Conic Multi-task Classification

Cong Li[1], Michael Georgiopoulos[1],
and Georgios C. Anagnostopoulos[2]

[1] University of Central Florida, Department of Electrical Engineering and Computer Science, 4000 Central Florida Blvd, Orlando, Florida, 32816, USA
congli@eecs.ucf.edu, michaelg@ucf.edu
[2] Florida Institute of Technology, Department of Electrical and Computer Engineering, 150 W University Blvd, Melbourne, FL 32901, USA
georgio@fit.edu

Abstract. Traditionally, Multi-task Learning (MTL) models optimize the average of task-related objective functions, which is an intuitive approach and which we will be referring to as Average MTL. However, a more general framework, referred to as Conic MTL, can be formulated by considering conic combinations of the objective functions instead; in this framework, Average MTL arises as a special case, when all combination coefficients equal 1. Although the advantage of Conic MTL over Average MTL has been shown experimentally in previous works, no theoretical justification has been provided to date. In this paper, we derive a generalization bound for the Conic MTL method, and demonstrate that the tightest bound is not necessarily achieved, when all combination coefficients equal 1; hence, Average MTL may not always be the optimal choice, and it is important to consider Conic MTL. As a byproduct of the generalization bound, it also theoretically explains the good experimental results of previous relevant works. Finally, we propose a new Conic MTL model, whose conic combination coefficients minimize the generalization bound, instead of choosing them heuristically as has been done in previous methods. The rationale and advantage of our model is demonstrated and verified via a series of experiments by comparing with several other methods.

Keywords: Multi-task Learning, Kernel Methods, Generalization Bound, Support Vector Machines.

1 Introduction

Multi-Task Learning (MTL) has been an active research field for over a decade, since its inception in [1]. By training multiple tasks simultaneously with shared information, it is expected that the generalization performance of each task can be improved, compared to training each task separately. Previously, various MTL schemes have been considered, many of which model the t-th task by a linear function with weight $\boldsymbol{w}_t, t = 1, \cdots T$, and assume a certain, underlying relationship between tasks. For example, the authors in [2] assumed all $\boldsymbol{w}_t$'s to

T. Calders et al. (Eds.): ECML PKDD 2014, Part II, LNCS 8725, pp. 193–208, 2014.

be part of a cluster centered at $\bar{\boldsymbol{w}}$, the latter one being learned jointly with $\boldsymbol{w}_t$. This assumption was further extended to the case, where the weights $\boldsymbol{w}_t$'s can be grouped into different clusters instead of a single global cluster [3,4]. Furthermore, a widely held MTL assumption is that tasks share a common, potentially sparse, feature representation, as done in [5,6,7,8,9,10,11], to name a few. It is worth mentioning that many of these works allow features to be shared among only a subset of tasks, which are considered "similar" or "related" to each other, where the relevance between tasks is discovered during training. This approach reduces and, sometimes, completely avoids the effect of "negative transfer", *i.e.*, knowledge transferred between irrelevant tasks, which leads to degraded generalization performance. Several other recent works that focused on the discovery of task relatedness include [12,13,14,15]. Additionally, some kernel-based MTL models assume that the data from all tasks are pre-processed by a (partially) common feature mapping, thus (partially) sharing the same kernel function; see [16,17,18], again, to name a few.

Most of these previous MTL formulations consider the following classic setting: A set of training data $\{\boldsymbol{x}_t^i, y_t^i\} \in \mathcal{X} \times \mathcal{Y}, i = 1, \cdots, N_t$ is provided for the t-th task ($t = 1, \cdots, T$), where $\mathcal{X}$, $\mathcal{Y}$ are the input and output spaces correspondingly. Each datum from the t-th task is assumed to be drawn from an underlying probability distribution $P_t(X_t, Y_t)$, where X_t and Y_t are random variables in the input and output space respectively. Then, a MTL problem is formulated as follows

$$\min_{\boldsymbol{w} \in \Omega(\boldsymbol{w})} \sum_{t=1}^{T} f(\boldsymbol{w}_t, \boldsymbol{x}_t, \boldsymbol{y}_t) \tag{1}$$

