Multi-Task Multiple Kernel Relationship Learning

Keerthiram Murugesan*   Jaime Carbonell*

Abstract

This paper presents a novel multitask multiple kernel learning framework that efficiently learns the kernel weights leveraging the relationship across multiple tasks. The idea is to automatically infer this task relationship in the *RKHS* space corresponding to the given base kernels. The problem is formulated as a regularization-based approach called *Multi-Task Multiple Kernel Relationship Learning* (*MK-MTRL*), which models the task relationship matrix from the weights learned from latent feature spaces of task-specific base kernels. Unlike in previous work, the proposed formulation allows one to incorporate prior knowledge for simultaneously learning several related task. We propose an alternating minimization algorithm to learn the model parameters, kernel weights and task relationship matrix. In order to tackle large-scale problems, we further propose a two-stage *MK-MTRL* online learning algorithm and show that it significantly reduces the computational time, and also achieves performance comparable to that of the joint learning framework. Experimental results on benchmark datasets show that the proposed formulations outperform several state-of-the-art multitask learning methods.

1 Introduction

There have been two main lines of work in multi-task learning: First, learn a shared feature representation across all the tasks, leveraging low-dimensional subspaces in the feature space [1, 9, 17, 23]. Second, learn the relationship between the tasks to improve the performance of the related tasks [19, 26, 28, 29]. Pairwise task relationships provide very useful information for characterizing and transferring information to similar tasks.

Despite the expressive power of these two different research directions, the learning space is restricted to a single kernel (per task), chosen by the user, that corresponds to a *RKHS* space. Multiple Kernel Learning (*MKL*), on the other hand, allows the user to specify a family of base kernels related to an application, and to use the training data to automatically learn the optimal combination of these kernels. We learn the weights of the base kernels along with the model parameters in

a single joint optimization problem. There is a large body of work in the recent years addressing several aspects of this problem, such as efficient learning of the kernel weights, fast optimization and providing better theoretical guarantees [2, 4, 11, 12, 15, 18, 22].

Recent work in multiple kernel learning in a multitask framework focuses on sharing common representations and assumes that the tasks are all related [10]. The motivation for this approach stems from multitask feature learning that learns joint feature representation shared across multiple tasks [1, 23]. Unfortunately, the assumption that all the tasks are related and share a common feature representation is too restrictive for many real-world applications. Similarly, based on previous work [29], one can extend the traditional multitask relationship learning *MTRL* with multiple task-specific base kernels. There are two main problems with such naive approach: First, the unknown variables (task model parameters, kernel weights and task relationship matrix) are intertwined in the optimization problem, and thus making it difficult to learn for large-scale applications. Furthermore, the task relationship matrix is learned in the original feature space rather than in the kernel spaces. We show in this paper, that learning the relationship between the kernel spaces empirically performs better than relations among the original feature spaces.

There have been a few attempts to imposing higher-order relationship between kernel spaces using kernel weights. Kloft et al. [12] propose a non-isotropic norm such as $\sqrt{\boldsymbol{\beta}^{\top}\Sigma^{-1}\boldsymbol{\beta}}$ on kernels weights $\boldsymbol{\beta}$ to induce the relationship between the base kernels in Reproducing Kernel Hilbert Spaces. For example, in neuroimaging, a set of base kernels are derived from several medical imaging modalities such as MRI, PET etc., or image processing methods such as morphometric or anatomical modeling, etc. Since some of the kernel functions share similar parameters such as patient information, disease progression stage, etc., we can expect that these base kernels are correlated based on how they were constructed. Such information can be obtained from medical domain experts as a part of the disease prognosis which then can be used as a prior knowledge Σ. Previous work either assumes Σ as a diagonal matrix or

*School of Computer Science, Carnegie Mellon University, Pittsburgh, PA 15213, USA. {*kmuruges, jgc*}@cs.cmu.edu

requires prior knowledge from the experts on the interaction of kernels [8, 12]. Unfortunately, such prior knowledge is not easily available in many applications either because it is time-consuming or it is expensive to elicit. [13]. In such applications, we want to induce this relationship matrix from the data along with the kernel weights and model parameters.

This paper addresses these problems with a novel regularization-based approach for multitask multiple kernel learning framework, called *multitask multiple kernel relationship learning* (*MK-MTRL*), which models the task relationship matrix from the weights learned from the latent feature spaces of task-specific base kernels. The idea is to automatically infer task relationships in (*RKHS*) spaces from their base kernels. We first propose an alternating minimization algorithm to learn the model parameters, kernel weights and task relationship matrix. The method uses a *wrapper* approach which efficiently uses any off-the-shelf *SVM* solver (or any kernel machine) to learn the task model parameters. However, like previous work, the proposed iterative algorithm suffers from scalability challenges. The run-time complexity of the algorithm increases with the number of tasks and the number of base kernels per task, as it needs these base kernels in memory to learn the kernel weights and the task relationship matrix.

For large-scale applications such as object detection, we introduce a novel two-stage online learning algorithm based on recent work [14] that learns the kernel weights independently from the model parameters. The first stage learns a good combination of base kernels in an online setting and the second stage uses the learned weights to estimate a linear combination of the base kernels, which can be readily used with a standard kernel method such as *SVM* or kernel ridge regression [5, 6]. We provide strong empirical evidence that learning the task relationship matrix in the RKHS space is beneficial for many applications such as stock market prediction, visual object categorization, etc. On all these applications, our proposed approach outperforms several state-of-the-art multitask learning baselines. It is worth noting that the proposed multitask multiple kernel relationship learning can be readily applied for heterogeneous and multi-view data with no modification to the proposed framework [7, 27].

The rest of the paper is organized as follows: we provide a brief overview of multitask multiple kernel learning in the next section. In section 3, we discuss the proposed model *MK-MTRL*, followed by our two-stage online learning approach in section 4. We then show comprehensive evaluations of the proposed model against the six baselines on several benchmark datasets in section 6.

2 Preliminaries

Before introducing our approach, we briefly review the multitask multiple kernel learning framework in this section. Suppose there are T learning tasks available with the training set $\mathcal{D} = \{(\mathbf{x}_{ti}, \boldsymbol{y}_{ti}), i = 1 \ldots N_t, t = 1 \ldots T\}$, where $\boldsymbol{x}_{ti}$ is the ith samples from the task t and it's corresponding output $\boldsymbol{y}_{ti}$. Let $\{\mathcal{K}_{tk}\}^{1 \leq k \leq K}$ be a set of task-specific base kernels, induced by the kernel mapping function $\phi_k(\cdot)$ on t^{th} task data. The objective of multitask multiple kernel learning problem is to learn a *good* linear combination of the task-specific base kernels $\sum_k \beta_{tk} \mathcal{K}_{tk}, \beta_{tk} \geq 0$ using the relationship between the tasks.

