification problem. One idea is the *Support Vector Domain Description* (SVDD): it maps the data into a feature space and seeks a sphere with minimum volume containing all or most of the samples in the target class [25]. When a future point falls in the ball, it is deemed to be a "target" object; otherwise, it is an outlier object. Another idea is the ν-SVM: this model maps the data into a feature space and aims to separate the given data from the origin with a maximum margin. The algorithm returns a decision function f taking the value $+1$ in a "small" region capturing most of the data points in the target class, and -1 elsewhere [24]. The latter approach introduces a favorable parameter $\nu \in (0, 1]$, which can control the fraction of outliers and the fraction of support vectors [22]. This model is termed as one-class ν-SVM. The above two approaches can be transformed and represented in a kernel form and the SVDD also can be introduced by the ν parameter [22]. In the following, we only introduce the one-class ν-SVM.

The idea of a one-class ν-SVM can be solved by the following quadratic program:

$$\begin{aligned} \min_{\mathbf{w},\boldsymbol{\xi},\rho} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + \frac{1}{\nu N}\sum_{i=1}^{N}\xi_i - \rho \qquad (1) \\ \text{s.t.} \quad & \mathbf{w}^\top\phi(\mathbf{x}_i) \geq \rho - \xi_i, \quad i = 1, \ldots, N, \\ & \mathbf{w} \in \mathbb{R}^f, \quad \boldsymbol{\xi} \in \mathbb{R}^N_+, \quad \rho \in \mathbb{R} \end{aligned}$$

where $\phi\colon \mathbb{R}^d \to \mathbb{R}^f$, is a function mapping the data in the original space to a new feature space, and $\nu \in (0, 1)$ is an introduced parameter which can control the faction of outliers and the faction of support vectors.

The optimal boundary is then determined by the support vectors expansion:

$$f(\mathbf{x}) = \text{sign}\left(\sum_{i=1}^{N}\alpha_i\mathbf{K}(\mathbf{x}_i, \mathbf{x}) - \rho\right),$$

where $\boldsymbol{\alpha}$ is the solution of the dual form of the above quadratic program and training samples $\mathbf{x}_i$ with non-zero α_i are support vectors. The kernel matrix $\mathbf{K}$ is defined by the inner product of mapping features and needs to satisfy Mercer's condition [27], [23].

III. Formulation

In this section, we consider one-class classification in the MTL frmaework.

Suppose there are T tasks, all sharing the common data space $\mathcal{X}$ and there are N samples in a data set $\{\mathbf{x}_i\} \subseteq \mathcal{X}$, where each sample belongs to one and only one task. Let $\mathcal{T}_t$ be the t-th task, consisting of its related samples. Hence, $\sum_{t=1}^{T}|\mathcal{T}_t| = N$ and $\mathcal{T}_k \cap \mathcal{T}_l = \emptyset$, $\forall k \neq l$. Next, suppose there is a *task relation network* indicating the relationships among tasks. The *task relation network* can be represented by a graph, where each node denotes a task and two nodes are connected by an edge if these two tasks are related to each other. The edge set in this network can be denoted by $\mathcal{E} = \{(i_m, j_m)_{m=1}^{M}\}$.

Similar to the idea of separating target data from the origin with maximum margin in the one-class ν-SVM, we seek the decision boundary corresponding to the t-th task as

$$\begin{aligned} f_t(\mathbf{x}) \;=\; & \text{sign}\left(\mathbf{w}_t^\top\phi(\mathbf{x}) - \rho_t\right), \\ & \mathbf{w}_t \in \mathbb{R}^f, \quad \rho_t \in \mathbb{R}, \quad t = 1, \ldots, T \end{aligned}$$

by making each task separates its target data from the origin with maximum margin and setting the solutions of related tasks are close to each other.

The first objective is similar to the optimal solution in (1). The second objective can be fulfilled by upper-bounding each difference between the solutions of related task pairs by a positive scalar η as [15]:

$$\frac{1}{2}\|\mathbf{w}_{i_m} - \mathbf{w}_{j_m}\|^2 \leq \eta, \quad \forall\ (i_m, j_m) \in \mathcal{E}.$$

This constraint is described as a local constraint in [26], and it makes the structure of related tasks close to each other. Hence, by imposing this constraint, the number of target training samples will be increased implicitly compared to training a task individually and the related tasks will share common information partially.