where $\boldsymbol{w} \triangleq (\boldsymbol{w}_1, \cdots, \boldsymbol{w}_T)$ is the collection of all $\boldsymbol{w}_t$'s, and, similarly, $\boldsymbol{x}_t \triangleq (\boldsymbol{x}_t^1, \cdots, \boldsymbol{x}_t^{N_t})$, $\boldsymbol{y}_t \triangleq (y_t^1, \cdots, y_t^{N_t})$. f is a function common to all tasks. It is important to observe that, without the constraint $\boldsymbol{w} \in \Omega(\boldsymbol{w})$, Problem (1) degrades to T independent learning problems. Therefore, in most scenarios, the set $\Omega(\boldsymbol{w})$ is designed to capture the inter-task relationships. For example, in [16], the model combines MTL with Multiple Kernel Learning (MKL), which is formulated as follows

$$\begin{aligned} f(\boldsymbol{w}_t, \boldsymbol{x}_t, \boldsymbol{y}_t) &\triangleq \frac{1}{2}\|\boldsymbol{w}_t\|^2 + C \sum_{i=1}^{N_t} l(\boldsymbol{w}_t, \phi_t(\boldsymbol{x}_t^i), \boldsymbol{y}_t^i) \\ \Omega(\boldsymbol{w}) &\triangleq \{\boldsymbol{w} = (\boldsymbol{w}_1, \cdots, \boldsymbol{w}_T) : \boldsymbol{w}_t \in \mathcal{H}_{\boldsymbol{\theta},\boldsymbol{\gamma}_t}, \boldsymbol{\theta} \in \Omega(\boldsymbol{\theta}), \boldsymbol{\gamma} \in \Omega(\boldsymbol{\gamma})\} \end{aligned} \tag{2}$$

Here, l is a specified loss function, $\phi_t : \mathcal{X} \to \mathcal{H}_{\boldsymbol{\theta},\boldsymbol{\gamma}_t}$ is the feature mapping for the t-th task, $\mathcal{H}_{\boldsymbol{\theta},\boldsymbol{\gamma}_t}$ is the Reproducing Kernel Hilbert Space (RKHS) with reproducing kernel function $k_t \triangleq \sum_{m=1}^{M}(\theta_m + \gamma_t^m)k_m$, where $k_m : \mathcal{X} \times \mathcal{X} \to \mathbb{R}, m = 1, \cdots, M$ are pre-selected kernel functions. $\|\boldsymbol{w}_t\| \triangleq \sqrt{\langle \boldsymbol{w}_t, \boldsymbol{w}_t \rangle}$ is the norm defined in $\mathcal{H}_{\boldsymbol{\theta},\boldsymbol{\gamma}_t}$. Also, $\Omega(\boldsymbol{\theta})$ is the feasible set of $\boldsymbol{\theta} \triangleq (\theta_1, \cdots, \theta_M)$, and, similarly, $\Omega(\boldsymbol{\gamma})$ is the feasible set of $\boldsymbol{\gamma} \triangleq (\boldsymbol{\gamma}_1, \cdots, \boldsymbol{\gamma}_T)$. It is not hard to see that, in this setting, $\Omega(\boldsymbol{w})$ is designed such that all tasks partially share the same

kernel function in a MKL manner, parameterized by the common coefficient $\boldsymbol{\theta}$ and task-specific coefficient $\boldsymbol{\gamma}_t, t = 1, \cdots, T$.

Another example, Sparse MTL [17], has the following formulation:

$$f(\boldsymbol{w}_t, \boldsymbol{x}_t, \boldsymbol{y}_t) \triangleq \sum_{i=1}^{N_t} l(\boldsymbol{w}_t, \phi_t(\boldsymbol{x}_t^i), \boldsymbol{y}_t^i)$$
$$\Omega(\boldsymbol{w}) \triangleq \{\boldsymbol{w} = (\boldsymbol{w}_1, \cdots, \boldsymbol{w}_T) : \boldsymbol{w}_t \triangleq (\boldsymbol{w}_t^1, \cdots, \boldsymbol{w}_t^M), \sum_{m=1}^{M} (\sum_{t=1}^{T} \|\boldsymbol{w}_t^m\|^q)^{p/q} \leq R\} \tag{3}$$