In addition to the non-negative constraints on β_{tk}, we need to impose an additional constraint or penalty to ensure that the units in which the margins are measured are meaningful (assuming that the base kernels are properly normalized). Recent work in *MKL* employs $\|\boldsymbol{\beta}\|_2^2$ to constrain the kernel weights. A direct extension of ℓ_2 regularized *MKL* to multi-task framework is given as follows [1]:

(2.1)
$$\min_{\mathbf{B} \geq 0} \min_{\mathbf{W}, \mathbf{c}, \xi} \sum_{t=1}^{T} \left(\frac{1}{2} \sum_{k=1}^{K} \frac{\|\mathbf{w}_{tk}\|_{\mathcal{H}_k}^2}{\beta_{tk}} + C \sum_{i=1}^{N_t} \xi_{ti} + \frac{\mu}{2} \|\boldsymbol{\beta}_t\|_2^2 \right)$$

$$\text{s.t.,} \ y_{ti}(\sum_{k=1}^{K} w_{tk}^\top \phi_k(\mathbf{x}_{ti}) + c_t) \geq 1 - \xi_{ti}, \ \xi_{ti} \geq 0$$

where $\mathcal{H}_k$ is the reproducing kernel Hilbert space associated with the kth kernel function and $\|\cdot\|_{\mathcal{H}_k}^2$ is the squared *RKHS* norm.

Similarly, we can use a general ℓ_p norm constraint with $p > 1$ on the kernel weights ($\|\boldsymbol{\beta}\|_p^2$). This can be thought of as a simple extension of ℓ_p-*MKL* to multitask setting [11]. Without any additional structural constraints on β_{tk}, the kernel weights are learned independently for each task and thus does not efficiently use the relationship between the tasks. Hence, we call the model in equation (2.1) as Independent Multiple Kernel Learning (*IMKL*).

Jawanpuria and Nath [10] proposed Multi-task Multiple Kernel Feature Learning (*MK-MTFL*), that employs mixed $(\ell_1, \ell_p), p \geq 2$ norm regularizer over the *RKHS* norm of the feature loadings corresponding to the tasks and the base kernels. The mixed norm regularization promotes a shared feature representations to combine the given set of task-specific base kernels.

The ℓ_p-norm regularizer learns the unequal weighting across the tasks, where as, ℓ_1-norm regularizer over the ℓ_p-norm leads to learning the shared kernel among the tasks.

The objective function for *MK-MTFL* is given as follows:

$$
(2.2) \quad \min_{\mathbf{W},\mathbf{c},\xi} \left(\frac{1}{2} \sum_{k=1}^{K} \Big(\sum_{t=1}^{T} \|\mathbf{w}_{tk}\|_2^p \Big)^{\frac{1}{p}} \right)^2 + C \sum_{t=1}^{T} \sum_{i=1}^{N_t} \xi_{ti}
$$
$$
\text{s.t., } y_{ti}(\sum_{k=1}^{K} w_{tk}^\top \phi_k(\mathbf{x}_{ti}) + c_t) \geq 1 - \xi_{ti}, \xi_{ti} \geq 0
$$

Note that the above objective function employs an ℓ_1-norm across the base kernels and ℓ_p norm across tasks. The above optimization problem can be equivalently written in the dual space as follows:

$$
(2.3) \quad \min_{\gamma \in \Delta_K} \max_{\lambda_j \in \Delta_{T,\bar{p}}} \max_{0 \leq \alpha \leq C} g(\lambda, \alpha, \gamma)
$$
$$
\text{s.t., } \alpha_t^\top \mathbf{y}_t = 0,
$$

where,

$$
g(\lambda, \alpha, \gamma) = \sum_{t=1}^{T} \left\{ \mathbf{1}^\top \alpha_t - \frac{1}{2} \alpha_t^\top \mathbf{Y}_t \Big[\sum_{k=1}^{K} \frac{\gamma_k \mathcal{K}_{tk}}{\lambda_{tk}} \Big] \mathbf{Y}_t \alpha_t \right\}
$$

Here $\boldsymbol{\alpha}_t$ is a vector of Langragian multipliers for the tth task, and corresponds to N_t constraints on the task data. $\mathbf{Y}_t$ is a diagonal matrix with entries as $\mathbf{y}_t$ and $\mathcal{K}_{tk}$ is the gram matrix of the tth task data w.r.t the kth kernel function. More specifically, $\boldsymbol{\gamma}$ selects the base kernels that are important for all the tasks, where as $\boldsymbol{\lambda}$ selects the base kernels that are specific to individual tasks. With this representation, *MK-MTFL* can be seen as a multiple kernel generalization to the multi-level multi-task learning proposed by Lozano and Swirszcz (2012) [23].

3 Multi-task Multiple Kernel Relationship Learning (*MK-MTRL*)

This section presents the details of the proposed model *MK-MTRL*. Since multitask learning seeks to improve performance of each task with the help of other *related* tasks, it is desirable in multiple kernel learning for the multitask framework to have a structural constraints on the task kernel weights β_{tk} to promote sharing of information from other related tasks. Note that the proposed approach is significantly different from the traditional *MTRL*, as explained in the introduction.

When prior knowledge on task relationship is available, the multiple kernel multitask learning model should incorporate this information for simultaneously learning several related tasks. Neither the *IMKL* or *MK-MTFL* consider the pairwise task relationship such as positive task correlation, negative task correlation, and task independence when learning the kernel weights for combining the base kernels. Based on the assumption that similar tasks are likely to give similar importance to their base kernels (and thereby, their respective *RKHS* spaces), we consider a regularization on the task kernel weights $tr(\boldsymbol{B}\boldsymbol{\Omega}^{-1}\boldsymbol{B}^\top)$, where, for notational convenience, we write $\boldsymbol{B} = \{\boldsymbol{\beta}_1, \boldsymbol{\beta}_2, \ldots, \boldsymbol{\beta}_T\}$. Mathematically, the proposed *MK-MTRL* formulation is written as follows:

$$
\min_{\boldsymbol{\Omega}, \mathbf{B} \geq 0} \min_{\mathbf{W},\mathbf{c},\xi} \sum_{t=1}^{T} \left(\frac{1}{2} \sum_{k=1}^{K} \frac{\|\mathbf{w}_{tk}\|_{\mathcal{H}_k}^2}{\beta_{tk}} + C \sum_{i=1}^{N_t} \xi_{ti} \right)
$$
$$
+ \frac{\mu}{2} tr(\mathbf{B}\boldsymbol{\Omega}^{-1}\mathbf{B}^\top)
$$
$$
(3.4) \quad \text{s.t., } y_{ti}(\sum_{k=1}^{K} w_{tk}^\top \phi_k(\mathbf{x}_{ti}) + c_t) \geq 1 - \xi_{ti}, \xi_{ti} \geq 0
$$
$$
\boldsymbol{\Omega} \succeq 0,
$$
$$
tr(\boldsymbol{\Omega}) \leq 1
$$

The key difference from the *IMKL* model is that the standard (squared) ℓ_p norm on $\boldsymbol{\beta}_t$ is replaced with a more meaningful structural penalty that incorporates the task relationship. Unlike in *MK-MTFL*, the shared information among the task is separate from the core problem (T SVMs). Here, $\boldsymbol{\Omega}$ encodes the task relationship such that similar tasks are forced to have similar kernel weights. It is easy to see that when $\boldsymbol{\Omega} = \mathbb{I}_{T \times T}$, the above problem reduces to equation (2.1).

