Assisting Main Task Learning by Heterogeneous Auxiliary Tasks with Applications to Skin Cancer Screening

[†]Ning Situ [‡]Xiaojing Yuan [†‡⋆]George Zouridakis
Departments of [†]Computer Science, [‡]Engineering Technology, and [⋆]Electrical and Computer Engineering
University of Houston, Houston, TX 77004, USA

Abstract

In typical classification problems, high level concept features provided by a domain expert are usually available during classifier training but not during its deployment. We address this problem from a multitask learning (MTL) perspective by treating these features as auxiliary learning tasks. Previous efforts in MTL have mostly assumed that all tasks have the same input space. However, auxiliary tasks can have different input spaces, since their learning targets are different. Thus, to handle cases with heterogeneous input, in this paper we present a newly developed model using heterogeneous auxiliary tasks to help main task learning. First, we formulate a convex optimization problem for the proposed model, and then, we analyze its hypothesis class and derive true risk bounds. Finally, we compare the proposed model with other relevant methods when applied to the problem of skin cancer screening and public datasets. Our results show that the performance of the proposed method is highly competitive compared to other relevant methods.

1 Introduction

Nonoperational are those features that are available during classifier development but not available after classifier deployment, because individual object annotation is expensive or impossible, due to time constraints or lack of expertise (Caruana, 1997). This situation arises in many biomedical applications where automated computer-aided image analysis is needed. Our application involves skin cancer screening whereby the main task is to make a decision on whether a given skin lesion is cancerous or not based on dermoscopy images. Dermatology experts rely on a set of high level concepts to characterize a lesion as malignant (Johr, 2002). For instance, the image in Fig. 1 exhibits *several* anatomical features highly suggestive of skin cancer, and for this reason, this lesion, which is indeed a melanoma, in terms of dermoscopic criteria is termed as having *multicomponent* global features. The local dermoscopic features observed in the image, namely the *irregular dot/globular structure* and the *blue-whithish veil*, are both hallmarks of melanoma. However, for a new test image, a dermatologist may not be available to identify these high-level anatomical features, and this poses a major limitation in developing accurate automated lesion-screening tools (Federman et al., 2002).

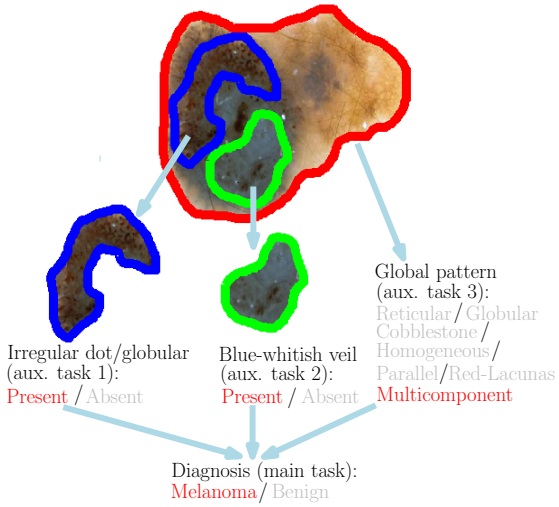


Figure 1: An example of the main task and auxiliary tasks in skin cancer detection based on an Epiluminescence microscopy (ELM) image.

In order to use nonoperational features, Caruana (1997) proposed to view nonoperational features as

Appearing in Proceedings of the 14^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2011, Fort Lauderdale, FL, USA. Volume 15 of JMLR: W&CP 15.

auxiliary tasks during the training phase and demonstrated that simultaneous learning of main task and auxiliary tasks can lead to a more accurate model. Many previous approaches on multi-task learning (Argyriou et al., 2008; Zhang et al., 2008) assumed that these tasks are homogeneous, i.e. they all share the same input space. This is not true in the skin cancer screening paradigm, as recognition of different anatomical features typically needs different low level features, such as geometry (Zouridakis et al., 2004), texture (Yuan et al., 2006), and color (Stanley et al., 2007). One obvious solution here is to combine all the heterogenous features into one large feature vector. However, this method does not use the knowledge of the split feature set and it may result in two potential problems. First, it increases the dimension of the feature vector for all tasks and probably combines some unrelated features for some tasks. Second, because multi-task learning methods typically assume that classifiers from different tasks are similar, it is likely that for some tasks, certain unrelated features are forced to impact the final output, especially when individual task classifier is assumed to be similar to their average classifier, as in the case in the models of Evgeniou et al. (2006), Daumé (2007), and Finkel and Manning (2009). Another solution that relies on Lemma 2 proposed by Evgeniou et al. (2006), is to map each of the different input spaces into one common space. This possible solution has only been described briefly by Evgeniou et al. (2006). In general, it is difficult and not straightforward to choose the common space or the mappings between spaces with difference dimensions.

To handle the heterogenous input condition, the proposed method considers *task relatedness* in the following way: we directly model the main task classifier as a weighted average of the output provided by the auxiliary task classifiers. It is important to note that we use the weighted average of the output of the classifiers, not the average of the classifier's parameters, such as, for instance, the weight of each feature in the linear case (Evgeniou and Pontil, 2004; Evgeniou et al., 2006; Daumé, 2007; Finkel and Manning, 2009). This simple idea allows us to perform domain adaptation easily regardless of the different forms of the auxiliary classifiers. The proposed method for modeling task relatedness is closely related to the ensemble learning scenario. Three representative methods using a similar ensemble idea are the adaptive support vector machine (A-SVM) (Yang et al., 2007), the linear programming boosting (LPB) (Demiriz et al., 2002; Gehler and Nowozin, 2009), and the gating network approach (Bonilla et al., 2007). These ensemble learning methods typically treat the training of each sub-model and their combination as two separate stages. Our proposed method, however, inspired from the multiple kernel learning (MKL) (Lanckriet et al., 2004) and the multi-task kernel approaches (Evgeniou et al., 2006), can learn the models for auxiliary tasks and the combined model for the main task together. Our method also increases the dimension of the feature space for the main task. However, we use a weighted average so that the impact of unrelated tasks can be decreased at the task level. This has an effect similar to the MKL, group lasso (Yuan and Lin, 2006), and their p-norm variants (i.e., non-sparse solution will be obtained when $p > 1$, in contrast to the original MKL and group lasso.) (Kloft et al.,, 2009).

To allow the auxiliary classifiers to adapt to the main task, we add a small term to each auxiliary classifier in the weighted sum, as that in A-SVM, and to build a general model, we allow the main task to have its own features that are different from those used to characterize auxiliary tasks.

Using the notion of task relatedness described above, and following the generic empirical risk minimization approach, we formulate a mathematical programming problem with a regularization term similar to that of the multiple kernel learning considered by Zien and Ong (2007) and its p-norm extensions (Kloft et al., 2009).