where $\boldsymbol{w}_t^m \in \mathcal{H}_m, \forall m = 1, \cdots, M, t = 1, \cdots, T$, $\boldsymbol{w}_t \in \mathcal{H}_1 \times \cdots \times \mathcal{H}_M$, $0 < p \leq 1$, $1 \leq q \leq 2$. Note that although the original Sparse MTL is formulated as follows

$$\min_{\boldsymbol{w}} \sum_{m=1}^{M} (\sum_{t=1}^{T} \|\boldsymbol{w}_t^m\|^q)^{p/q} + C \sum_{t=1}^{T} \sum_{i=1}^{N_t} l(\boldsymbol{w}_t, \phi_t(\boldsymbol{x}_t^i), \boldsymbol{y}_t^i) \tag{4}$$

due to the first part of Proposition 12 in [19], which we restate as Proposition 1 below[1], it is obvious that, for any $C > 0$, there exists a $R > 0$, such that Problem (1) and Problem (4) are equivalent.

Proposition 1. *Let $\mathcal{D} \subseteq \mathcal{X}$, and let $f, g : \mathcal{D} \mapsto \mathbb{R}$ be two functions. For any $\sigma > 0$, there must exist a $\tau > 0$, such that the following two problems are equivalent*

$$\min_{x \in \mathcal{D}} f(x) + \sigma g(x) \tag{5}$$

$$\min_{x \in \mathcal{D}, g(x) \leq \tau} f(x) \tag{6}$$

The formulation given in Problem (1), which we refer to as *Average MTL*, is intuitively appealing: It is reasonable to expect the average generalization performance of the T tasks to be improved, by optimizing the average of the T objective functions. However, as argued in [20], solving Problem (1) yields only a particular solution on the Pareto Front of the following Multi-Objective Optimization (MOO) problem

$$\min_{\boldsymbol{w} \in \Omega(\boldsymbol{w})} \boldsymbol{f}(\boldsymbol{w}, \boldsymbol{x}, \boldsymbol{y}) \tag{7}$$

where $\boldsymbol{f}(\boldsymbol{w}, \boldsymbol{x}, \boldsymbol{y}) \triangleq [f(\boldsymbol{w}_1, \boldsymbol{x}_1, \boldsymbol{y}_1), \cdots, f(\boldsymbol{w}_T, \boldsymbol{x}_T, \boldsymbol{y}_T)]'$. This is true, because scalarizing a MOO problem by optimizing different conic combinations of the objective functions, leads to the discovery of solutions that correspond to points on the convex part of the problem's Pareto Front [21, p. 178]. In other words, by

[1] Note that the difference between Proposition 1 here and Proposition 12 in [19] is that, Proposition 1 does not require convexity of f, g and $\mathcal{D}$; these are requirements necessary for the second part of Proposition 12 in [19], which we do not utilize here.

conically scalarizing Problem (7) using different $\boldsymbol{\lambda} \triangleq [\lambda_1, \cdots, \lambda_T]'$, $\lambda_t > 0, \forall t = 1, \cdots, T$, the optimization problem

$$\min_{\boldsymbol{w} \in \Omega(\boldsymbol{w})} \sum_{t=1}^{T} \lambda_t f(\boldsymbol{w}_t, \boldsymbol{x}_t, \boldsymbol{y}_t) \tag{8}$$

yields different points on the Pareto Front of Problem (7). Therefore, there is little reason to believe that the solution of Problem (8) for the special case of $\lambda_t = 1, \forall t = 1, \cdots, T$, *i.e.*, the Average MTL's solution, is the best achievable. In fact, there might be other points on the Pareto Front that result in better generalization performance for each task, hence, yielding better average performance of the T tasks. Therefore, instead of solving Problem (1), one can accomplish this by optimizing Problem (8).