Hence, we formulate the *multi-task learning* framework for one-class classification (MTL–OC) as follows:

$$\begin{aligned} \min_{\mathbf{w},\boldsymbol{\xi},\boldsymbol{\rho},\eta} \quad & \frac{1}{2T}\sum_{t=1}^{T}\|\mathbf{w}_t\|^2 + \frac{1}{N}\sum_{t=1}^{T}\frac{1}{\nu_t}\sum_{i\in\mathcal{T}_t}\xi_i \\ & \qquad - \sum_{t=1}^{T}\rho_t + C_\eta\eta, \qquad (2) \\ \text{s.t.} \quad & \mathbf{w}_t^\top\phi(\mathbf{x}_i) \geq \rho_t - \xi_i, \quad \forall\ i \in \mathcal{T}_t, \quad t = 1, \ldots, T, \\ & \frac{1}{2}\|\mathbf{w}_{i_m} - \mathbf{w}_{j_m}\|^2 \leq \eta, \quad \forall\ \ (i_m, j_m) \in \mathcal{E}, \\ & \mathbf{w} \in \mathbb{R}^{f\times T}, \quad \boldsymbol{\xi} \in \mathbb{R}^N_+, \quad \boldsymbol{\rho} \in \mathbb{R}^T, \quad \eta \in \mathbb{R}_+ \end{aligned}$$

where $\mathbf{w} \equiv [\mathbf{w}_1, \ldots, \mathbf{w}_T]$ and $\boldsymbol{\xi} \equiv [\xi_1, \ldots, \xi_N]^\top$. Here, we introduce parameters $\boldsymbol{\nu}_T \equiv [\nu_1, \ldots, \nu_T]$, where $\nu_t \in (0, 1]$ for the corresponding task. The advantage of ν_t is similar to that in the ν-SVM; we will analyze its properties in Section V. C_η is another positive trade-off parameter for the upper bound of related tasks. Different C_η values will affect the weights of the related tasks. More detailed properties of C_η are discussed in Section V.

IV. Kernel Version

In the following, we give the dual formulation for the problem of MTL–OC (2) in kernel form and cast it into an SOCP problem, more specially, through a *matrix-fractional program* (MFP) [19].

A. Duality

First, we define some notations for the kernels. Let $\mathbf{K}^{\text{fea}}$ be the feature kernel matrix whose (i, j)-th element is the inner product of feature vectors $\phi(\mathbf{x}_i)$ and $\phi(\mathbf{x}_j)$, i.e.,

$\mathbf{K}^{\text{fea}}_{ij} = \phi(\mathbf{x}_i)^\top \phi(\mathbf{x}_j)$. Hence, this feature kernel matrix is a positive semidefinite matrix. Through defining different feature kernels, $\mathbf{K}^{\text{fea}}$, the data domain information can be mapped correspondingly.

The second kernel matrix is the task relationship matrix with an M-dimensional non-negative parameter vector $\boldsymbol{\beta} \in \mathbb{R}^M_+$:

$$\mathbf{K}^{\text{task}}(\boldsymbol{\beta}) = \left(\frac{1}{T}\mathbf{I}_T + \mathcal{M}\boldsymbol{\beta}\right)^{-1} \tag{3}$$

where $\boldsymbol{\beta}$ are dual variables corresponding to the first constraints in (2). $\mathbf{I}_T \in \mathbb{R}^{T\times T}$ is an identity matrix, $\mathcal{M}\boldsymbol{\beta} = \sum_{m=1}^M \beta_m \mathcal{M}_m$, where $\mathcal{M}_m = \mathbf{E}_{i_m i_m} + \mathbf{E}_{j_m j_m} - \mathbf{E}_{i_m j_m} - \mathbf{E}_{j_m i_m}$, and $\mathbf{E}_{ij} \in \mathbb{R}^{T\times T}$ is a sparse matrix whose (i,j)-element is one and all the others are zero. This gives a graph Laplacian kernel, where the m-th edge is weighted by the factor β_m.

Now, let $\mathbf{Z} \in \mathbb{N}^{T\times N}$ be the indicator of a task and a sample such that $Z_{t,i} = 1$, if $i \in \mathcal{T}_t$, and $Z_{t,i} = 0$, otherwise. Then the information about the tasks is presented by an $N \times N$ matrix $\mathbf{Z}^\top \mathbf{K}^{\text{task}}(\boldsymbol{\beta})\mathbf{Z}$. These two kernel matrices are combined together as

$$\mathbf{K}^{\text{com}}(\boldsymbol{\beta}) = \mathbf{K}^{\text{fea}} \circ (\mathbf{Z}^\top \mathbf{K}^{\text{task}}(\boldsymbol{\beta})\mathbf{Z}),$$

where $\circ$ is the *Hadamard product*, or element-wise product. This parameterized matrix $\mathbf{K}^{\text{com}}(\boldsymbol{\beta})$ is guaranteed to be positive semidefinite [12].