The contributions of this paper are: (a) Development of a convex optimization problem for main task learning with heterogeneous auxiliary tasks (section 2.3); (b) Derivation of error bounds for the proposed model (section 3). The obtained error bounds are informative. (c) A practical application of our model to skin cancer screening (section 4); (d) A comparison with three relevant methods, namely MKL, multi-task kernel (Evgeniou et al., 2006), and LPB (section 4).

2 The Proposed Learning Model

2.1 Symbols and Notations

The most important symbols and notations are listed in Table 1, while other symbols are introduced in the text. As is customary, bold-faced letters denote vectors, e.g., $\boldsymbol{d} = (d_1, \ldots, d_m)$, where the dimension m of the vector is determined in the specific context. We use $\boldsymbol{e}$ to represent a column vector of ones and $\boldsymbol{I}$ to denote the identity matrix. Generally, for $x \in \mathcal{X}^\dagger$, where $\mathcal{X}^\dagger$ is a reproducing kernel Hilbert space (RKHS), we only consider a classifier $f(x)$ of the form $f(x) = w^T x$, with $w \in \mathcal{X}^\dagger$. We consider the bias parameter by mapping x into $(x, 1)$. When a classifier has the form $\left((d_l w_l)_{(l=1)}^{(m)}\right)$, with $d_l \in \mathbb{R}$, $w_l \in \mathcal{X}_l$, and each $\mathcal{X}_l$ is an RKHS, its decision on an instance $\left((x^l)_{(l=1)}^{(m)}\right)$ can

be explicitly written as $\sum_{l=1}^{m} d_l w_l^T x^l$. We use $\|\cdot\|_p$ to denote the p-norm in an RKHS and $\|\cdot\|$ refers to the 2-norm by default.

Table 1: Symbols and Notations

Symbols	Meanings
$\mathcal{X}_l$	Feature space of task $l, 1 \le l \le m$. $\mathcal{X}_{m+1}$ denotes feature space used by the main task only. Each $\mathcal{X}_l$ is an RKHS.
$\mathcal{X}$	Product space $\mathcal{X}_1 \times \mathcal{X}_2 \times \ldots \times \mathcal{X}_{m+1}$.
x_i^l	Feature of the *i-th* instance of task l, with $1 \le l \le m$. When $l = m+1$, it denotes feature of instance i used by the main task only. For all $1 \le l \le m+1$, $x_i^l \in \mathcal{X}_l$
K_l	$n \times n$ symmetric positive definite (s.p.d.) kernel matrix where $K_l(i,j) = \langle x_i^l, x_j^l \rangle$
y_i^l	$y_i^l \in \{-1, 1\}$, label of the *i-th* instance from task l.
$\left(x_i^l\right)_{(l,i)=(1,1)}^{(m,n)}$	Vector $\left(x_1^1, x_2^1, \ldots, x_{n-1}^m, x_n^m\right)$.

2.2 Problem Statement and Task Relatedness

Assuming that there are m auxiliary tasks and only one main task (the $(m+1)$-*th* task), our goal is to choose a decision function h from a given hypothesis class $\mathcal{H} \subseteq \{-1,1\}^{\mathcal{X}}$, such that h achieves the *best true error rate* on the *main* task. We assume that all tasks are binary classification problems and the training data are n independent and identically distributed (i.i.d.) data instances $X_i = \left(\left(x_i^l\right)_{l=1}^{m+1}, \left(y_i^l\right)_{l=1}^{m+1}\right)$, $1 \le i \le n$, drawn from an unknown distribution $\mathcal{P}$ defined on the domain $\mathcal{X} \times \{-1,1\}^{m+1}$. It is important to notice that the independence assumption here is for two different i's. Inside the *i-th* training instance X_i, the x_i^l's and y_i^l's, $1 \le l \le m+1$, may not be independent of each other. This is slightly different from the setting studied by Evgeniou and Pontil (2004), Evgeniou et al. (2006), Daumé (2007), and Finkel and Manning (2009) where each of the $\left(x_i^l, y_i^l\right)$'s is an i.i.d. training data, for any i and any l. A testing instance has the form $\left(\left(x^l\right)_{l=1}^{m+1}\right)$ and our goal is to predict y^{m+1}. In the testing instance, $\left(y^l\right)_{l=1}^{m}$ are unknown.

To explicitly represent the task relatedness between main and auxiliary tasks, let the decision function for auxiliary task l be $w_l \in \mathcal{X}_l$. Assuming that the classifier for the main task can be written as $\left((d_l(w_l + v_l))_{l=1}^{m+1}\right)$, with $w_l \in \mathcal{X}_l$, $v_l \in \mathcal{X}_l$, $\|v_{m+1}\| = 0$, $d_l \ge 0$ for $1 \le l \le m+1$, and $\|\boldsymbol{d}\|_p^p \le 1$ for some $p \ge 1$, where $\boldsymbol{d}$ denotes the vector $(d_1, d_2, \ldots, d_{m+1})$, the *relatedness* between the main task and a specific auxiliary task l is captured by $\|v_l\|$, $1 \le l \le m$. Obviously, the smaller the $\|v_l\|$, the more related the main and the auxiliary tasks are. In the special case when $\|v_l\| = 0$, $1 \le l \le m+1$, and $\|w_{m+1}\| = 0$, the prediction value of the main task classifier for an instance $\left(x^l\right)_{l=1}^{m+1}$ becomes $\sum_{l=1}^{m} d_l w_l^T x^l$, which is just a weighted sum of output provided by auxiliary task classifiers.

2.3 Mathematical Programming

We formulate a mathematical programming for the problem of main task learning using heterogeneous auxiliary tasks as follows:

$$
\begin{aligned}
\mathbf{P1} \quad & \min_{\boldsymbol{w},\boldsymbol{v},\boldsymbol{\xi},\boldsymbol{d}} \frac{1}{2}\sum_{l=1}^{m} d_l \|v_l\|^2 + \frac{1}{2}\sum_{l=1}^{m} C_l d_l \|w_l\|^2 \\
& + \frac{1}{2} d_{m+1}\|w_{m+1}\|^2 + \frac{1}{2}\sum_{l=1}^{m}\sum_{i=1}^{n} d_l (\xi_i^l)^2 + C\sum_{i=1}^{n} \xi_i^{m+1} \\
\mathbf{s.t.} \quad & y_i^l (w_l^T \phi_l(x_i^l)) \ge 1 - \xi_i^l,\ 1 \le l \le m,\ 1 \le i \le n; \\
& \boldsymbol{d} \ge \mathbf{0}, \quad \|\boldsymbol{d}\|_p^p \le 1, \quad \boldsymbol{\xi} \ge \mathbf{0}, \\
& y_i^{m+1}\Big(\sum_{l=1}^{m+1} d_l (w_l + v_l)^T \phi_l(x_i^l)\Big) \ge 1 - \xi_i^{m+1}, \\
& \text{when } 1 \le i \le n
\end{aligned} \tag{1}
$$

where $\mathbf{0}$ is a column vector of zeros and $\|v_{m+1}\| = 0$. Inequalities between two vectors are taken element-wise, and $C_l, 1 \le l \le m$ and C are positive user defined parameters.