A previous work along these lines was performed in [20]. The authors considered the following MTL formulation, named *Pareto-Path MTL*

$$\min_{\boldsymbol{w} \in \Omega(\boldsymbol{w})} [\sum_{t=1}^{T} (f(\boldsymbol{w}_t, \boldsymbol{x}_t, \boldsymbol{y}_t))^p]^{1/p} \tag{9}$$

which, assuming all objective functions are positive, minimizes the L_p-norm of the objectives when $p \geq 1$, and the L_p-pseudo-norm when $0 < p < 1$. It was proven that, for any $p > 0$, Problem (9) is equivalent to Problem (8) with

$$\lambda_t = \begin{cases} \frac{f(\boldsymbol{w}_t, \boldsymbol{x}_t, \boldsymbol{y}_t)^{p-1}}{\sum_{t=1}^{T} (f(\boldsymbol{w}_t, \boldsymbol{x}_t, \boldsymbol{y}_t))^p} & \text{if } p > 1 \\ 1 & \text{if } p = 1 \\ \frac{\sum_{t=1}^{T} (f(\boldsymbol{w}_t, \boldsymbol{x}_t, \boldsymbol{y}_t))^{\frac{1-p}{p}}}{f(\boldsymbol{w}_t, \boldsymbol{x}_t, \boldsymbol{y}_t)^{1-p}} & \text{if } 0 < p < 1 \end{cases}, \forall t = 1, \cdots, T \tag{10}$$

Thus by varying $p > 0$, the solutions of Problem (9) trace a path on the Pareto Front of Problem (7). While Average MTL is equivalent to Problem (9), when $p = 1$, it was demonstrated that the experimental results are usually better when $p < 1$, compared to $p = 1$, in a Support Vector Machine (SVM)-based MKL setting. Regardless of the close correlation of the superior obtained results to our previous argument, the authors did not provide a rigorous basis of the advantage of considering an objective function other than the average of the T task objectives. Therefore, use of the L_p-(pseudo-)norm in the paper's objective function remains so far largely a heuristic element of their approach.

In light of the just-mentioned potential drawbacks of Average MTL and the lack of supporting theory in the case of Pareto-Path MTL, in this paper, we analytically justify why it is worth considering Problem (8), which we refer to as *Conic MTL*, and why it is advantageous. Specifically, a major contribution of this paper is the derivation of a generalization bound for Conic MTL, which illustrates that, indeed, the tightest bound is not necessarily achieved, when all λ_t's equal to 1. Therefore, it answers the previous question, and justifies the importance of considering Conic MTL. Also, as a byproduct of the generalization bound, in Section 2, we theoretically show the benefit of Pareto-Path MTL: the

generalization bound of Problem (9) is usually tighter when $p < 1$, compared to the case, when $p = 1$. Therefore, it explains Pareto-Path MTL's superiority over Average MTL.

Regarding Conic MTL, a natural question is how to choose the coefficients λ_t's. Instead of setting them heuristically, such as what Pareto-Path MTL does, we propose a new Conic MTL model that learns the λ_t's by minimizing the generalization bound. It ensures that our new model achieves the tightest generalization bound compared to any other settings of the λ_t values and, potentially, leads to superior performance. The new model is described in Section 3 and experimentally evaluated in Section 4. The experimental results verified our theoretical conclusions: Conic MTL can indeed outperform Average MTL and Pareto-Path MTL in many scenarios and, therefore, learning the coefficients λ_t's by minimizing the generalization bound is reasonable and advantageous. Finally, we summarize our work in Section 5.

In the sequel, we'll be using the following notational conventions: vector and matrices are denoted in boldface. Vectors are assumed to be columns vectors. If $\boldsymbol{v}$ is a vector, then $\boldsymbol{v}'$ denotes the transposition of $\boldsymbol{v}$. Vectors $\mathbf{0}$ and $\mathbf{1}$ are the all-zero and all-one vectors respectively. Also, $\succeq$, $\succ$, $\preceq$ and $\prec$ between vectors will stand for the component-wise $\geq$, $>$, $\leq$ and $<$ relations respectively. Similarly, for any $\boldsymbol{v}$, $\boldsymbol{v}^p$ represents the component-wise exponentiation of $\boldsymbol{v}$.