To solve the primal problem of the MTL–OC in (2), we can use the Lagrange multipliers method and obtain its dual problem as follows:

$$\begin{aligned}
\min_{\boldsymbol{\alpha},\boldsymbol{\beta}} \quad & \frac{1}{2}\boldsymbol{\alpha}^\top \mathbf{K}^{\text{com}}(\boldsymbol{\beta})\boldsymbol{\alpha}, \\
\text{s.t.} \quad & 0 \le \alpha_i \le \frac{1}{N\,\nu_i}, \quad \text{if} \quad i \in \mathcal{T}_t, \quad \nu_i = \nu_t, \\
& \mathbf{Z}\boldsymbol{\alpha} = 1, \\
& \mathbf{1}_M^\top \boldsymbol{\beta} \le C_\eta, \\
& \boldsymbol{\alpha} \in \mathbb{R}^N_+, \quad \boldsymbol{\beta} \in \mathbb{R}^M_+,
\end{aligned} \tag{4}$$

where $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are Lagrange multipliers corresponding to the first and the second kinds of constraints in (2); and $\mathbf{1}_k \in \mathbb{R}^k$ is a k-dimensional vector with all element values equal to 1.

In the test stage, a new sample $\mathbf{x}$ in the k-th task can be determined by

$$f_k(\mathbf{x}) = \text{sign}\left(\sum_{i=1}^N \sum_{t=1}^T \alpha_i \mathbf{K}^{\text{fea}}(\mathbf{x}_i, \mathbf{x})\mathbf{K}^{\text{task}}(t,k) Z_{t,i} - \rho_k\right),$$

where $\mathbf{K}^{\text{fea}}(\cdot\,,\cdot)$ and $\mathbf{K}^{\text{task}}(\cdot\,,\cdot)$ are the kernel functions over features and tasks, respectively.

B. SOCP Transformation

In the following, we will solve the optimization in (4) using the standard procedure in [7, ch. 4].

Now, suppose the feature kernel matrix $\mathbf{K}^{\text{fea}}$ has rank r and can be decomposed as $\mathbf{K}^{\text{fea}} = \mathbf{U}^{\text{fea}}\mathbf{U}^{\text{fea}\top}$, where $\mathbf{U}^{\text{fea}} \in \mathbb{R}^{N\times r}$. Let $\mathbf{U}^{\text{fea}} \equiv [\mathbf{f}_1, \dots, \mathbf{f}_r] \in \mathbb{R}^{N\times r}$ and matrices $\mathbf{G}_\ell \equiv \mathbf{Z}\text{diag}\,(\mathbf{f}_\ell)$, for $\ell = 1, \dots, r$. Using these representations, the objective function in (4) can be rewritten as

$$J_r(\boldsymbol{\alpha},\boldsymbol{\beta}) = \frac{1}{2}\sum_{\ell=1}^r \boldsymbol{\alpha}^\top \mathbf{G}_\ell^\top \left(\frac{1}{T}\mathbf{I}_T + \mathcal{M}\boldsymbol{\beta}\right)^{-1} \mathbf{G}_\ell \boldsymbol{\alpha}$$

The above formulation is a combination of r MFPs.

Next, let $\mathbf{q}_m \in \mathbb{R}^T$ for each edge, i.e., we can denote the task relatedness: $\mathbf{q} = \mathbf{e}_{i_m} - \mathbf{e}_{j_m}$, where $\mathbf{e}_{i_m}$ is a unit vector with the i_m-th element being one. Again, let $\mathbf{Q}$ be a matrix consisting of $\mathbf{q}$ as: $\mathbf{Q} = [\mathbf{q}_1, \dots, \mathbf{q}_M] \in \mathbb{R}^{T\times M}$. Thus, the graph Lagrangian matrix of task relatedness can be expressed by $\mathcal{M}\boldsymbol{\beta} = \mathbf{Q}\text{diag}\,(\boldsymbol{\beta})\mathbf{Q}^\top$. Hence, the objective function in (4), i.e., $J_r(\boldsymbol{\alpha},\boldsymbol{\beta})$, is cast into the following MFP problem:

$$\min_{\boldsymbol{\alpha},\boldsymbol{\beta}} \ \frac{1}{2}\sum_{\ell=1}^r \boldsymbol{\alpha}^\top \mathbf{G}_\ell^\top \left(\frac{1}{T}\mathbf{I}_T + \mathbf{Q}\text{diag}\,(\boldsymbol{\beta})\mathbf{Q}^\top\right)^{-1} \mathbf{G}_\ell \boldsymbol{\alpha} \tag{5}$$

subject to the same constraints in (4).

Hence, we obtain a standard MFP form of (5). We can easily transform it into the following SOCP problem:

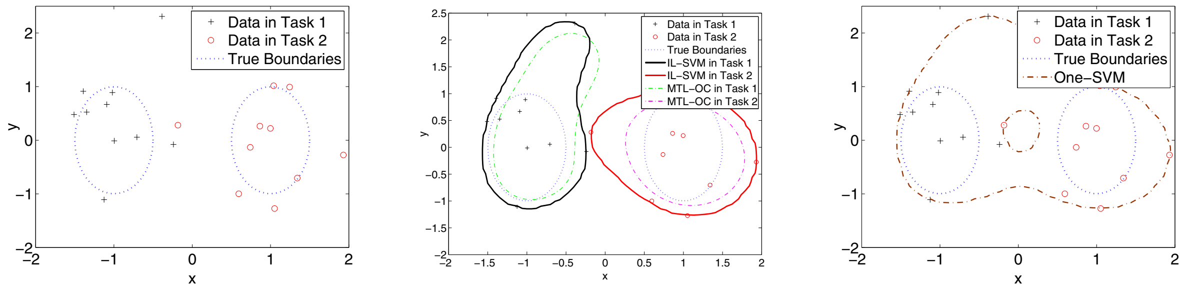
$$\begin{aligned}
\min_{\mathbf{v}_0,\,\mathbf{v},\,\boldsymbol{\alpha},\,\boldsymbol{\beta},\,\mathbf{t}_0,\,\mathbf{t}_m} \quad & \frac{1}{2}\sum_{\ell=1}^r \left(t_{0,\ell} + \sum_{m=1}^M t_{m,\ell}\right) \\
\text{s.t.} \quad & \boldsymbol{\alpha} \le \frac{1}{\boldsymbol{\nu}N}\mathbf{1}_N, \quad \mathbf{Z}\boldsymbol{\alpha} = \mathbf{1}_T, \\
& \mathbf{1}^\top\boldsymbol{\beta} \le C_\eta, \\
& \frac{1}{\sqrt{T}}\mathbf{v}_{0,\ell} + \mathbf{Q}\mathbf{v}_\ell = \mathbf{G}_\ell\boldsymbol{\alpha}, \qquad \forall \ell \\
& \left\| \begin{bmatrix} 2\,\mathbf{v}_{0,\ell} \\ t_{0,\ell} - 1 \end{bmatrix} \right\|_2 \le t_{0,\ell} + 1\,, \qquad \forall \ell \\
& \left\| \begin{bmatrix} 2\,\mathbf{v}_{m,\ell} \\ \beta_m - t_{m,\ell} \end{bmatrix} \right\|_2 \le \beta_m + t_{m,\ell}\,, \forall m, \forall \ell \\
& \mathbf{v}_0 \in \mathbb{R}^{T\times r},\ \mathbf{v} \in \mathbb{R}^{M\times r},\ \boldsymbol{\alpha} \in \mathbb{R}^N_+, \\
& \boldsymbol{\beta} \in \mathbb{R}^M_+,\ \mathbf{t}_0 \in \mathbb{R}^r,\ \mathbf{t}_m \in \mathbb{R}^r, \\
& \ell = 1, \dots, r,\ m = 1, \dots, M\,.
\end{aligned} \tag{6}$$

Hence, we transform the problem of kernelized MTL–OC into a new SOCP problem. Based on the computational complexity analysis of SOCP problems in [19], we can summarize the result as follows:

Theorem 1: The dual problem of MTL in one-class classification of (4) can be cast as an SOCP problem in (6) and be solved in $O((M\,r)^2((M+T)\,r + N))$.

Hence, the optimization of the kernelized MTL–OC can be solved by the SOCP in (6), which attains a global optimal solution. To solve an SOCP problem, one can adopt different methods, e.g., interior-point methods, barrier methods, etc. and use some standard solvers, e.g., SeDuMi, SDPT, etc. Here, we use the cvx toolbox [7] to solve our model.

Further, for our model, if the above time complexity is dominated by N, the bound of the time for our model is linear to N, which should be very efficient. However, in an actual computation, the time complexity is not dominated by N. A further study is to consider how to speed it up and extend the scalability.



(a) Data and True Boundaries

(b) IL–SVM vs. MTL–OC

(c) One–SVM

Fig. 1. Toy example with two tasks demonstrates the proposed model.

V. Discussion

In the following, we discuss the properties of our proposed MTL for one-class classification.

- **Connection to one-class ν-SVM**
  The one-class ν-SVM [22], [24] is a special case of our proposed MTL model. Actually, when the number of tasks is one, the optimization in (2) is reduced to the one-class ν-SVM.
  In addition, if we set $C_\eta = 0$, i.e., we discard the control of the closeness of the weights in related tasks, then our MTL framework for one-class classification corresponds to training each individual one-class ν-SVM.
- **Relation to MTL via Conic Programming**
  Our proposed MTL model focuses on the problem of one-class classification. When it employs the label information in a binary classification paradigm, it can be considered as a ν trick of the MTL via conic programming, which distinguishes itself from the formulation in [15].
- **Proposition of νs**
  Similar to the property in one-class ν-SVM [24], we have the following proposition:
  Proposition 2: Suppose the solution of (2) satisfies $\rho_t \neq 0$ for the t-th task. The following statements hold:
  1) ν_t is an upper bound on the fraction of outliers for the t-th task.
  2) ν_t is a lower bound on the fraction of support vectors for the t-th task.
  3) Suppose the data of the t-th task were generated independently from a distribution $P(\mathbf{x})$ without discrete components and the kernel is analytic and non-constant. With probability 1, asymptotically, ν_t equals both the fraction of support vectors and outliers for the t-th task.