To convert **P1** into a convex optimization problem, we can simply replace w_l , v_l $(1 \le l \le m+1)$, and ξ_i^l $(1 \le i \le n$ and $1 \le l \le m)$ with $\hat{w}_l/d_l$, $\hat{v}_l/d_l$, and $\hat{\xi}_i^l/d_l$, respectively. If $d_l = 0$, we define $a/d_l = \infty$ when $a \ne 0$, and $a/d_l = 0$ when $a = 0$. Then, we can use the cutting plane algorithm to solve the convex optimization problem. We omit the detailed steps here because this is a standard procedure employed in many MKL optimization algorithms[1] (Sonnenburg et al., 2006; Kloft et al., 2009).

One possible concern is that during the optimization if $d_l = 0$, the corresponding auxiliary task's error can be encouraged to be away from 0. To avoid this situation, we can simply replace the constraint $\boldsymbol{d} \ge \mathbf{0}$ with $\boldsymbol{d} \ge \boldsymbol{\epsilon_d}$, where ϵ_d is a small positive parameter. When $\epsilon_d > 0$, the optimization method is still the same as the case of $\epsilon_d = 0$. Our experiments, as detailed in section 4, indicate that setting $\epsilon_d = 10^{-4}$ works well when $p = 2$.

In **P1** we use the hinge loss for main task error (i.e., ξ_i^{m+1}) as that in the classic SVM formulation. For the error term of auxiliary task (i.e., ξ_i^l, $1 \le l \le m$), our

[1]We provide the convex formulation and the semi-infinite programming in appendix. For completeness, we provide the full details in our supplementary materials

analysis in section 3 shows that a risk bound on main task can be obtained when quadratic loss is used for auxiliary tasks. The truncated quadratic loss is used in **P1** because we empirically observe that it performs better than the quadratic loss. For optimization, the procedure is similar with respect to these three types of loss functions (hinge loss, quadratic loss, and truncated quadratic loss).

3 Formal Analysis

We provide a risk bound analysis of the proposed learning model. We use the quadratic loss to replace the truncated quadratic loss on auxiliary tasks in **P1** (i.e., $\xi_i^l \geq 0$ is removed when $1 \leq l \leq m$) for the purpose of error bound analysis here. We first analyze the hypothesis class of our model for main task learning. Given a training set with n i.i.d. data, $X = \left\{ \left(\left(x_i^l\right)_{l=1}^{m+1}, \left(y_i^l\right)_{l=1}^{m+1} \right), 1 \leq i \leq n \right\}$, a $B > 0$, and user defined positive parameters C_l, with $1 \leq l \leq m$, the hypothesis class $\mathcal{H}_1$ for the proposed learning model is defined as

$$\mathcal{H}_1(B, X, \boldsymbol{C_l}) := \left\{ \left(x^l\right)_{l=1}^{m+1} \rightarrow \sum_{l=1}^{m+1} \left(d_l(w_l + v_l)\right)^T x^l \right|$$
$$w_l, v_l \in \mathcal{X}_l, 1 \leq l \leq m+1;\ v_{m+1} = \mathbf{0};$$
$$(2.a)\ \|\boldsymbol{d}\|_p^p \leq 1,\ \boldsymbol{d} \geq 0;\ \text{and}$$
$$\underbrace{\frac{1}{2}\sum_{l=1}^{m} d_l \|v_l\|^2}_{(2.b)} + \underbrace{\frac{1}{2}\sum_{l=1}^{m} C_l d_l \|w_l\|^2 + \frac{1}{2} d_{m+1}\|w_{m+1}\|^2}_{(2.c)} +$$
$$\underbrace{\frac{1}{2}\sum_{l=1}^{m}\sum_{i=1}^{n} d_l (w_l^T x_i^l - y_i^l)^2}_{(2.d)} \leq B \Bigg\} \qquad (2)$$

where the main task classifier is $\left(\left(d_l(w_l + v_l)\right)_{l=1}^{m+1}\right)$, and w_l is the l-th auxiliary task classifier with $1 \leq l \leq m$. Here $\mathbf{0}$ denotes the zero element in the corresponding RKHS. It should be noted that the loss of auxiliary task (i.e., (2.d)) is viewed as one of the regularization terms. In the above definition, the term (2.a) restricts the p-norm of the weight vector to be smaller than 1, similar to that of the non-sparse MKL (Kloft et al., 2009). Furthermore, the term (2.b) regularizes the task relatedness (as described in section 2.2), while the term (2.c) regularizes the norm of classifiers of different tasks. We use a weighted sum formulation similar to the regularization term of the intuitive multiple kernel learning (Zien and Ong, 2007), which was shown to be the same as the regularization term considered by Sonnenburg et al. (2006) in the case of MKL. The relation between using this kind of regularization term (i.e., the terms (2.b) and (2.c)) and using the group lasso regularization (Yuan and Lin, 2006) for MKL is discussed by Bach (2008). The term (2.d) stipulates the performance of the auxiliary task classifiers by a total quadratic error. The quadratic error here is in a weighted form. If an auxiliary task has a higher impact on the main task (i.e., d_l is high), its error will be penalized more. If $d_l = 0$, there will be no restriction on the error of the *l-th* auxiliary task and this is because w_l (the *l-th* auxiliary task classifier) has no effect on the main task classifier $\left(\left(d_l(w_l + v_l)\right)_{l=1}^{m+1}\right)$ as long as $d_l = 0$ (recall that our goal is to preform classification on the main task). Another motivation to use a weighted error term (2.d) is that it can easily lead to a convex optimization problem as described in section 2.3. The parameters C_l's provide a function similar to the regularization parameter in an SVM. When taking out all the terms related to w_l in (2.c) and (2.d) for a fixed l, we obtain $d_l\left(\frac{1}{2}C_l\|w_l\|^2 + \frac{1}{2}\sum_{i=1}^{n}(w_l^T x_i^l - y_i^l)^2\right)$, which is exactly an SVM with the quadratic loss for the *l-th* auxiliary task.

Assuming that $\|x_i^l\|^2 = 1$, with $1 \leq l \leq m+1, 1 \leq i \leq n$, for a fixed $l, 1 \leq l \leq m+1$, the $n \times n$ kernel matrix K_l with the (i,j)-*th* element $(1 \leq i, j \leq n)$ equaling $\langle x_i^l, x_j^l \rangle$, has a trace of n. We also need to assume that each of the $\mathcal{X}_l$ is an Euclidean space[2]. The dimension of $\mathcal{X}_l$ can be different for different l.