3.1 MK-MTRL in Dual Space
In this section, we consider the proposed approach in the dual space. By writing the above objective function in Lagrangian form and introducing Lagrangian multiplier α_{tk} for the constraints, we can write the corresponding dual objective function as:

$$
\min_{\boldsymbol{\Omega}, \mathbf{B} \geq 0} \max_{0 \leq \alpha \leq C} h(\alpha, \mathbf{B}) + \frac{\mu}{2} tr(\mathbf{B}\boldsymbol{\Omega}^{-1}\mathbf{B}^\top)
$$
$$
(3.5) \quad \text{s.t., } \alpha_t^\top \mathbf{y}_t = 0,
$$
$$
\boldsymbol{\Omega} \succeq 0,
$$
$$
tr(\boldsymbol{\Omega}) \leq 1
$$

where,

$$
h(\alpha, \mathbf{B}) = \sum_{t=1}^{T} \left\{ \mathbf{1}^\top \alpha_t - \frac{1}{2} \alpha_t^\top \mathbf{Y}_t \Big(\sum_{k=1}^{K} \beta_{tk} \mathcal{K}_{tk} \Big) \mathbf{Y}_t \alpha_t \right\}
$$

The above objective function is a bi-convex optimization problem. Note that we can further reduce the problem by eliminating α_{tk}, then the dual problem becomes:

$$\min_{\boldsymbol{\Omega}} \max_{0 \leq \alpha \leq C} \sum_{t=1}^{T} \left\{ \mathbf{1}^{\top} \alpha_t - \frac{1}{2} \|\mathcal{G}\|_{\Omega} \right\}$$

(3.6)
$$\text{s.t., } \alpha_t^{\top} \mathbf{y}_t = 0,$$
$$\boldsymbol{\Omega} \succeq 0,$$
$$tr(\boldsymbol{\Omega}) \leq 1$$

where $\mathcal{G}_{tk} = \beta_{tk}\alpha_t^{\top}\mathbf{Y}_t\mathcal{K}_{tk}\mathbf{Y}_t\alpha_t$ which corresponds to $\frac{\|\boldsymbol{w}_{tk}\|_2^2}{\beta_{tk}}$ in the primal space and we write $\|\mathcal{G}\|_{\Omega} = \sqrt{tr(\mathcal{G}\boldsymbol{\Omega}\mathcal{G}^{\top})}$. We will use this representation for deriving closed-form solution for the task kernel weights $\boldsymbol{B}$

3.2 Optimization We use an alternating minimization procedure for learning the kernel weights and the model parameters iteratively. We implement a two-layer *wrapper* approach commonly used in these *MKL* solvers for our problem. The wrapper methods alternate between minimizing the primal problem (3.4) w.r.t $\boldsymbol{\beta}_t$ via a simple analytical update step and minimizing all other variables in terms of the dual variables $\boldsymbol{\alpha}_t$ from equation (3.5).

When $\{\boldsymbol{B}, \boldsymbol{\Omega}\}$ are fixed, *MK-MTRL* equation (3.5) reduces to T independent sub-problems. One can use any conventional *SVM* solver (or any kernel method) to optimize for $\boldsymbol{\alpha}_t$ independently. We focus on optimizing the kernel coefficients $\boldsymbol{B}$ and $\boldsymbol{\Omega}$ next.

Optimizing w.r.t $\boldsymbol{B}$ when $\{\boldsymbol{\alpha}, \boldsymbol{\Omega}\}$ are fixed Given $\{\boldsymbol{\alpha}, \boldsymbol{\Omega}\}$, we find $\boldsymbol{B}$ by setting the gradient of equation (3.4) w.r.t $\boldsymbol{B}$ to zero and we get:

(3.7)
$$\boldsymbol{B} = \frac{1}{\mu}(\mathcal{W} \circ \boldsymbol{B}^{-2})\boldsymbol{\Omega}$$

where $\boldsymbol{B}^{-2} = \{\beta_{kt}^{-2}, 1 \leq k \leq K, 1 \leq t \leq T\}$, $\mathcal{W}_{tk} = \|\mathbf{w}_{tk}\|_{\mathcal{H}_k}^2$ and $\boldsymbol{A} \circ \boldsymbol{B}$ is an element-wise product operation.

By incorporating the last term in equation (3.4) into the constraint set, we can eliminate the regularization parameter μ to obtain an analytical solution for $\boldsymbol{B}$. Because $\boldsymbol{\Omega} \succeq 0$ and $\boldsymbol{B} \geq 0$, the constraint $tr(\boldsymbol{B}\boldsymbol{\Omega}^{-1}\boldsymbol{B}^{\top}) \leq 1$ must be active at optimality. We can now use the above equation to solve for μ.

(3.8)
$$\boldsymbol{B} = \frac{(\mathcal{W} \circ \boldsymbol{B}^{-2})\boldsymbol{\Omega}}{\sqrt{tr((\mathcal{W} \circ \boldsymbol{B}^{-2})\boldsymbol{\Omega}(\mathcal{W} \circ \boldsymbol{B}^{-2})^{\top})}}$$

Since the task relationship matrix is independent of the number of base kernels K, one may use the above closed-form solution when the number of tasks is small.

For some applications, it may be desirable to employ an iterative approach such as first-order method (*FISTA*) or second-order method (*Newton's*). The parameter μ can be easily learned by cross-validation.

Optimizing w.r.t $\boldsymbol{\Omega}$ when $\{\boldsymbol{\alpha}, \boldsymbol{B}\}$ are fixed In the final step of the optimization, we fix $\boldsymbol{\alpha}$ and $\boldsymbol{B}$ and solve the problem w.r.t $\boldsymbol{\Omega}$. By taking the partial derivative of the objective function with respect to $\boldsymbol{\Omega}$ and setting it zero, we get an analytical solution for $\boldsymbol{\Omega}$ [29]:

(3.9)
$$\boldsymbol{\Omega} = \frac{(\boldsymbol{B}^{\top}\boldsymbol{B})^{\frac{1}{2}}}{tr((\boldsymbol{B}^{\top}\boldsymbol{B})^{\frac{1}{2}})}$$

Substituting the above solution in equation 3.4, we can see that the the objective function of *MK-MTRL* is related to the trace norm regularization. Instead of ℓ_p norm regularization (as in L_p-MKL) or mixed-norm regularization (as in *MK-MTFL*), our model seeks a low-rank $\boldsymbol{B}$, using $\|\boldsymbol{B}\|_*$, such that similar base kernels are selected among the similar tasks.