  The above proposition can be proved based on the constraints of the dual problem and the fact that outliers must have Lagrange multipliers at the upper bound.
  For practical applications, we can use a global ν, where ν_t is proportional to the number of the training samples in the related t-th task, as $\nu_t = \nu\frac{|\mathcal{T}|_t}{N}$. So we can use a single global parameter ν to control the fraction of support vectors and outliers consistently.

VI. Experiments

In this section, we demonstrate the validity and advantage of the proposed method through experiments.

A. Models and Measurement

In the experiment, we compare three methods: our proposed MTL–OC model; individually learned SVM (IL–SVM) and One–SVM. For the MTL–OC model, data for all tasks and the information of their tasks relationships are fed into the model to get the boundaries for different tasks. For IL–SVM, data for each individual task are trained in a one-class ν-SVM individually. The decision boundary for each task is obtained correspondingly. For the One–SVM, all samples in the multiple tasks are considered as one big task and they are trained by the one-class ν-SVM.

For all three models, the gaussian kernel, $k(\mathbf{x}, \mathbf{y}) = \exp(-\gamma\|\mathbf{x} - \mathbf{y}\|^2)$, is used as the feature kernel. The corresponding parameters are expressed in detail in the following subsection. For real-world datasets, the values of the parameters C_η and $\boldsymbol{\nu}_T$ for the MTL–OC model and related parameters for IL–SVM and One–SVM are tuned by cross-validation over the training set.

A good one-class classifier will try to minimize two types of errors, namely the fraction of false positives (FP) and the fraction of false negatives (FN). For a classifier, by varying the threshold, these two errors can be obtained correspondingly, and a Receiver Operating Characteristics (ROC) curve [20] is then obtained. Usually, the area under the ROC curve, AUC, can be used to measure the performance of a one-class classifier [8]. The larger the AUC, the better the one-class classifier. In the experiment, the AUC of the ROC is calculated by the trapezoid area.

B. USPS Dataset

The U.S. Postal Service (USPS) database [1] is a handwritten digits database containing 9298 digit images of size $16 \times 16 = 256$ pixels, of which the last 2007 comprise the test set. Pixel values in each image are scaled to the range of -1.0 and 1.0.

Here, we create two fake but related tasks from this USPS database to mimic the application of recognizing some noisy images with the help of clear and related images. We choose digit '4' as the target object and create two additional mask patterns with random noises which are generated from uniform distribution: one is with thin noise and the other one is with thick noise. We then add these two masks to the images of digit '4' in the original training set of the USPS database. Figure 2 shows some samples of the final images on both the training and test datasets. The objective of this experiment is to show how clean data can be used for improving the performance of outlier detection on the noisy related data.

In the training procedure, data in the created fake and related tasks are fed into the corresponding models to get the decision boundaries. The parameter, γ, of the feature kernel is set to $\frac{1}{0.5 \cdot 256}$ as in [22]. To test the effect of using different numbers of training samples, we randomly select 5, 10, 20 and 40 samples from the training data for each task. In the test procedure, we only use the test set (2007 samples) of the USPS database and vary the threshold to get the corresponding ROC curve on the test set. We repeat the above procedure 20 times and average their AUCs.

Table I reports the result on this task. Since there is only one test set for the target object, we average the AUCs of the IL–SVM and the MTL–OC as the final AUCs. It is obvious that our proposed MTL–OC model shows significant improvement over the IL–SVM and the One–SVM. For the IL–SVM, its performance reduces largely when training on the samples with thick noise. Moreover, its performance reduces as the number of training samples decreases for the thick noise case. This means that the more noise samples used in the training, the worse decision boundary may distract from the true one. On the contrary, our proposed MTL–OC can overcome the problem of the IL–SVM. Comparing our MTL–OC with the IL–SVM on the thick noise case, the performance of our model improves greatly. Overall, our model achieves the best performance in terms of the average AUCs. Although the performance of our MTL–OC training on the samples with thick noise does not beat that of the One–SVM on all samples, the corresponding performance of our MTL–OC is very close to that of the One–SVM and we achieve an overall better performance. Another observation is that as the number of training samples increases, the AUC increases correspondingly for the One–SVM and our MTL–OC. In the test, when the number of training samples increases from 20 to 40, there is no significant improvement on the performance. This means that when the number of training samples achieves a certain value, it will not improve the performance for our MTL–OC.

This experiment also gives us an illumination of how to detect outliers when the given one-class samples are noisy. For that case, we may try to collect some clean data and incorporate them in the training procedure to improve the detecting performance.

C. Protein Super-Family Dataset

In the following, we test the one-class classification on a real-world protein super-family dataset [1]. Table II gives a structural view of the dataset and the task relationships performed in the experiments. We will interpret it in the following.