We provide true risk bounds for functions from $\mathcal{H}_1$ on the main task. We need some notations first. Let $\lambda_l, 1 \leq l \leq m+1$, be the maximum eigenvalue of K_l and $\mathcal{C}(\mathcal{K}^+) := \min\left(m+1, \|((\lambda_l)_{l=1}^{m+1}\|_q\right)$, when $p > 1$, $1/p + 1/q = 1$. When $p = 1$ we define $q = \infty$. We let $C_{min} := \min\{C_1, C_2, \ldots, C_m\}$, $E_1 := B/(2C_{min}) + m\left(B + m^{1/(2q)} n\sqrt{2B/C_{min}}\right)/(2n)$, and for any $\overline{E}, c > 0, E'(\overline{E}, c) := \overline{E} + \left(4\sqrt{4m\overline{E}/n} + 6\sqrt{\ln(c/\delta)/2n}\right)\left(\overline{E}/2 + \overline{E}m/2\right)$. Before stating the bound, we need the concept of *landmark set* introduced by Shivaswamy and Jebara (2010). A landmark set $U = \left\{ \left(\left(u_i^l\right)_{l=1}^{m+1}, \left(\bar{y}_i^l\right)_{l=1}^{m+1} \right), 1 \leq i \leq n \right\}$ of size n is a set of n data drawn i.i.d. from the same distribution as that of the training set X.

Theorem 1 *Fix $\gamma > 0$, and $C_l > 0$, $1 \leq l \leq m$. Let X be a training set of n i.i.d. data drawn from a distribution $\mathcal{P}$, and U be a landmark set of size n, for any $h \in \mathcal{H}_1(B, X, \boldsymbol{C_l})$ with $B > 0$:*

(i) With probability at least $1 - \delta$ over a random draw

[2]For the RBF (radial basis function) kernel, its feature space has an infinite dimension. Using the Taylor expansion, we can approximate it with a polynomial by truncating the higher order terms.

of X, we have

$$\Pr_{\mathcal{P}}\Big[y^{m+1} \neq \text{sign}\Big(h\big((x^l)_{l=1}^{m+1}\big)\Big)\Big] \leq \frac{1}{n\gamma}\sum_{i=1}^{n}\xi_i^{m+1} + 3\sqrt{\frac{\ln(8/\delta)}{2n}} + \frac{8m^{\frac{3}{2}}\sqrt{\ln(4/\delta)E'(E_1,8)}}{\gamma n} + \frac{2\sqrt{2B\mathcal{C}(\mathcal{K}^+)}}{\gamma\sqrt{n}} + \frac{4\sqrt{2E'(E_1,8)}}{\gamma n}\boldsymbol{E}_U[T(U,X)]$$

where

$$T(U,X) := \Bigg[\sum_{i=1}^{n}\big((x_i^l)_{l=1}^m\big)^T \Big(\frac{1}{2}\boldsymbol{I} + \frac{1}{2n}\sum_{j=1}^{n}\big((u_j^l)_{l=1}^m\big)\big((u_j^l)_{l=1}^m\big)^T\Big)^{-1}\big((x_i^l)_{l=1}^m\big)\Bigg]^{\frac{1}{2}}$$

(ii) With probability at least $1-\delta$ over a random draw of X, we have

$$\Pr_{\mathcal{P}}\Big[y^{m+1} \neq \text{sign}\Big(h\big((x^l)_{l=1}^{m+1}\big)\Big)\Big] \leq \frac{1}{n\gamma}\sum_{i=1}^{n}\xi_i^{m+1} + 3\sqrt{\frac{\ln(8/\delta)}{2n}} + \frac{8m^{\frac{3}{2}}\sqrt{\ln(4/\delta)E'(E_1,8)}}{\gamma n} + \frac{\sqrt{8B}}{\gamma\sqrt{n}}\times\sqrt{(m+1)} + \frac{4\sqrt{mE'(E_1,8)}}{\gamma\sqrt{n}}$$

where $\xi_i^{m+1} = \max\Big(0, \gamma - y_i^{m+1}h\big((x_i^l)_{l=1}^{m+1}\big)\Big)$ are the so-called slack variables.

A proof with complete details is provided in our supplementary materials. We follow the path using empirical Rademacher complexity to derive error bounds. One difficulty is that $\mathcal{H}_1$ is dependent on the training data, and hence it is hard to bound $\mathcal{H}_1$'s true risk with its empirical Rademacher complexity directly (Shivaswamy and Jebara, 2010). Our proof follows the approach developed by Shivaswamy and Jebara (2010) which used the landmark set to overcome the problem of data-dependent hypothesis class. The obtained bound in part (i) of Theorem 1 is dependent on the training data while that in part (ii) is independent of the training data but could be looser than the former. The asymptotic behavior of the bound in (ii) with respect to the number of training data n is: *main task's empirical margin error* (i.e., ξ_i^{m+1}) + $\mathcal{O}(1/\sqrt{n})$.

4 Empirical Results

4.1 Skin Cancer Screening

We first demonstrate an application of the proposed algorithm to skin cancer screening based on Epiluminescence microscopy (ELM) images. Our dataset is collected from `Interactive CD of Dermoscopy` (Argenziano et al., 2000). We have 360 skin lesion images, in which 270 are benign and 90 are melanoma. The typical resolution of the images is 500×740. We use manual segmentation (e.g., the red boundary in Fig. 1) to exclude healthy skin, and this ensures that comparison of classification performance between algorithms is not affected by incorrect automated detection of lesions boundaries. Automated lesion segmentation (Zouridakis et al., 2004) can be viewed as an orthogonal research area to the study here.

4.1.1 Main Task

The main task in skin lesion screening is to detect melanoma. We use the well known bag-of-features scheme which is widely used in computer vision to build a feature vector for each lesion. We first sample randomly $10,000$ 16×16 patches from each lesion, and then we compute Haar wavelet coefficients and color moments on each patch, and build histograms for wavelet and color moments respectively. We use a codebook size of 100 for both the wavelet and color moment features. Hence, the length of the main task feature vector (i.e., the input space before mapping into $\mathcal{X}_{m+1}$ in the analysis above) is 200.