4 Two-Stage Multi-task Multiple Kernel Relationship Learning

The proposed optimization procedure in the previous section involves T independent *SVM* (or *kernel ridge regression*) calls, followed by two closed-form expressions for jointly learning the kernel weights $\boldsymbol{B}$, task relationship matrix $\boldsymbol{\Sigma}$ and the task parameters $\boldsymbol{\alpha}$. Even though this approach is simple and easy to implement, it requires the precomputed kernel matrices to be loaded into memory for learning the kernel weights. This could add a serious computational burden especially when the number of tasks T is large [25].

In this section, we consider an alternative approach to address this problem inspired by [5, 6]. It follows a two-stage approach: first, we independently learn the weights of the given task-specific base kernels using the training data and then, we use the weighted sum of these base kernels in a standard kernel machines such as *SVM* or *kernel ridge regression* to obtain a classifier. This approach significantly reduces the amount of computational overhead involved in the traditional multiple kernel learning algorithms that estimate the kernel weights and the classifier by solving a joint optimization problem.

We propose an efficient binary classification framework for learning the weights of these task-specific base kernels, based on target alignment [6]. The proposed framework formulates the kernel learning problem as a linear classification in the kernel space (so called $\mathcal{K}$-classifier). In this space, any task classifier with weight parameters directly corresponds to the task kernel weights.

Algorithm 1: Wrapper method for Multi-task Multiple Kernel Relationship Learning (*MK-MTRL*)

Input : *Base kernels* $\{\mathcal{K}_{tk}\}_{1\leq t\leq T}^{1\leq k\leq K}$,
labels $\{\boldsymbol{y}_t\}_{t=1}^T$,
regularization parameter $\mu > 0$

Output: $\boldsymbol{\alpha}, \boldsymbol{B}, \boldsymbol{\Omega}$

1 *Initialize* $\boldsymbol{\Omega} = \frac{1}{T}\mathbb{I}_{T\times T}$;

2 repeat

3   **repeat**

4   *Set* $\mathcal{K}_t \leftarrow \sum_{k=1}^K \beta_{tk}\mathcal{K}_{tk}, \forall t \in [T]$;

5   *Solve for* $\boldsymbol{\alpha}_t, t \in [T]$
     (4.10)
$$\max_{0\leq\boldsymbol{\alpha}_t\leq C,\boldsymbol{\alpha}_t^\top \boldsymbol{y}_t=0}\left\{\mathbf{1}^\top\alpha_t - \frac{1}{2}\alpha_t^\top \mathbf{Y}_t\mathcal{K}_t\mathbf{Y}_t\alpha_t\right\} \;(SVM)$$

6   *Solve for* $\boldsymbol{B} = \{\boldsymbol{\beta}_1, \boldsymbol{\beta}_2, \ldots, \boldsymbol{\beta}_T\}$,
     (4.11)
$$\min_{\mathbf{B}\geq 0}\frac{1}{2}\sum_{t=1}^T\sum_{k=1}^K\frac{\|\mathbf{w}_{tk}\|_{\mathcal{H}_k}^2}{\beta_{tk}} + \frac{\mu}{2}tr(\mathbf{B}\boldsymbol{\Omega}^{-1}\mathbf{B}^\top)$$

     where $\|\mathbf{w}_{tk}\|_{\mathcal{H}_k}^2 = \beta_{tk}^2\alpha_t^\top \mathbf{Y}_t\mathcal{K}_{tk}\mathbf{Y}_t\alpha_t$

7   **until** *converges*;

8   *Solve for* $\boldsymbol{\Omega}$,
    (4.12)
$$\min_{\boldsymbol{\Omega}\succeq\mathbf{0},tr(\boldsymbol{\Omega})\leq 1} tr(\boldsymbol{\Omega}^{-1}(\boldsymbol{B}^\top \boldsymbol{B}))$$

9 until *converges*;

Algorithm 2: Two-stage, online learning of (*MK-MTRL*)

Input : *Base kernels* $\{\mathcal{K}_{tk}\}_{1\leq t\leq T}^{1\leq k\leq K}$,
labels $\{\boldsymbol{y}_t)\}_{t=1}^T$,
regularization parameter $\mu > 0$,
Number of rounds R

Output: $\boldsymbol{\alpha}, \boldsymbol{B}, \boldsymbol{\Omega}$

1 Initialize $\boldsymbol{\beta}_t^{(1)} = \mathbf{0}, \boldsymbol{\Omega} = \frac{1}{T}\mathbb{I}_{T\times T}$;

2 for $r = 1\ldots R$ **do**

3   *Construct* $(z_{t,ii'}, l_{t,ii'})$ *using* $\mathcal{K}$ *for any two examples* $(\mathbf{x}_{ti}, y_{ti})$ *and* $(\mathbf{x}_{ti'}, y_{ti'})$ *and for any task* t, *where*
$$z_{t,ii'} = \{\mathcal{K}_1(\mathbf{x}_{ti}, \mathbf{x}_{ti'}), \mathcal{K}_2(\mathbf{x}_{ti}, \mathbf{x}_{ti'}),$$
$$(4.13)\qquad \ldots, \mathcal{K}_K(\mathbf{x}_{ti}, \mathbf{x}_{ti'})\}$$
$$l_{t,ii'} = 2.\mathbf{1}\{y_{ti} = y_{ti'}\} - 1$$

4   *Predict* $\hat{l}_{t,ii'} = \boldsymbol{\beta}_t^{(r)\top}z_{t,ii'}$

5   **if** $(l_{t,ii'} \neq \hat{l}_{t,ii'})$ **then**

6   **for** $t' = 1\ldots T$ **do**

7 $\beta_{t'}^{(r+1)} = \beta_{t'}^{(r)} + \frac{1}{\mu}l_{t,ii'}\boldsymbol{\Omega}_{t,t'}z_{t,ii'}$

8   **end**

9   *Solve for* $\boldsymbol{\Omega}$,
     (4.14)$\qquad \min_{\boldsymbol{\Omega}\succeq\mathbf{0},tr(\boldsymbol{\Omega})\leq 1} tr(\boldsymbol{\Omega}^{-1}(\boldsymbol{B}^\top \boldsymbol{B}))$

10   **end**

11 end

12 *Set* $\mathcal{K}_t \leftarrow \sum_{k=1}^K \beta_{tk}^{(R)}\mathcal{K}_{tk}, \forall t \in [T]$;