2 Generalization Bound

Similar to previous theoretical analyses of MTL methods [22,23,24,25,26,27], in this section, we derive the Rademacher complexity-based generalization bound for Conic MTL, *i.e.*, Problem (8). Specifically, we assume the following form of f and $\Omega(\boldsymbol{w})$ for classification problems:

$$\begin{aligned} f(\boldsymbol{w}_t, \boldsymbol{x}_t, \boldsymbol{y}_t) &\triangleq \frac{1}{2}\|\boldsymbol{w}_t\|^2 + C\sum_{i=1}^{N} l(y_t^i \langle \boldsymbol{w}_t, \phi(\boldsymbol{x}_t^i)\rangle) \\ \Omega(\boldsymbol{w}) &\triangleq \{\boldsymbol{w} = (\boldsymbol{w}_1, \cdots, \boldsymbol{w}_T) : \boldsymbol{w}_t \in \mathcal{H}_{\boldsymbol{\theta}}, \boldsymbol{\theta} \in \Omega(\boldsymbol{\theta})\} \end{aligned} \tag{11}$$

where l is the margin loss:

$$l(x) = \begin{cases} 0 & \text{if } \rho \leq x \\ 1 - x/\rho & \text{if } 0 \leq x \leq \rho \\ 1 & \text{if } x \leq 0 \end{cases} \tag{12}$$

$\phi : \mathcal{X} \to \mathcal{H}_{\boldsymbol{\theta}}$ is the common feature mapping for all tasks. $\mathcal{H}_{\boldsymbol{\theta}}$ is the RKHS defined by the kernel function $k \triangleq \sum_{m=1}^{M} \theta_m k_m$, where $k_m : \mathcal{X} \times \mathcal{X} \to \mathbb{R}, m = 1, \cdots, M$ are the pre-selected kernel functions. Furthermore, we assume the training data $\{\boldsymbol{x}_t^i, y_t^i\} \in \mathcal{X} \times \mathcal{Y}, t = 1, \cdots, T, i = 1, \cdots, N$ are drawn from the probability distribution $P_t(X_t, Y_t)$, where X_t and Y_t are random variables in the input and output space respectively. Note that, here, we assumed all tasks have equal number of training data and share a common kernel function.

These two assumptions were made to simplify notation and exposition, and they do not affect extending our results to a more general case, where an arbitrary number of training samples is available for each task and partially shared kernel functions are used; in the latter case, only relevant tasks may share the common kernel function, hence, reducing the effect of "negative transfer".

Substituting (11) into Problem (8) and based on Proposition 1, it is not hard to see that for any C in Eq. (11), there exist a $R > 0$ such that Problem (8) is equivalent to the following problem

$$\begin{aligned} \min_{\boldsymbol{w} \in \Omega(\boldsymbol{w})} & \sum_{t=1}^{T} \sum_{i=1}^{N_t} \lambda_t l(y_t^i \langle \boldsymbol{w}_t, \phi(\boldsymbol{x}_t^i) \rangle) \\ s.t. & \sum_{t=1}^{T} \lambda_t \|\boldsymbol{w}_t\|^2 \leq R \end{aligned} \tag{13}$$

Obviously, solving Problem (13) is the process of choosing the $\boldsymbol{w}$ in the hypothesis space $\mathcal{F}_{\boldsymbol{\lambda}}$, such that the empirical loss, *i.e.*, the objective function of Problem (13), is minimized. The relevant hypothesis space is defined below:

$$\mathcal{F}_{\boldsymbol{\lambda}} \triangleq \{\boldsymbol{w} = (\boldsymbol{w}_1, \cdots, \boldsymbol{w}_T) : \sum_{t=1}^{T} \lambda_t \|\boldsymbol{w}_t\|^2 \leq R, \boldsymbol{w}_t \in \mathcal{H}_{\boldsymbol{\theta}}, \boldsymbol{\theta} \in \Omega(\boldsymbol{\theta})\} \tag{14}$$

By defining the Conic MTL expected error $er(\boldsymbol{w})$ and empirical loss $\hat{er}_{\boldsymbol{\lambda}}(\boldsymbol{w})$ as follows

$$er(\boldsymbol{w}) = \frac{1}{T} \sum_{t=1}^{T} E[\mathbf{1}_{(-\infty,0]}(Y_t \langle \boldsymbol{w}_t, \phi(X_t) \rangle)] \tag{15}$$

$$\hat{er}_{\boldsymbol{\lambda}}(\boldsymbol{w}) = \frac{1}{TN} \sum_{t=1}^{T} \sum_{i=1}^{N} \lambda_t l(y_t^i \langle \boldsymbol{w}_t, \phi(\boldsymbol{x}_t^i) \rangle) \tag{16}$$

one of our major contribution is the following theorem, which gives the generalization bound of Problem (13) in the context of MKL-based Conic MTL for any $\lambda_t \in (1, r_\lambda), \forall t = 1, \cdots, T$, where r_λ is a pre-specified upper-bound for the λ_t's.