4.1.2 Auxiliary tasks

Auxiliary tasks include the global dermoscopic feature and three local dermoscopic features. The **global dermoscopic feature** has 7 classes as listed in Fig. 1, but we only classify each lesion as *multicomponent* or *not multicomponent*, since being *multicomponent* is a sign of melanoma. We use the local binary patterns $LBP^{riu}_{16,2}$ and $LBP^{riu}_{24,3}$ proposed by Ojala et al. (2002) as low level features for the global dermoscopic feature and the standard deviation and entropy of the histograms built from wavelet and color moments, respectively. Feature vector size here is 48. The **three local dermoscopic features** include *irregular dot/globular*, *irregular network*, and *blue-whitish veil*. Because of the *weak label* of the database (Argenziano et al., 2000), we know whether certain local pattern exists in the whole lesion, but we do not know where exactly in the lesion it is present. To solve this problem, we first perform a segmentation inside the lesion by graph cut (5 segments for each lesion). If any of the segmented region contains a certain local feature, that local feature is considered as present in the lesion. This is a typical multi-instance learning (MIL) problem that we convert into a single instance learning problem by the method proposed by Li and Yeung (2009) which can then be intergraded into our framework. This MIL method identifies instance prototypes (IPs) (Maron and Lozano-Pérez, 1998) for each task based on low level features

of each segmented region (instance) which are the same as those of the main task. Different tasks will have distinct IPs that lead to distinct feature spaces. The final sizes of the feature vectors for the three local dermoscopic patterns are 75, 69, and 130 respectively.

4.1.3 Experimental Settings

We use five-fold cross-validations (CV) on all methods applied to the dataset. We are only concerned with performance on the main task, which is the goal of skin cancer screening. Our dataset is imbalanced and the accuracy value is not a good measure of classifiers. Hence, we use the area under the receiver operating characteristic curve (AUC) of the main task as performance measure. In this study we compare the following methods.

`Simple`: Concatenation of all features from all tasks and training with the main task label using an SVM.

`MTK`: Multi-task kernel. The general definition of MTK was proposed by Evgeniou et al. (2006) and our implementation uses a specific type of MTK defined in Eq. (22) of their paper (RBF kernel is used to replace the dot product for non-linear mapping). Their experiment has demonstrated the effectiveness of this kind of MTK. All MTK models (Evgeniou and Pontil, 2004; Evgeniou et al., 2006; Finkel and Manning, 2009; Daumé, 2007) focused on considering homogenous feature spaces. To adapt MTK to our problem, we concatenate all heterogenous features here as that in `Simple`.

`Single` (Baseline): A single kernel built from the feature of the main task (i.e., K_{m+1}) and training with the main task label using an SVM.

`MKL`: Multiple kernel learning, which was used to combine heterogeneous data sources (Lanckriet et al., 2004). We use the non-sparse multiple kernel learning (Kloft et al., 2009) and the 2-norm in the constraint for the weight variables (i.e., $\|\boldsymbol{d}\|_2^2 \leq 1$). MKL does not use labels of auxiliary tasks but uses the features of auxiliary tasks.

`LPB`: Linear programming boosting (Demiriz et al., 2002) was not designed in a multi-task setting originally. However, it is straightforward to generalize LPB to our problem. We use the ν-LPB formulation[3] from Gehler and Nowozin (2009), which was shown to be more effective than MKL in combining heterogeneous features (Gehler and Nowozin, 2009). In our experiments, ν-LPB first trains $m+1$ SVMs for m auxiliary tasks and the main task with their own features. Then, ν-LPB builds a linear weighted combination of the $m+1$ classifiers for the main task. This method is the closest to the proposed learning framework. The major difference is that LPB performs learning in two steps, namely learns each task first, and then based on the output of the first step, learns the main task. In our model, learning is performed coherently, namely all auxiliary tasks and the main task are learned simultaneously.

`CMHA`: our model which concurrently learns main and heterogeneous auxiliary tasks (CMHA) and $p = 2$.

In order to see the effect of auxiliary task labels (i.e., nonoperational features), we consider replacing all the auxiliary task labels in `CMHA` and `LPB` with the main task label of the lesion, and still use the features of auxiliary tasks. For LPB, this is just its original formulation (Gehler and Nowozin, 2009) to combine heterogenous features. We call these two methods without auxiliary task labels as `CMHA-WOA` and `LPB-WOA` respectively.

We also test `MKL` and `CMHA` by setting their weight variables (i.e., $\boldsymbol{d}$) to be uniform, which are denoted as `MKL-ave` and `CMHA-ave` respectively.

We use the RBF kernel with a parameter[4] of 1 (For `Simple` and `MTK` we also try values of 1/5 and 1/522 for they concatenate five sets of features and the total feature length is 522. Best results are reported and this could benefit them in comparison). We select the regularization parameter for `Simple`, `Single` (Baseline), `MKL`, and `MKL-ave` from the set $\{1000, 100, 50, 25, 10, 0.1\}$ based on their best prediction performance which could benefit these methods in comparison. The '(λ, γ)' pair of `MTK` (Evgeniou et al., 2006) is searched in $\{(0.1, 0.2, \ldots, 0.9) \times (1000, 100, 50, 25, 10, 0.1)\}$ and the result is also reported for its best setting from the final performance. For all regularization parameters of methods from our model (**P1**) and LPB, including 'C', 'C_l', and the 'ν' in ν-LPB (Gehler and Nowozin, 2009), we choose them based on a validation set (20% of training data) and the final model is retrained with the whole training set. We could not afford to test all possible combinations. The '$1/C_l$' in our model (**P1**) and the 'C' of the SVM for the *l-th* auxiliary task in LPB have similar functions in regularization[5] as discussed at the end of section 3. So, we can use a heuristic method (Gehler and Nowozin, 2009): selecting those values individually based on the validation set performance by training an SVM for the *l-th* task. For the global pattern, we select '$1/C_l$' from $10^4 \times \{2, 1.5, 1\}$. For local

[3] We also try to replace the linear programming with a quadratic programming and observe similar performance.

[4] The RBF kernel has the form $\exp(-\gamma\|x-y\|^2)$, where x and y are two vectors, and the parameter refers to γ.

[5] Notice that in **P1** 'C_l' is multiplied with $\|w_l\|^2$ and the 'C' of a typical SVM is multiplied with the error terms.

patterns, we use the set $10^{-2} \times \{1, \sqrt{2}/2, 1/2\}$. After selecting parameters for auxiliary tasks, we choose the 'C' in our model (**P1**) from $100 \times \{20, 15, 10, 5, 1\}$ and the 'ν' in ν-LPB from $\{0.01, 0.1, 1, 10, 100\}$ based on the validation set result.

4.1.4 Results and Discussions

The baseline (`Single`) method's AUC is 74.01% (std: 7.12%). We report the AUC's of different methods minus that of the baseline model (`Single`) and their standard deviations in Table 2. These results can be summarized as follows:

Table 2: AUC's (%) of Various Methods Minus That of Baseline (Diff. AUC)

Methods	`Simple`	`MTK`	`MKL`
Diff. AUC(std)	-0.39^* (3.34)	2.67^* (6.36)	1.73^* (3.50)
Methods	`MKL-ave`	`LPB`	`LPB-WOA`
Diff. AUC(std)	3.51^* (4.13)	6.17^* (6.27)	7.28^* (6.70)
Methods	`CMHA`	`CMHA-ave`	`CMHA-WOA`
Diff. AUC(std)	**9.42** (5.32)	**9.39** (5.70)	6.91^* (5.77)

A '$*$' sign indicates that the result of the corresponding method is significantly different from that of `CMHA` by a paired t-test at the 95% confidence level.