The data from the SCOP database is the same as that of [15]: 20 kinds of amino acids consist of 400 features. In this dataset, there are four super-classes which are termed as folds [15]: DNA/RNA binding fold, Flavodoxin fold, OB-fold and SH3 fold. Each fold is divided into several super-families [15]. The DNA/RNA binding fold contains three super-families and we denote them as d1, d2 and d3, respectively. The Flavodoxin fold contains four super-families and we denote them as f1, f2, f3, and f4, respectively. The OB-fold contains three super-families and we denote them as o1, o2 and o3, respectively. There are two super-families in the SH3 fold and are denoted as s1 and s2, respectively.

The tasks' relationships are constructed as follows: classifying a super-family is considered as one task for the one-class classification. If two one-class classification tasks are in the same fold, we set them as related tasks and connect them by an edge. For an isolated task without any edge connection, our formulation of MTL–OC will define it as an independent task and its solution is consistent with that solved by the IL–SVM. Hence, we can perform the experiments on each fold respectively. The effect of the number of training samples is also tested on this dataset. We randomly choose N samples, where N equals 5, 10, and 20, from each super-family, in training for each task. The parameter of the feature kernel, $\gamma = \frac{1}{2\sigma^2}$, where σ^2 is set to the average of the squared distances to the fifth nearest neighbors, as [15]. We then train on these samples to obtain the corresponding decision boundaries for all three models and calculate their AUCs correspondingly. The above procedure is repeated ten times and we average the results of the AUCs.

The average results are shown in Fig. 3 and Fig. 4. From the results, we clearly see that the MTL–OC outperforms the IL–SVM and One–SVM mothods. It is interesting to note that the results of One–SVM are substantially worse than those of the IL–SVM. An exception exists for the subtask of the super-family f1. We guess this may be due to the skewness of the data. It is also noted that the AUCs are very small for the Flavonoid fold in all three models. Through experimental observations, there are very high false positive errors for three models in this fold. Overall, this dataset again demonstrates the advantage of our proposed MTL–OC model.

[1] `http://www-stat.stanford.edu/~tibs/ElemStatLearn/data.html`

(a) Training Samples

(b) Test Samples

Fig. 2. Samples in the USPS dataset.

TABLE I

THE PERFORMANCE (AUC) OF EACH METHOD FOR THE USPS DATASET (%).

#	IL–SVM			One–SVM	MTL–OC		
	thin	thick	average	average	thin	thick	average
5	82.2±5.1	57.6±3.1	69.9±3.9	81.7±5.1	83.9±5.1	81.2±5.2	**82.6**±5.2
10	86.2±2.7	56.3±2.3	71.2±2.3	82.8±2.9	86.2±2.9	82.8±2.4	**84.5**±2.6
20	87.2±1.9	55.7±2.1	71.4±1.8	83.3±1.2	87.2±1.3	83.1±1.6	**85.1**±1.4
40	87.3±1.3	53.8±1.5	70.5±1.2	84.1±1.1	87.2±1.1	83.1±1.5	**85.2**±1.3

TABLE II

DESCRIPTION OF THE PROTEIN SUPER-FAMILY DATASET.

Item	Content											
Folds	DNA/RNA			Flavodoxin				OB			SH3	
Super -families	d1	d2	d3	f1	f2	f3	f4	o1	o2	o3	s1	s2
IL–SVM	—	—	—	—	—	—	—	—	—	—	—	—
One–SVM	—			—				—			—	
MTL–OC	—			—				—			—	

VII. Conclusion and Future Work

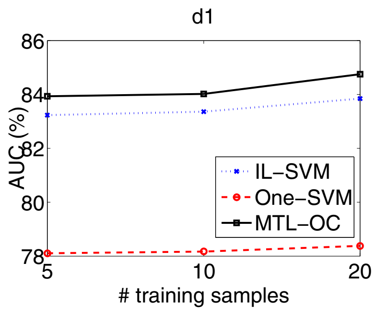
In this paper, we have proposed a new multi-task learning framework for one-class classification. The framework is to extend the one-class ν-SVM and to bound the distance between solutions of the related paired-tasks. The formulation is cast into a second-order cone program and is solved efficiently with a global optimal solution. We also demonstrated the advantage of our proposed model in the experiments on toy data, USPS digit data and a protein super-family dataset.

There are still several promising directions on the work.