Theorem 1. *For fixed $\rho > 0$, $r_\lambda \in \mathbb{N}$ with $r_\lambda > 1$, and for any $\boldsymbol{\lambda} = [\lambda_1, \cdots, \lambda_T]'$, $\lambda_t \in (1, r_\lambda), \forall t = 1, \cdots, T$, $\boldsymbol{w} \in \mathcal{F}_{\boldsymbol{\lambda}}$, $0 < \delta < 1$, the following generalization bound holds with probability at least $1 - \delta$:*

$$er(\boldsymbol{w}) \leq \hat{er}_{\boldsymbol{\lambda}}(\boldsymbol{w}) + \frac{\sqrt{2} r_\lambda}{\rho} R(\mathcal{F}_{\boldsymbol{\lambda}}) + \sqrt{\frac{9}{TN} \ln \left(\frac{2 r_\lambda}{T} \sum_{t=1}^{T} \frac{1}{\lambda_t} \right)} + \sqrt{\frac{9 \ln \frac{1}{\delta}}{2TN}} \tag{17}$$

where $R(\mathcal{F}_{\boldsymbol{\lambda}})$ is the empirical Rademacher complexity of the hypothesis space $\mathcal{F}_{\boldsymbol{\lambda}}$, which is defined as

$$R(\mathcal{F}_{\boldsymbol{\lambda}}) \triangleq \frac{2}{TN} E[\sup_{\boldsymbol{w}\in\mathcal{F}_{\boldsymbol{\lambda}}} \sum_{t=1}^{T}\sum_{i=1}^{N} \sigma_t^i \langle \boldsymbol{w}_t, \phi(\boldsymbol{x}_t^i)\rangle] \tag{18}$$

and the σ_t^i's are i.i.d. Rademacher-distributed (i.e., Bernoulli(1/2)*-distributed random variables with sample space $\{-1, +1\}$).*

Based on Theorem 1, one is motivated to choose $\boldsymbol{\lambda}$ that minimizes the generalization bound, instead of heuristically selecting $\boldsymbol{\lambda}$ as in Eq. (10), which was suggested in [20]. Indeed, doing so does not guarantee obtaining the tightest generalization bound.

However, prior to proposing our new Conic MTL model that minimizes the generalization bound, it is still of interest to theoretically analyze why Pareto-Path MTL, *i.e.*, Problem (9), usually enjoys better generalization performance when $0 < p < 1$, rather than when $p = 1$, as described in Section 1. While the analysis is not given in [20], fortunately, we can provide some insights of the good performance of the model, when $0 < p < 1$, by utilizing Theorem 1 and with the help of the following two theorems.

Theorem 2. *For $\boldsymbol{\lambda} \succ \mathbf{0}$, the empirical Rademacher complexity $R(\mathcal{F}_{\boldsymbol{\lambda}})$ is monotonically decreasing with respect to each $\lambda_t, t = 1, \cdots, T$.*

Theorem 3. *Assume $f(\boldsymbol{w}_t, \boldsymbol{x}_t, \boldsymbol{y}_t) > 0, \forall t = 1, \cdots, T$. For $\boldsymbol{\lambda}$ that is defined in Eq. (10), when $0 < p < 1$, we have $\lambda_t > 1$ and λ_t is monotonically decreasing with respect to p, $\forall t = 1, \cdots, T$.*

Based on Eq. (10), if $f(\boldsymbol{w}_t, \boldsymbol{x}_t, \boldsymbol{y}_t) > 0, \forall t = 1, \cdots, T$, there must exist a fixed $r_\lambda > 0$, such that $\lambda_t \in (1, r_\lambda), \forall t = 1, \cdots, T$. Therefore we can analyze the generalization bound of Pareto-Path MTL based on Theorem 1, when $0 < p < 1$. Although Theorem 1 is not suitable for the case when $p = 1$, we can approximate its bound by letting p to be infinitely close to 1.