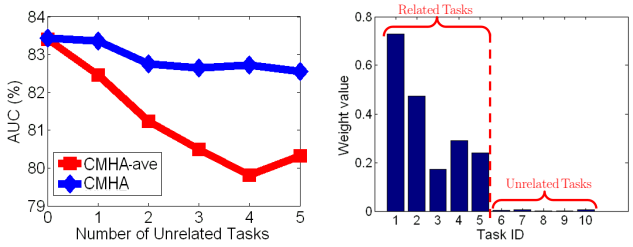
(a) Among methods without using auxiliary labels, `Simple` performs worse than the two MKL models, `LPB-WOA` and `CMHA-WOA`. When auxiliary labels are used, `MTK` performs significantly worse than `LPB` and `CMHA` (at the 95% confidence level by paired t-test). This shows that ignoring the "natural splitting" of the feature set (e.g., `Simple` and `MTK`), motivated from different learning targets, is not a very competitive scheme for this particular dataset.

(b) Both `CMHA-WOA` and `LPB-WOA` outperform `MKL`, and the reason is similar to that discussed by Gehler and Nowozin (2009): the Lagrange multiplier is restricted to be the same for all kernels in `MKL` but not in `CMHA-WOA` and `LPB-WOA` resulting in more flexible models.

(c) `CMHA` provides a statistically significant improvement compared to `LPB-WOA` which achieves the best results among methods without using the additional auxiliary task labels (i.e., nonoperational features). This shows the effectiveness of using nonoperational features by our learning framework.

(d) Our method's performance is comparable to LPB's when learning with main label only (a paired t-test shows that the difference between `CMHA-WOA` and `LPB-WOA` is not significant with a p-value of 0.6208). However, when including nonoperational features, `CMHA` provides an improvement over `CMHA-WOA`, while `LPB` performs worse than `LPB-WOA`. One plausible explanation is given by the main difference between these two schemes: `CMHA` learns the auxiliary tasks and main task together, allowing the auxiliary task classifier to adapt to the main task, while LPB learns them separately.

(e) The difference between the weighted scheme `CMHA` and the average scheme `CMHA-ave` is not significant and this is similar to MKL (Gehler and Nowozin, 2009). To see the advantage of `CMHA` over `CMHA-ave`, we add some randomly generated auxiliary tasks and apply the same experimental setting above to test `CMHA` and `CMHA-ave`. Fig. 2(a) shows that `CMHA` is more robust than `CMHA-ave` in the presence of unrelated tasks. Fig. 2(b) shows the weights (i.e., $\boldsymbol{d}$) learned by `CMHA` (average from the five-fold CV) when there are 5 unrelated tasks and obviously `CMHA` can successfully exclude the unrelated tasks.



(a) (b)

Figure 2: Comparing `CMHA` with `CMHA-ave` when unrelated auxiliary tasks present. (a) AUC vs. number of unrelated tasks; (b) The weights learned by `CMHA` (average from the five-fold CV). 1-5: related tasks; 6-10: randomly generated unrelated tasks.

4.2 Experiments and Results on Public Datasets

In this section, we demonstrate that our model is general and can be readily applied to other domains. We use the `CAL500` dataset[6] (Barrington et al., 2008) from the UCSD multiple kernel learning repository. `CAL500` is consisted of 502 songs and each song is annotated with its genre, emotion, instrument, etc.. We consider two main tasks: predicting whether a song is annotated with `electronica` and `alternative` respectively. We use the annotation `electric guiter` (including both `clean` and `distorted`) as the auxiliary label. We use the `MFCC` kernel as feature for main task and the `last.fm` kernel as feature for auxiliary task. Following Barrington et al. (2008), we test our method `CMHA` with ten-fold CV and use AUC as the performance measure. Barrington et al. (2008) reported results using MKL on combining four kernels including the `MFCC` kernel and the `last.fm` kernel. To be fair in comparison, we also apply `MKL` and `LPB` (section 4.1.3) only using these two kernels as that in `CMHA`. Regularization parameters are chosen as described above. As the results shown in Table 3, our method can still

[6] Available at `http://mkl.ucsd.edu/sites/default/files/cal500.tgz`

slightly outperform MKL and LPB when generalized to a new application.

Table 3: AUC's (%) of Various Methods on Two Music Genre Recognition Tasks

Methods	CMHA	MKL	LPB	(Barrington 2008)
electronica	**90.53**	89.03	88.27	86
alternative	**81.99**	81.83	81.21	81

5 Related Work

Several attempts have been made to integrate heterogenous data sources with MKL (Lanckriet et al., 2004), its p-norm extension (Kloft et al., 2009), and LPB (Demiriz et al., 2002; Gehler and Nowozin, 2009) for single task learning. In this study, we combine and extend these ideas to multi-task learning. In our model, Lagrange multipliers for different kernels can be different, unlike those in MKL which are required to be the same for all kernels. LPB (Demiriz et al., 2002) also allows different multipliers for different kernels (Gehler and Nowozin, 2009) and has been shown to achieve superior performance for image classification by combining heterogeneous feature spaces (Gehler and Nowozin, 2009). Our work, however, is different from Demiriz et al., (2002) mainly in two aspects: (a) the bound obtained by Demiriz et al. (2002) is from covering numbers, while our formulation is derived from Rademacher complexity; and (b) LPB optimizes each auxiliary classifier first, and combines their decisions later, thus resulting in an ensemble of auxiliary classifiers. Our model (**P1**), on the other hand, optimizes the auxiliary classifiers and builds a weighted combination for the main task simultaneously. This also differentiates our model from A-SVM (Yang et al., 2007) and the gating network approach (Bonilla et al., 2007). Furthermore, A-SVM focuses on homogeneous input space and computes the weights of auxiliary classifiers using unlabeled data and the gating network approach is designed to utilize task-specific features that are the same for all data from one task and are from a homogeneous space across all tasks.