13 *Solve for* $\boldsymbol{\alpha}_t, t \in [T]$
  (4.15)
$$\max_{0\leq\boldsymbol{\alpha}_t\leq C,\boldsymbol{\alpha}_t^\top \boldsymbol{y}_t=0}\left\{\mathbf{1}^\top\alpha_t - \frac{1}{2}\alpha_t^\top \mathbf{Y}_t\mathcal{K}_t\mathbf{Y}_t\alpha_t\right\} \;(SVM)$$

For a given set of $T*K$ base kernels $\{\mathcal{K}_{tk}\}_{1\leq t\leq T}^{1\leq k\leq K}$ (K base kernels per task), we define a binary classification framework over a new instance space (so called $\mathcal{K}$-space) defined as follows:
(4.16)
$$z_{t,ii'} = \{\mathcal{K}_1(\mathbf{x}_{ti}, \mathbf{x}_{ti'}), \mathcal{K}_2(\mathbf{x}_{ti}, \mathbf{x}_{ti'}), \ldots, \mathcal{K}_K(\mathbf{x}_{ti}, \mathbf{x}_{ti'})\}$$
$$l_{t,ii'} = 2.\mathbf{1}\{y_{ti} = y_{ti'}\} - 1$$

Any hypothesis $h_t : \mathbb{R}^K \to \mathbb{R}$ for a task t induces a similarity function $\tilde{\mathcal{K}}_{h_t}(\mathbf{x}_{ti}, \mathbf{x}_{ti'})$ between instances $\mathbf{x}_{ti}$ and $\mathbf{x}_{ti'}$ in the original space:

$$\tilde{\mathcal{K}}_{h_t}(\mathbf{x}_{ti}, \mathbf{x}_{ti'}) = h_t(z_{t,ii'})$$
$$= h_t(\mathcal{K}_1(\mathbf{x}_{ti}, \mathbf{x}_{ti'}), \ldots, \mathcal{K}_K(\mathbf{x}_{ti}, \mathbf{x}_{ti'}))$$

Suppose we consider a linear function for our task hypothesis $h_t(z_{t,ii'}) = \boldsymbol{\beta}_t.z_{t,ii'}$ with the non-negative constraints $\boldsymbol{\beta}_t \geq 0$, then the resulting induced kernel $\tilde{\mathcal{K}}_{h_t}$ is also positive semi-definite. The key idea behind this two-stage approach is that if a $\mathcal{K}$-classifiers h_t is a good classifier in the $\mathcal{K}$-space, then the induced kernel $\tilde{\mathcal{K}}_{h_t}(\mathbf{x}_{ti}, \mathbf{x}_{ti'})$ will likely be positive when $\mathbf{x}_{ti}$ and $\mathbf{x}_{ti'}$

belong to the same class and negative otherwise. Thus the problem of learning a good combination of base kernels can be framed as a problem of learning a good $\mathcal{K}$-classifier.

With this framework, the optimization problem for learning $\boldsymbol{\beta}_t$ for each task t can be formulated as follows:
(4.17)
$$\min_{\mathbf{B}\geq 0}\sum_{t=1}^T \ell(l_{t,ii'}, \langle\beta_t, z_{t,ii'}\rangle) + \frac{\mu}{2}\mathcal{R}(\mathbf{B})$$

$$\ell(l_{t,ii'}, \langle\beta_t, z_{t,ii'}\rangle) = \frac{1}{\binom{N_t}{2} + N_t}\sum_{1\leq i\leq i'\leq N_t}\left[1 - l_{t,ii'}\beta_t z_{t,ii'}\right]_+$$

where $[1 - s]_+ = \max\{0, 1 - s\}$ and $\mathcal{R}(\mathbf{B})$ is the regularization function on the kernel weights $\boldsymbol{B}$. Since

we are interested in learning task relationships using the task kernel weights $\boldsymbol{\beta}_t$, we can directly extend the above formulation to incorporate the regularization on $\boldsymbol{\beta}_t$ based on *MK-MTRL*.

(4.18)

$$\min_{\boldsymbol{\Omega}} \min_{\mathbf{B} \geq 0} \sum_{t=1}^{T} \ell(l_{t,ii'}, \langle \beta_t, z_{t,ii'} \rangle) + \frac{\mu}{2} tr(\mathbf{B}\boldsymbol{\Omega}^{-1}\mathbf{B}^{\top})$$
$$\boldsymbol{\Omega} \succeq 0,$$
$$tr(\boldsymbol{\Omega}) \leq 1$$

Since the above objective function depends on every pair of observations, we consider an online learning procedure for faster computation that learns the kernel weights and the task relationship matrix sequentially. Due to space limitations, we show the online version of our algorithms in the supplementary section. Note that with the above formulation, one can easily extend the existing approach to jointly learn both the feature and task relationship matrices using matrix normal penalty [28].

5 Algorithms

Algorithm 1 shows the pseudo-code for *MK-MTRL*. It outlines the update steps explained in Section 3. The algorithm alternates between learning the model parameters, kernel weights and task relationship matrix until it reaches the maximum number of iterations [2] or when there are minimal changes in the subsequent $\boldsymbol{B}$.

The two-stage, online learning of *MK-MTRL* is given in Algorithm 2. The online learning of $\boldsymbol{\beta}_t$ and $\boldsymbol{\Omega}$ is based on the recent work by Saha et. al., 2011 [20]. We set the maximum number of rounds to $100,000$. Since we construct the examples in kernel space on the fly, there is no need keep the base kernel matrices in memory. This significantly reduces the computational burden required in computing $\boldsymbol{B}$.

We use *libSVM* to solve the T individual SVMs (equation 4.10). All the base kernels are normalized to unit trace. Note that equation 4.12 requires computing Singular Value Decomposition (SVD) on $(\boldsymbol{B}^{\top}\boldsymbol{B})$. One may use an efficient decomposition algorithm such as the randomized SVD to speed up the learning process [16].

6 Experiments

We evaluate the performance of our proposed model on several benchmark datasets. We compare our proposed model with five state-of-the-art baselines in multitask learning and in multitask multiple kernel learning. All reported results in this section are averaged over 10

random runs of the training data. Unless otherwise specified, all model parameters are chosen via 5-fold cross validation. The best model and models with statistically comparable results are shown in bold.

6.1 Compared Models We compare the following models for our evaluation.

- Single-Task Learning (STL) learns the tasks independently. STL uses either *SVM* (in case of binary classification tasks) or Kernel Ridge regression (in case of regression tasks) to learn the individual models.

- Multi-task Feature Learning (MTFL [1]) learns a shared feature representation from all the tasks using regularization. It learns this shared feature representation along with the task model parameters alternatively[3].

- Multi-task Relationship Learning (MTRL [29]) learns task relationship matrix under a regularization framework. This model can be viewed as a multitask generalization for single-task learning. It learns the task relationship matrix and the task parameters in an iterative fashion[4].