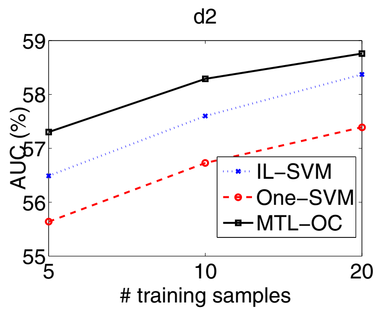
1) Our framework is derived from the ν-SVM. It is interesting to derive a similar framework from the support vector domain description.
2) Our method exploits the information from related task through model structure assumption, but there are still other methods making use of the information inherent in multi-tasks through other kinds of knowledge, e. g., common features. How to utilize other kinds of inherent knowledge in related tasks is also an interesting problem.
3) The effectiveness of our model has been demonstrated through experimental comparison. It is important and valuable to derive a framework to provide more theoretical justification of our model, e.g., analyzing the generalization error bound of the one-class classification in the MTL framework.
4) Now we have used standard toolboxes with standard methods, interior point method, to solve the SOCP problem. Standard methods contain the problem of scalability. Based on the specific form of our formulation, we believe there are still other methods to speed up the procedure of solving SOCP problem. How to speed it up and extend the scalability of our model is a promising research problem.

Acknowledgments

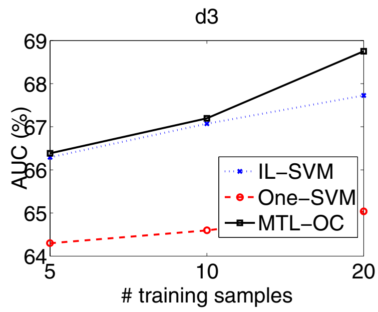
The work described in this paper was substantially supported by two grants from the Research Grants Council of the Hong Kong SAR, China (Project No. CUHK4128/08E



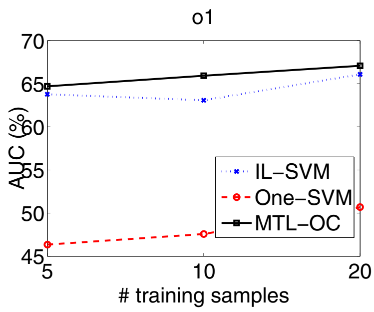
(a) d1



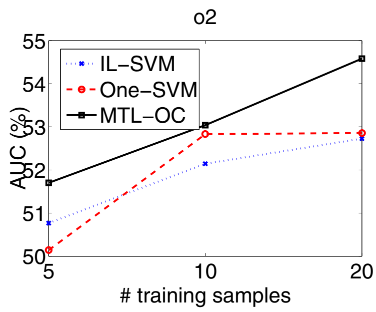
(b) d2



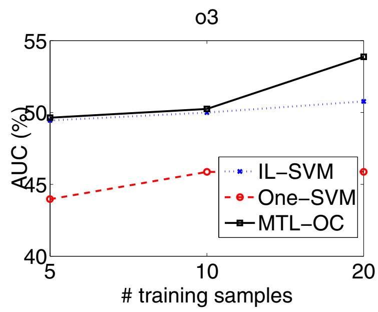
(c) d3



(d) o1

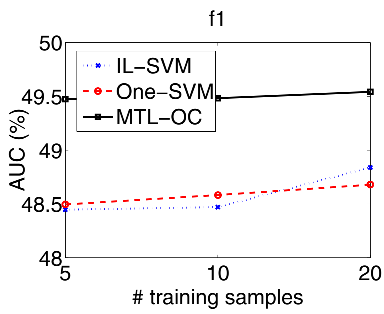


(e) o2

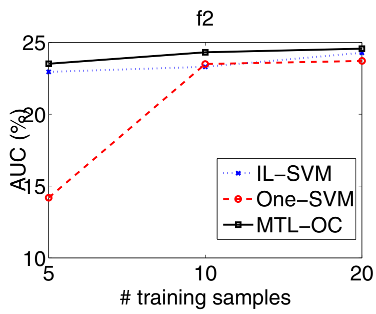


(f) o3

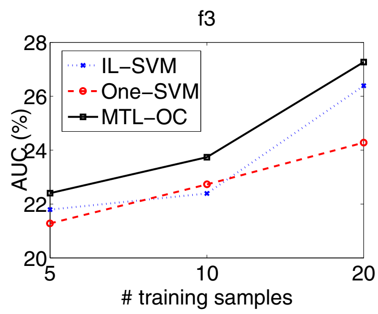
Fig. 3. The performance (AUC) of each method on the d1, d2, d3, o1, o2, o3 data of the protein super-family dataset (%).



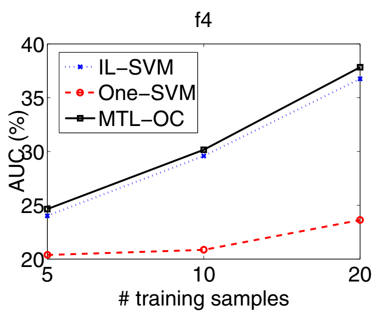
(a) f1



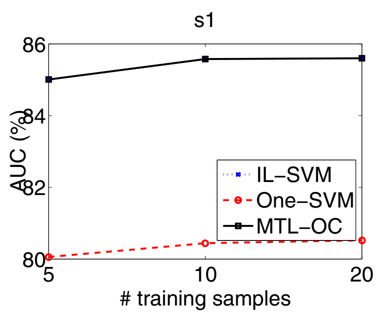
(b) f2



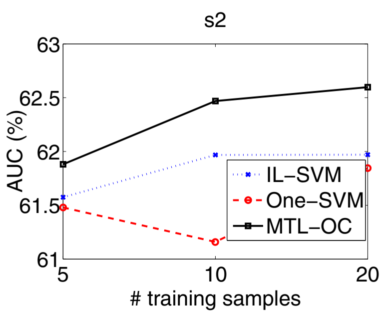
(c) f3



(d) f4



(e) s1



(f) s2

Fig. 4. The performance (AUC) of each method on the f1, f2, f3, f4, s1, s2 data of the protein super-family dataset (%).