Modeling task relatedness has attracted lots of interests because it is a critical factor allowing multi-task learning to outperform single task learning. In multi-task kernel (Evgeniou and Pontil, 2004; Evgeniou et al., 2006) and its similar models (Daumé, 2007; Finkel and Manning, 2009) task relatedness is modeled in the following way: let the main task classifier be h, the task relatedness is captured by $\|h - 1/(m+1)\sum_{i=1}^{m+1} w_i\|$. However, in our problem, the $\mathcal{X}_l$'s are different and therefore the average term $1/(m+1)\sum_{i=1}^{m+1} w_i$ is ill-defined (recall that $w_l \in \mathcal{X}_l$). Simply combining all features from all tasks may not always be a very competitive solution as shown in our experiments in contrast to the case of task-specific features considered by Bonilla et al. (2007). Evgeniou et al. (2006) briefly mentioned another solution mapping all $\mathcal{X}_l$'s into one common space, but a practical application of this approach is not shown by Evgeniou et al. (2006). Our model, however, will not suffer from this heterogeneous input space problem because it only uses the predictions from the auxiliary classifiers. Evgeniou et al. (2006) didn't consider this simple method because of a slight difference between their problem setting and ours: in our motivated application, for any two tasks l and r, both (x_i^l, y_i^l) and (x_i^r, y_i^r) are from the i-th training instance (lesion), while in previous methods (Evgeniou and Pontil, 2004; Evgeniou et al., 2006; Finkel and Manning, 2009; Daumé, 2007), (x_i^l, y_i^l) and (x_i^r, y_i^r) are two i.i.d. training instances. Thus, previous models (Evgeniou and Pontil, 2004; Evgeniou et al., 2006; Finkel and Manning, 2009; Daumé, 2007) did not use x_i^l $(1 \le l \le m)$ to predict the $(m+1)$-th task while our model uses that to predict the main task (see Eq. (1)).

The techniques developed by Srebro and Ben-david (2006), Ying and Campbell (2009), and Cortes et al. (2009) for error bound analysis of MKL could be applied to handle the terms (2.b) and (2.c) of $\mathcal{H}_1$ and to improve the fourth term in the bounds of Theorem 1(i),(ii). However, the major difficulty in deriving the error bounds of our model lies on the way to handle the data-dependent term (2.d) of $\mathcal{H}_1$. We relax $\mathcal{H}_1$ to a form similar to the function classes considered by Shivaswamy and Jebara (2010) and then use the landmark set method (Shivaswamy and Jebara, 2010) to deal with data-dependent regularization terms.

6 Conclusions and Future Work

Viewing nonoperational features as auxiliary tasks was initially proposed by Caruana (1997). Starting from the simple idea of expressing the main task classifier as a weighted sum of the prediction values of the auxiliary classifiers, we devise a learning framework (with proven error bounds) that allows the use of nonoperational features. When the method is applied to skin cancer screening we obtain very encouraging results. One immediate extension of our model is to incorporate unlabeled data as that in co-training and multi-view learning. Obtaining shaper error bounds and hyperparameter selection for our model and LPB are also interesting future topics.

Acknowledgements

This work was supported in part by NIH grant no. 1R21AR057921, and by grants from UH-GEAR and the Texas Learning and Computation Center at the University of Houston.

References

G. Argenziano, HP Soyer, V. De Giorgi, D. Piccolo, P. Carli, M. Delfino, A. Ferrari, R. Hofmann-Wellenhof, D. Massi, G. Mazzocchetti, et al., *Dermoscopy: a tutorial*, vol. 12, 2000.

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

F.R. Bach. Consistency of the Group Lasso and Multiple Kernel Learning. *Journal of Machine Learning Research*, 9:1179–1225, 2008.

L. Barrington, M. Yazdani, D. Turnbull, and G. Lanckriet. Coombining Feature Kernels for Semantic Music Retrieval. In *Proceedings of the 9th International Society for Music Information Retrieval Conference*, 2008.

E.V. Bonilla, F.V. Agakov, and C.K.I. Williams. Kernel Multi-task Learning using Task-specific Features. In *Proceedings of the 11th International Conference on Artificial Intelligence and Statistics*, 2007.

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

C. Cortes, M. Mohri, and A. Rostamizadeh. Generalization Bounds for Learning Kernels. In *Proceedings of the 27th International Conference on Machine Learning*, ACM, 2010.

H. Daumé. Frustratingly easy domain adaptation. In *Annual meeting-association for computational linguistics*, volume 45, pp. 256, 2007.

A. Demiriz, K.P. Bennett, and J. Shawe-Taylor. Linear programming boosting via column generation. *Machine Learning*, 46(1):225–254, 2002.

T. Evgeniou, and M. Pontil. Regularized multi–task learning. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 109–117. ACM, 2004.

T. Evgeniou, C.A. Micchelli, and M. Pontil. Learning multiple tasks with kernel methods. *Journal of Machine Learning Research*, 6(1):615, 2006.

D.G. Federman, J.D. Kravetz, and R.S. Kirsner. Skin cancer screening by dermatologists: prevalence and barriers. *Journal of the American Academy of Dermatology*, 46(5):710, 2002.

J.R. Finkel and C.D. Manning. Hierarchical bayesian domain adaptation. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 602–610. Association for Computational Linguistics, 2009.

P. Gehler, and S. Nowozin. On Feature Combination for Multiclass Object Classification. In *IEEE International Conference on Computer Vision*, 2009.

R.H. Johr. Dermoscopy: alternative melanocytic algorithmsthe ABCD rule of dermatoscopy, Menzies scoring method, and 7-point checklist. *Clinics in dermatology*, 20(3):240–247, 2002.

M. Kloft, U. Brefeld, S. Sonnenburg, P. Laskov, K.R. Müller, , and A. Zien. Efficient and accurate lp-norm multiple kernel learning. In *Advances in Neural Information Processing Systems 22*, pp. 997–1005, 2009.

G. Lanckriet, N. Cristianini, P. Bartlett, L. El Ghaoui, and M.I. Jordan. Learning the Kernel Matrix with Semidefinite Programming. *Journal of Machine Learning Research*, 5:27–72, 2004.

W.J. Li, and D.Y. Yeung. Localized Content-Based Image Retrieval Through Evidence Region Identification. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2009.

O. Maron, and T. Lozano-Pérez. A framework for multiple-instance learning. In *Advances in Neural Information Processing Systems*, pp. 570–576, 1998.

C.A. Micchelli, and M. Pontil. Feature space perspectives for learning the kernel. *Machine Learning*, 66(2): 297–319, 2007.

T. Ojala, M. Pietikäinen, and T. Mäenpää. Multiresolution Gray-Scale and Rotation Invariant Texture Classification with Local Binary Patterns. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 24(7):971, 2002.

A. Rakotomamonjy, F. Bach, Y. Grandvalet, and S. Canu. SimpleMKL. *Journal of Machine Learning Research*, 9:2491–2521, 2008.

P.K. Shivaswamy, and T. Jebara. Maximum Relative Margin and Data-Dependent Regularization. *Journal of Machine Learning Research*, 11:747–788, 2010.

S. Sonnenburg, G. Rätsch, C. Schäfer, and B. Schölkopf. Large scale multiple kernel learning. *The Journal of Machine Learning Research*, 7:1565, 2006.

N. Srebro and S. Ben-david. Learning bounds for support vector machines with learned kernels. In *Annual Conference On Learning Theory (COLT)*, pp. 169–183, 2006.

R.J. Stanley, W.V. Stoecker, and R.H. Moss. A relative color approach to color discrimination for malignant melanoma detection in dermoscopy images. *Skin Research and Technology*, 13(1):62–72, 2007.