- Single-task Multiple Kernel Learning (IMKL) learns independent MKL for each task. This baseline does not use any shared information between the tasks. We use ℓ_p-MKL for each task. We tune the value of p from $[2, 3, 4, 6, 8.67]$ using 5-fold cross validation.

- Multi-task Multiple Kernel Feature Learning (MK-MTFL [10]) learns a shared kernel for feature representation from all tasks. This is a multiple kernel generalization of multitask feature learning problem. Again, we tune the value of $\tilde{p}$ from $[2, 3, 4, 6, 8.67]$ using 5-fold cross validation[5].

Unless otherwise specified, the kernels for *STL*, *MTFL* and *MTRL* are chosen (via cross validation) from either a Gaussian RBF kernel with different bandwidth or a linear kernel for each dataset. The value for C is chosen from $[10^{-3}, \ldots, 10^3]$. We tune the value of μ from $[10^{-7}, \ldots, 10^3]$. We use *Newton's* method to learn the task kernel weight matrix $\boldsymbol{B}$ for the alternating minimization algorithm. We compare our models on several

---

Table 1: Mean Squared Error (MSE) for each company ($\times 1000$)

	OLS	Lasso	MRCE	FES	STL	IKL	IMKL	MK-MTFL	MK-MTRL
Walmart	0.98	0.42	0.41	0.40	0.44	0.43	0.45	0.44	0.44
Exxon	0.39	0.31	0.31	0.29	0.34	0.32	0.33	0.32	0.32
GM	1.68	0.71	0.71	0.62	0.82	0.62	0.60	0.61	**0.56**
Ford	2.15	0.77	0.77	0.69	0.91	0.56	0.53	0.55	**0.49**
GE	0.58	0.45	0.45	0.41	0.43	0.41	0.40	0.40	**0.39**
ConocoPhillips	0.98	0.79	0.79	0.79	0.84	0.81	0.83	0.80	0.80
Citigroup	0.65	0.66	0.62	0.59	0.64	0.66	0.62	0.62	0.60
IBM	0.62	0.49	0.49	0.51	0.48	0.47	0.45	0.45	**0.43**
AIG	1.93	1.88	1.88	1.74	1.91	1.94	1.88	1.89	1.83
AVG	1.11	0.72	0.71	0.67	0.76	0.69	0.68	0.68	**0.65**

applications: Asset Return Prediction, Landmine Detection and Object Recognition [6]. It is worth noting that different applications require different types of base kernels and there is no common set of kernel functions that will work for all applications. We choose these base kernels based on the application and the type of data.

6.2 Asset Return Prediction We begin our experiments with asset return prediction data used in [19] [7]. It consists of weekly log returns of 9 stocks from the year 2004. This dataset is considered in linear multivariate regression with output covariance estimation techniques [19]. We consider first-order vector autoregressive models of the form $\boldsymbol{x}_t = f(\boldsymbol{x}_{t-1})$ where $\boldsymbol{x}_t$ corresponds to the 9-dimensional vector of weekly log-returns from 9 companies as shown in table 1. The dataset is split evenly such that the first 26 weeks of the year is used as the training set and the next 26 weeks is used as the test set. Following [21], we use univariate Gaussian kernels with 13 varying bandwidth, generated from each feature, as base kernels. The total number of base kernels sums to 117.

Performance is measured by the average mean-squared prediction error over the test set for each task. The experimental setup for this dataset follows exactly [19]. We compare the results from our proposed and baseline model with the results from Ordinary Least Square (*OLS*), Lasso, Multivariate Regression with Covariate Estimation (*MRCE*) and Factor Estimation and Selection (*FES*) models reported in [19] (See [19] for more details about the models). In addition to the standard baselines, we include Input Kernel Learning (*IKL*), which learns a vector of kernel weights $\boldsymbol{\beta}$ shared

by all tasks [24].

After running *MK-MTRL* on these 117 base kernels, the model sets most of them to 0 except for base kernels corresponding to bandwidths $(1e - 4, 1)$. These bandwidth selections represent the long-term and short-term dependencies common in temporal data. We reran the model with these selected non-zero bandwidths and report the results for these selected base kernels. We can see that the proposed model *MK-MTRL* performs better than all the baselines.

6.3 Landmine Detection This dataset [8] consists of 19 tasks collected from different landmine fields. Each task is a binary classification problem: landmines $(+)$ or clutter $(-)$ and each example consists of 9 features extracted from radar images with four moment-based features, three correlation-based features, one energy ratio feature and a spatial variance feature. Landmine data is collected from two different terrains: tasks $1 - 10$ are from highly foliated regions and tasks $11 - 19$ are from desert regions, therefore tasks naturally form two clusters. Any hypothesis learned from a task should be able to utilize the information available from other tasks belonging to the same cluster.

We choose $\{30, 50, 80\}$ examples per task for this dataset. We use a polynomial kernel with power $\{1, 2, 3, 4, 5\}$ for generating our base kernels. Note that we intentionally kept the size of the training data small to drive the need for learning from other tasks, which diminishes as the training sets per task become large. Due to class-imbalance issue (with few $(+)$ examples compared to $(-)$ examples), we use average Area Under the ROC Curve (*AUC*) as the performance measure. This dataset has been previously used for

---

[6] See supplementary material for additional experiments

[7] http://cran.r-project.org/web/packages/MRCE/index.html

[8] http://www.ee.duke.edu/~lcarin/LandmineData.zip

Table 2: Average AUC scores for different samples of *landmine* dataset. The table reports the mean and standard errors over 10 random runs.

	30 samples	50 Samples	80 Samples
STL	0.6315 ± 0.032	0.6540 ± 0.026	0.6542 ± 0.027
MTFL	0.6387 ± 0.037	0.6968 ± 0.015	0.7051 ± 0.020
MTRL	0.6555 ± 0.034	0.6933 ± 0.023	0.7074 ± 0.024
IMKL	0.6857 ± 0.024	0.7138 ± 0.011	0.7278 ± 0.011
MK-MTFL	$\mathbf{0.6866 \pm 0.018}$	0.7145 ± 0.009	0.7305 ± 0.009
MK-MTRL	$\mathbf{0.6870 \pm 0.033}$	$\mathbf{0.7242 \pm 0.011}$	$\mathbf{0.7405 \pm 0.014}$

jointly learning feature correlation and task correlation [28]. Hence, *landmine* dataset is an ideal dataset for evaluating all the models.

Table 2 reports the results from the experiment. We can see that *MK-MTRL* performs better in almost all cases. When the number of training examples is small, *MK-MTRL* has difficulty in learning the task relationship matrix $\mathbf{\Omega}$ as it depends on the kernel weights. On the other hand, *MK-MTFL* performs equally well as it shares the feature representation among the tasks which is especially useful when the number of training is relatively low. As we get more and more training data, *MK-MTRL* performs significantly better than all the other baselines.