The above two theorems indicate that the empirical Rademacher complexity for the hypothesis space of Pareto-Path MTL monotonically increases with respect to p, when $0 < p < 1$. Therefore, the second term in the generalization bound decreases as p decreases. This is also true for the third term in the bound, based on Theorem 3. Thus, it is not a surprise that the generalization performance is usually better when $0 < p < 1$ than when $p = 1$, and it is reasonable to expect the performance to get improved when p decreases. In fact, such a monotonicity is reported in the experiments of [20]: the classification accuracy is usually monotonically increasing, when p decreases. It is worth mentioning that, although rarely observed, we may not have such monotonicity in performance, if the first term in the generalization bound, *i.e.*, the empirical loss, grows quickly as p decreases. However, the monotonic behavior of the generalization bound (except the empirical loss) is still sufficient for explaining the experimental results of Problem (9), which justifies the rationale of employing an arbitrarily

weighted conic combination of objective functions instead of using the average of these functions.

Finally, we provide two theorems that not only are used in the proof of Theorem 1, but also may be of interest on their own accord. Subsequently, in the next section, we describe our new MTL model.

Theorem 4. *Given* $\boldsymbol{\gamma} \triangleq [\gamma_1, \cdots, \gamma_T]'$ *with* $\boldsymbol{\gamma} \succ \mathbf{0}$*, define*

$$R(\mathcal{F}_{\boldsymbol{\lambda}}, \boldsymbol{\gamma}) = \frac{2}{TN} E[\sup_{\boldsymbol{w} \in \mathcal{F}_{\boldsymbol{\lambda}}} \sum_{t=1}^{T} \sum_{i=1}^{N} \gamma_t \sigma_t^i \langle \boldsymbol{w}_t, \phi(\boldsymbol{x}_t^i) \rangle] \tag{19}$$

For fixed $\boldsymbol{\lambda} \succ \mathbf{0}$*,* $R(\mathcal{F}_{\boldsymbol{\lambda}}, \boldsymbol{\gamma})$ *is monotonically increasing with respect to each* γ_t*.*

Theorem 5. *For fixed* $r_\lambda \geq 1$*,* $\rho > 0$*,* $\boldsymbol{\lambda} = [\lambda_1, \cdots, \lambda_T]', \lambda_t \in [1, r_\lambda], \forall t = 1, \cdots, T$*, and for any* $\boldsymbol{w} \in \mathcal{F}_{\boldsymbol{\lambda}}$*,* $0 < \delta < 1$*, the following generalization bound holds with probability at least* $1 - \delta$*:*

$$er(\boldsymbol{w}) \leq \hat{er}_{\boldsymbol{\lambda}}(\boldsymbol{w}) + \frac{r_\lambda}{\rho} R(\mathcal{F}_{\boldsymbol{\lambda}}) + \sqrt{\frac{9 \ln \frac{1}{\delta}}{2TN}} \tag{20}$$

Note that the difference between Theorem 5 and Theorem 1 is that, Theorem 1 is valid for *any* $\lambda_t \in (1, r_\lambda)$, while Theorem 5 is only valid for *fixed* $\lambda_t \in [1, r_\lambda]$. While the bound given in Theorem 1 is more general, it is looser due to the additional third term in (17) and due to the factor $\sqrt{2}$ multiplying the empirical Rademacher complexity.

3 A New MTL Model

In this section, we propose our new MTL model. Motivated by the generalization bound in Theorem 1, our model is formulated to select $\boldsymbol{w}$ and $\boldsymbol{\lambda}$ by minimizing the bound

$$\hat{er}_{\boldsymbol{\lambda}}(\boldsymbol{w}) + \frac{\sqrt{2} r_\lambda}{\rho} R(\mathcal{F}_{\boldsymbol{\lambda}}) + \sqrt{\frac{9}{TN} \ln \left(\frac{2 r_\lambda}{T} \sum_{t=1}^{T} \frac{1}{\lambda_t} \right)} + \sqrt{\frac{9 \ln \frac{1}{\delta}}{2TN}} \tag{21}$$

instead of choosing the coefficients $\boldsymbol{\lambda}$ heuristically, such as via Eq. (10) in [20]. Note that the bound's last term does not depend on any model parameters, while the third term has only a minor effect on the bound, when $\lambda_t \in (1, r_\lambda)$. Therefore, we omit these two terms, and propose the following model:

$$\begin{aligned} &\min_{\boldsymbol{w}, \boldsymbol{\lambda}} \hat{er}_{\boldsymbol{\lambda}}(\boldsymbol{w}) + \frac{\sqrt{2} r_\lambda}{\rho} R(\mathcal{F}_{\boldsymbol{\lambda}}) \\ &s.t.\ \boldsymbol{w} \in \mathcal{F}_{\boldsymbol{\lambda}}, \mathbf{1} \prec \boldsymbol{\lambda} \prec r_\lambda \mathbf{1}. \end{aligned} \tag{22}$$