J. Yang, R. Yan, and A.G. Hauptmann. Cross-domain video concept detection using adaptive svms. In *Proceedings of the 15th international conference on Multimedia*, pp. 197. ACM, 2007.

Y. Ying, and C. Campbell. Generalization bounds for learning the kernel. In *Proc. of the 22nd Annual Conference on Learning Theory*, 2009.

M. Yuan, and Y. Lin. Model selection and estimation in regression with grouped variables. *Journal-Royal Statistical Society Series B Statistical Methodology*, 68 (1):49, 2006.

X. Yuan, Z. Yang, G. Zouridakis, and N. Mullani. SVM-based texture classification and application to early melanoma detection. In *Annual International Conference of the IEEE Engineering in Medicine and Biology Society.*, volume 1, pp. 4775, 2006.

J. Zhang, Z. Ghahramani, and Y. Yang. Flexible latent variable models for multi-task learning. *Machine Learning*, 73(3):221–242, 2008.

A. Zien, and C.S. Ong. Multiclass multiple kernel learning. In *Proceedings of the 24th International Conference on Machine Learning*, pp. 1198. ACM, 2007.

G. Zouridakis, M. Doshi, and N. Mullani. Early diagnosis of skin cancer based on segmentation and measurement of vascularization and pigmentation in nevoscope images. In *26th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, volume 1, 2004.

APPENDIX–SUPPLEMENTARY MATERIAL

To convert **P1** into a convex optimization problem, we can simply replace w_l , v_l $(1 \le l \le m+1)$, and ξ_i^l $(1 \le i \le n$ and $1 \le l \le m)$ with $\hat{w}_l/d_l$, $\hat{v}_l/d_l$, and $\hat{\xi}_i^l/d_l$, respectively. If $d_l = 0$, we define $a/d_l = \infty$ when $a \neq 0$, and $a/d_l = 0$ when $a = 0$. Omitting the *hat* notation in $\hat{w}$, $\hat{v}$, and $\hat{\xi}$ for simplicity, **P1** becomes

$$\textbf{P2} \quad \min_{\boldsymbol{w},\boldsymbol{v},\boldsymbol{\xi},\boldsymbol{d}} \frac{1}{2}\sum_{l=1}^{m}\frac{\|v_l\|^2}{d_l} + \frac{1}{2}\sum_{l=1}^{m}\frac{C_l\|w_l\|^2}{d_l}$$

$$+ \frac{1}{2}\frac{\|w_{m+1}\|^2}{d_{m+1}} + \frac{1}{2}\sum_{l=1}^{m}\sum_{i=1}^{n}\frac{(\xi_i^l)^2}{d_l} + C\sum_{i=1}^{n}(\xi_i^{m+1}) \qquad (3)$$

$$\textbf{s.t.} \quad y_i^l(w_l^T\phi_l(x_i^l)) \ge d_l - \xi_i^l,\ 1 \le l \le m;\ 1 \le i \le n \qquad (4)$$

$$y_i^{m+1}\Big(\sum_{l=1}^{m+1}(w_l + v_l)\phi_l(x_i^l)\Big) \ge 1 - \xi_i^{m+1},\ 1 \le i \le n \qquad (5)$$

$$\boldsymbol{d} \ge \mathbf{0}, \qquad \|\boldsymbol{d}\|_p^p \le 1, \qquad \boldsymbol{\xi} \ge \mathbf{0} \qquad (6)$$

Now it is clear from **P2** that the quadratic error weighted by '$\boldsymbol{d}$' (see Eq. 2.d) makes it easy to formulate a convex problem. Without the '$\boldsymbol{d}$' in Eq. (2.d), the error terms for auxiliary tasks in **P2** become $\left(\xi_i^l\right)^2/d_l^2$ which is not convex. It is also important to note that analytically eliminating '$\boldsymbol{d}$' in **P2** is not so simple as that in MKL (nor its p-norm variant considered by Micchelli and Pontil (2007)) with the technique proposed by Rakotomamonjy et al. (2008), and Micchelli and Pontil (2007), because '$\boldsymbol{d}$' presents in both the objective function and the auxiliary task's constraint of **P2**.

Fixing $\boldsymbol{d}$, we can solve the partial Lagrangian w.r.t. $\boldsymbol{w}, \boldsymbol{v}$, and $\boldsymbol{\xi}$, and we obtain the semi-infinite programming as follows:

$$\textbf{P3} \ \min_{\boldsymbol{d},\rho} \ \rho \qquad \textbf{s.t.} \qquad \boldsymbol{d} \ge \mathbf{0}, \quad \|\boldsymbol{d}\|_p^p \le 1,$$
$$\textit{and} \quad \rho \ge \Big(d_1\boldsymbol{e}^T, d_2\boldsymbol{e}^T, \ldots, d_m\boldsymbol{e}^T, \boldsymbol{e}^T\Big)\boldsymbol{\alpha} - \frac{1}{2}\Big(\sum_{l=1}^{m+1} d_l Q_l\Big)$$
$$\text{for all} \quad \boldsymbol{\alpha} \text{ satisifying } \mathbf{0} \le \boldsymbol{\alpha}, \text{ and } \boldsymbol{\alpha}^{m+1} \le C$$

where $\boldsymbol{\alpha}$ is the Lagrange multiplier vector such that $\boldsymbol{\alpha} = \Big((\boldsymbol{\alpha}^l)_{(l=1)}^{(m+1)}\Big)$, and for each l, $\boldsymbol{\alpha}^l = \Big((\alpha_i^l)_{(i=1)}^{(n)}\Big)$, and

$$Q_l = \Big(\boldsymbol{\alpha}^l\Big)^T\left(\tfrac{1}{C_l}K_l \circ \Big(\boldsymbol{y}^l(\boldsymbol{y}^l)^T\Big) + \boldsymbol{I}\right)\boldsymbol{\alpha}^l + \overline{C}_l\Big(\boldsymbol{\alpha}^{m+1}\Big)^T\left(K_l \circ \Big(\boldsymbol{y}^{m+1}(\boldsymbol{y}^{m+1})^T\Big)\right)\boldsymbol{\alpha}^{m+1} + \tfrac{2}{C_l}\Big(\boldsymbol{\alpha}^l\Big)^T\left(K_l \circ \Big(\boldsymbol{y}^l(\boldsymbol{y}^{m+1})^T\Big)\right)\boldsymbol{\alpha}^{m+1},$$

where $\overline{C}_l = 1/C_l + 1$, for $1 \le l \le m$. When $l = m+1$,

$$Q_{m+1} = \Big(\boldsymbol{\alpha}^{m+1}\Big)^T\left(K_{m+1} \circ \Big(\boldsymbol{y}^{m+1}(\boldsymbol{y}^{m+1})^T\Big)\right)\boldsymbol{\alpha}^{m+1},$$

where "$\circ$" denotes the element-wise product between two matrices.