6.4 Object Recognition In this section, we evaluate our two proposed algorithms for *MK-MTRL* with computer vision datasets, *Caltech101*[9] and *Oxford* flowers [10] in terms of accuracy and training time. *Caltech101* dataset consists of $9,144$ images from 102 categories of objects such as faces, watches, animals, etc. The minimum, average and maximum number of images per category are 31,90 and 800 respectively. The *Caltech101* base kernels for each task are generated from feature descriptors such as geometric blur, PHOW gray/color, self-similarity, etc. For each of the 102 classes, we select 30 examples (for a total of 3060 examples per task) and then split these 30 examples into testing and training folds, which ensures matching training and testing distributions. *Oxford* flowers consists of 17 varieties of flowers and the number of images per category is 80. The *Oxford* base kernels for each task are generated from a subset of feature values. Each one-vs-all binary classification problem is considered as a single task, which amount to 102 and 17 tasks with 38 and 7-base kernels per task, respectively. Following

the previous work, we set the value of $C = 1000$ for *Caltech101* dataset.

In addition to the baselines used before, we compare our algorithms with Multiple Kernel Learning by Stochastic Approximation (*MKL-SA*) [3]. *MKL-SA* has a similar formulation to that of (*MK-MTFL*), except that it sets $\lambda_{tk} = \lambda_t, \forall k$ in equation 2.3. At each time step, it samples one task, according to the multinomial distribution $Multi(\lambda_1, \lambda_2, \ldots, \lambda_T)$, to update it's model parameter, making it suitable for multitask learning with large number of tasks.

The results for *Caltech101* and *Oxford* are shown in Figure 1. The left plots show how the mean accuracy varies with respect to different training set sizes. The right plots show the average training time taken by each model with varying training set sizes. From the plots, we can see that *MK-MTRL* outperforms all the other state-of-the art baselines in both *Caltech101* and *Oxford* datasets. But one may notice that the run-time of *MK-MTFL* and *MK-MTRL* grows steeply in the number of samples per class. Similar results are observed when we increase the number of tasks or number of base kernels per task.

Since both *MK-MTFL* and *MK-MTRL* require the base kernels in memory to learn the kernel weights and the task relationship matrix iteratively, this poses a serious computational burden and explains our need for efficient learning algorithm for multitask multiple kernel learning problems. We report *MK-MTRL* with two-stage, online procedure as one of the baselines. On both *Caltech101* and *Oxford*, the two-stage procedure yields comparable performance to that of *MK-MTRL*.

The run-time complexity of two-stage, online *MK-MTRL* learning is significantly better than almost all the baselines. Since AVG takes the average of the task-specific base kernels, it has the lowest computational time. It is interesting to see that two-stage, online *MK-MTRL* performs better than *MKL-SA* both in terms of

Figure 1: **Top**: Mean accuracy (left) and runtime (right) calculated for *Caltech101* dataset with varying training set sizes. **Bottom**: Mean accuracy (left) and runtime (right) calculated for *Oxford* dataset with varying training set sizes.

accuracy and running time. We believe that since *MKL-SA* updates the kernel weights after learning a single model parameter, it takes more iterations to converge (in term of model parameters and the kernel weights).

6.5 Robot Inverse Dynamics We consider the problem of learning the inverse dynamics of a 7-DOF *SARCOS* anthropomorphic data [11]. The dataset consists of 28 dimensions, of which first 21 dimensions are considered as features and the last 7 dimensions are used as outputs. We add an additional feature to account for the bias. There are 7 regression tasks and use kernel ridge regression to learn the task parameters and kernel weights. The feature set includes seven joint positions, seven joint velocities and seven joint accelerations, which is used to predict seven joint torques for the seven degrees of freedom (DOF). We randomly sample 2000 examples, of which $\{15, 50, 100, 150, 200, 600\}$ are used

for training sets and the rest of the examples are used for test set.

This dataset has been previous shown to include positive correlation, negative correlation and task unrelatedness and will be a challenging problem for baselines that doesn't learn the task correlation.

Following [29], we use normalized Mean Squared error (nMSE), which is the mean squared error divided by the variance of the ground truth. We generate 31 base kernels from multivariate Gaussian kernels with 10 varying bandwidth (based on the range of the data) and feature-wise linear kernel on each of the 21 dimensions. We use linear kernel for single task learning. The results calculated for different training set size is reported in Figure 2. We can see that *MK-MTRL* performs better than all the baselines. Contrary to the results report in [10], *MK-MTFL* performs the worst. As the model sees more data, it struggles to learn the task relationship and even performs worse than the single task learning.

Moreover, we report the individual nMSE for each

Table 3: Comparison for multiple kernel models using nMSE on SARCOS data

	STL	IMKL	MK-MTRL
1st DOF	0.0862	0.0838	**0.0717**
	± 0.0033	± 0.0032	**± 0.0075**
2nd DOF	0.0996	0.0945	**0.0686**
	± 0.0041	± 0.0045	**± 0.0070**
3rd DOF	0.0918	0.0871	**0.0649**
	± 0.0042	± 0.0040	**± 0.0071**
4th DOF	0.0581	0.0514	**0.0298**
	± 0.0021	± 0.0020	**± 0.0037**
5th DOF	0.1513	0.1405	**0.1070**
	± 0.0063	± 0.0057	**± 0.0053**
6th DOF	0.2911	0.2822	**0.1835**
	± 0.0094	± 0.0081	**± 0.0125**
7th DOF	0.0715	0.0628	**0.0457**
	± 0.0025	± 0.0024	**± 0.0036**
AVG	0.1214	0.1146	**0.0816**
	± 0.0015	± 0.0013	**± 0.0028**

Figure 2: nMSE vs Number of training example for SARCOS data

Table 4: Experiment on *school* dataset

	Explained Variance
STL	0.1883 ± 0.020
IMKL	0.1975 ± 0.017
MK-MTFL	0.2024 ± 0.016
MK-MTRL	**0.2134 ± 0.016**

DOF in Table 3. It shows that *MK-MTRL* consistently outperforms in all the tasks. Comparing the results to the one reported in [29], we can see that *MT-MTRL* (with 0.0816 AVG nMSE score) performs better than *MTFL* and *MTRL* (with 0.3149 and 0.0912 AVG nMSE scores respectively).

6.6 Exam Score Prediction For completeness, we include the results for benchmark dataset in multitask regression [12]. The *school* dataset consists of examination scores of 15362 students from 139 schools in London. Each school is considered as a task and the feature set includes the year of the examination, four school-specific and three student-specific attribute. We replace each categorical attribute with one binary variable for each possible attribute value, as in [1]. This results in 26 attributes with additional attribute to account for the bias term. We generate univariate Gaussian kernel with 13 varying bandwidths from each of the 26 attributes as our base kernels. Training and test set are obtained by dividing examples of each task

into 60%-40%. We use explained variance as in [1], which is defined as one minus nMSE. We can see that *MK-MTRL* is better than both *IMKL* and *MK-MTFL*.