and Project No. CUHK4154/09E).

REFERENCES

[1] A. A., H. D., B. S.E., H. T.J.P., C. C., and M. A.G. Scop database in 2004: refinements integrate structure and sequence family data. *Nucl. Acid Res.*, 32:D226–D229, 2004.

[2] A. Argyriou, T. Evgeniou, and M. Pontil. Multi-task feature learning. In B. Schölkopf, J. C. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 41–48, 2006.

[3] A. Argyriou, T. Evgeniou, and M. Pontil. Convex multi-task feature learning. *Machine Learning*, 73(3):243–272, 2008.

[4] A. Argyriou, C. A. Micchelli, M. Pontil, and Y. Ying. A spectral regularization framework for multi-task structure learning. In J. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 25–32. MIT Press, Cambridge, MA, 2008.

[5] J. Baxter. A model of inductive bias learning. *Journal of Artificial Intelligence Research*, 12:149–198, 2000.

[6] C. Bishop. Novelty detection and neural network validation. In *IEE Proceedings on Vision, Image and Signal processing, Special issue on applications of neural networks*, pages 141(4):217–222, 1994.

[7] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[8] A. Bradley. The use of the area under the ROC curve in the evaluation of machine learning algorithm. *Pattern Recognition*, 30(7):1145–1159, 1997.

[9] R. Caruana. Multitask learning. *Machine Learning*, 28(1):41–75, 1997.

[10] T. Evgeniou, C. A. Micchelli, and M. Pontil. Learning multiple tasks with kernel methods. *Journal of Machine Learning Research*, 6:615–637, 2005.

[11] T. Evgeniou and M. Pontil. Regularized multi–task learning. In W. Kim, R. Kohavi, J. Gehrke, and W. DuMouchel, editors, *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 109–117, 2004.

[12] D. Haussler. Convolution kernels on discrete structures. Technical report, UC Santa Cruz, 1999.

[13] L. Jacob, F. Bach, and J.-P. Vert. Clustered multi-task learning: A convex formulation. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21*, pages 745–752, 2008.

[14] T. Jebara. Multi-task feature and kernel selection for svms. In L. D. Raedt and S. Wrobel, editors, *Proceedings of the Twenty-first International Conference*, 2004.

[15] T. Kato, H. Kashima, M. Sugiyama, and K. Asai. Multi-task learning via conic programming. In J. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 737–744. MIT Press, Cambridge, MA, 2008.

[16] L. Liang and V. Cherkassky. Connection between svm+ and multi-task learning. In *IJCNN*, pages 2048–2054, 2008.

[17] J. Liu, S. Ji, and J. Ye. Multi-task feature learning via efficient $l_{2,1}$ norm minimization. In *UAI*, 2009.

[18] Q. Liu, X. Liao, and L. Carin. Semi-supervised multitask learning. In J. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 937–944. MIT Press, Cambridge, MA, 2008.

[19] M. Lobo, L. Vandenberghe, S. Boyd, and H. Lebret. Applications of second-order cone programming. *Linear Algebra and its Applications*, 284:193–228, 1998.

[20] M. A. Maloof. On machine learning, ROC analysis, and statistical tests of significance. In *Proceedings of the Sixteenth International Conference on Pattern Recognition (ICPR-2002)*, pages 204–207, Los Alamitos, CA, 2002. IEEE Press.

[21] G. Obozinski, B. Taskar, and M. I. Jordan. Joint covariate selection and joint subspace selection for multiple classification problems. *Statistics and Computing*, 2009.

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

[23] B. Schölkopf and A. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.

[24] B. Schölkopf, R. C. Williamson, A. J. Smola, J. Shawe-Taylor, and J. C. Platt. Support vector method for novelty detection. In *NIPS*, pages 582–588, 1999.

[25] D. M. J. Tax and R. P. W. Duin. Support vector domain description. *Pattern Recognition Letters*, 20(11-13):1191–1199, 1999.

[26] K. Tsuda and W. S. Noble. Learning kernels from biological networks by maximizing entropy. In *ISMB/ECCB (Supplement of Bioinformatics)*, pages 326–333, 2004.

[27] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer, New York, 2nd edition, 1999.