7 Conclusion

We proposed a novel multiple kernel multitask learning algorithm that uses inter-task relationships to efficiently learn the kernel weights. The key idea is based on the assumption that the related tasks will have similar weights for the task-specific base kernels. We proposed an iterative algorithm to jointly learn this task relationship matrix, kernel weights and the task model parameters. For large-scale datasets, we introduced a novel two-stage online learning algorithm to learn kernel weights efficiently. The effectiveness of our algorithm is empirically verified over several benchmark datasets. The results showed that both multiple kernel learning and task relationship learning for multitask problems significantly helps in boosting the performance of the model.

---

[12] http://ttic.uchicago.edu/~argyriou/code/mtl_feat/school_splits.tar

Acknowledgements

We thank the anonymous reviewers for their helpful comments.

References

[1] Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. Convex multi-task feature learning. *Machine Learning*, 73(3):243–272, 2008.

[2] Francis R Bach, Gert RG Lanckriet, and Michael I Jordan. Multiple kernel learning, conic duality, and the smo algorithm. In *Proceedings of the twenty-first international conference on Machine learning*, page 6. ACM, 2004.

[3] Serhat Bucak, Rong Jin, and Anil K Jain. Multi-label multiple kernel learning by stochastic approximation: Application to visual object recognition. In *Advances in Neural Information Processing Systems*, pages 325–333, 2010.

[4] Corinna Cortes, Mehryar Mohri, and Afshin Rostamizadeh. Generalization bounds for learning kernels. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 247–254, 2010.

[5] Corinna Cortes, Mehryar Mohri, and Afshin Rostamizadeh. Two-stage learning kernel algorithms. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 239–246, 2010.

[6] N Cristianini. On kernel-target alignment. *Advances in Neural Information Processing Systems*, 2002.

[7] Jingrui He and Rick Lawrence. A graph-based framework for multi-task multi-view learning. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 25–32, 2011.

[8] Chris Hinrichs, Vikas Singh, Jiming Peng, and Sterling Johnson. Q-mkl: Matrix-induced regularization in multi-kernel learning with applications to neuroimaging. In *Advances in neural information processing systems*, pages 1421–1429, 2012.

[9] Ali Jalali, Sujay Sanghavi, Chao Ruan, and Pradeep K Ravikumar. A dirty model for multi-task learning. In *Advances in Neural Information Processing Systems*, pages 964–972, 2010.

[10] Pratik Jawanpuria and J Saketha Nath. Multi-task multiple kernel learning. In *Society for Industrial and Applied Mathematics. Proceedings of the SIAM International Conference on Data Mining*, page 828. Society for Industrial and Applied Mathematics, 2011.

[11] Marius Kloft, Ulf Brefeld, Pavel Laskov, Klaus-Robert Müller, Alexander Zien, and Sören Sonnenburg. Efficient and accurate lp-norm multiple kernel learning. In *Advances in neural information processing systems*, pages 997–1005, 2009.

[12] Marius Kloft, Ulf Brefeld, Sören Sonnenburg, and Alexander Zien. Lp-norm multiple kernel learning. *The Journal of Machine Learning Research*, 12:953–997, 2011.

[13] Meghana Kshirsagar, Jaime Carbonell, and Judith Klein-Seetharaman. Multitask learning for host–pathogen protein interactions. *Bioinformatics*, 29(13):i217–i226, 2013.

[14] Abhishek Kumar, Alexandru Niculescu-Mizil, Koray Kavukcuoglu, and Hal Daume III. A binary classification framework for two-stage multiple kernel learning. In *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, 2012.

[15] Gert RG Lanckriet, Nello Cristianini, Peter Bartlett, Laurent El Ghaoui, and Michael I Jordan. Learning the kernel matrix with semidefinite programming. *The Journal of Machine Learning Research*, 5:27–72, 2004.

[16] Edo Liberty, Franco Woolfe, Per-Gunnar Martinsson, Vladimir Rokhlin, and Mark Tygert. Randomized algorithms for the low-rank approximation of matrices. *Proceedings of the National Academy of Sciences*, 104(51):20167–20172, 2007.

[17] Jun Liu, Shuiwang Ji, and Jieping Ye. Multi-task feature learning via efficient l 2, 1-norm minimization. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*, pages 339–348. AUAI Press, 2009.

[18] Alain Rakotomamonjy, Francis Bach, Stéphane Canu, and Yves Grandvalet. Simplemkl. *Journal of Machine Learning Research*, 9:2491–2521, 2008.

[19] Adam J Rothman, Elizaveta Levina, and Ji Zhu. Sparse multivariate regression with covariance estimation. *Journal of Computational and Graphical Statistics*, 19(4):947–962, 2010.

[20] Avishek Saha, Piyush Rai, Suresh Venkatasubramanian, and Hal Daume. Online learning of multiple tasks and their relationships. In *International Conference on Artificial Intelligence and Statistics*, pages 643–651, 2011.

[21] Vikas Sindhwani, Minh Ha Quang, and Aurélie C Lozano. Scalable matrix-valued kernel learning for high-dimensional nonlinear multivariate regression and granger causality. In *Proceedings of the twenty-ninth conference on uncertainty in artificial intelligence*. AUAI Press, 2013.

[22] Zhaonan Sun, Nawanol Ampornpunt, Manik Varma, and Svn Vishwanathan. Multiple kernel learning and the smo algorithm. In *Advances in neural information processing systems*, pages 2361–2369, 2010.

[23] Grzegorz Swirszcz and Aurelie C Lozano. Multi-level lasso for sparse multi-task regression. In *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, pages 361–368, 2012.

[24] Lei Tang, Jianhui Chen, and Jieping Ye. On multiple kernel learning with multiple labels. In *IJCAI*, pages 1255–1260, 2009.

[25] Kilian Weinberger, Anirban Dasgupta, John Langford, Alex Smola, and Josh Attenberg. Feature hashing for large scale multitask learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 1113–1120. ACM, 2009.

[26] Ya Xue, Xuejun Liao, Lawrence Carin, and Balaji

Krishnapuram. Multi-task learning for classification with dirichlet process priors. *The Journal of Machine Learning Research*, 8:35–63, 2007.

[27] Jintao Zhang and Jun Huan. Inductive multi-task learning with multiple view data. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 543–551. ACM, 2012.

[28] Yi Zhang and Jeff G Schneider. Learning multiple tasks with a sparse matrix-normal penalty. In *Advances in Neural Information Processing Systems*, pages 2550–2558, 2010.

[29] Yu Zhang and Dit-Yan Yeung. A regularization approach to learning task relationships in multitask learning. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 8(3):12, 2014.