Furthermore, due to the complicated nature of $R(\mathcal{F}_{\boldsymbol{\lambda}})$, it is difficult to optimize Problem (22) directly. Therefore, in the following theorem, we prove an upper

bound for $R(\mathcal{F}_{\boldsymbol{\lambda}})$, which yields a simpler expression. We remind the readers that the hypothesis space $\mathcal{F}_{\boldsymbol{\lambda}}$ is defined as

$$\mathcal{F}_{\boldsymbol{\lambda}} \triangleq \{\boldsymbol{w} = (\boldsymbol{w}_1, \cdots, \boldsymbol{w}_T) : \sum_{t=1}^{T} \lambda_t \|\boldsymbol{w}_t\|^2 \leq R, \boldsymbol{w}_t \in \mathcal{H}_{\boldsymbol{\theta}}, \boldsymbol{\theta} \in \Omega(\boldsymbol{\theta})\} \tag{23}$$

where $\mathcal{H}_{\boldsymbol{\theta}}$ is the RKHS defined by the kernel function $k \triangleq \sum_{m=1}^{M} \theta_m k_m$.

Theorem 6. *Given the hypothesis space $\mathcal{F}_{\boldsymbol{\lambda}}$, the empirical Rademacher complexity can be upper-bounded as follows:*

$$R(\mathcal{F}_{\boldsymbol{\lambda}}) \leq \frac{2}{TN} \sqrt{\sum_{t=1}^{T} \frac{1}{\lambda_t}}\, E\left[\sqrt{\sup_{\boldsymbol{w} \in \mathcal{F}_{\mathbf{1}}} \sum_{t=1}^{T} \left(\sum_{i=1}^{N} \sigma_t^i \langle \boldsymbol{w}_t, \phi(\boldsymbol{x}_t^i)\rangle\right)^2}\right] \tag{24}$$

where the feasible region of $\boldsymbol{w}$, i.e., $\mathcal{F}_{\mathbf{1}}$, is the same as $\mathcal{F}_{\boldsymbol{\lambda}}$ but with $\boldsymbol{\lambda} = \mathbf{1}$.

Note that, for a given $\Omega(\boldsymbol{\theta})$, the expectation term in (24) is a constant. If we define

$$s \triangleq E\left[\sqrt{\sup_{\boldsymbol{w} \in \mathcal{F}_{\mathbf{1}}} \sum_{t=1}^{T} \left(\sum_{i=1}^{N} \sigma_t^i \langle \boldsymbol{w}_t, \phi(\boldsymbol{x}_t^i)\rangle\right)^2}\right] \tag{25}$$

we arrive at our proposed MTL model:

$$\begin{aligned} \min_{\boldsymbol{w}, \boldsymbol{\lambda}} & \sum_{t=1}^{T} \sum_{i=1}^{N} \lambda_t l(y_t^i \langle \boldsymbol{w}_t, \phi(\boldsymbol{x}_t^i)\rangle) + \frac{2\sqrt{2} s r_\lambda}{\rho} \sqrt{\sum_{t=1}^{T} \frac{1}{\lambda_t}} \\ s.t.\ & \boldsymbol{w}_t \in \mathcal{H}_{\boldsymbol{\theta}}, \forall t = 1, \cdots, T \\ & \boldsymbol{\theta} \in \Omega(\boldsymbol{\theta}),\ \sum_{t=1}^{T} \lambda_t \|\boldsymbol{w}_t\|^2 \leq R, \mathbf{1} \prec \boldsymbol{\lambda} \prec r_\lambda \mathbf{1}. \end{aligned} \tag{26}$$

The next proposition provides an equivalent optimization problem, which is easier to solve.

Proposition 2. *For any fixed $C > 0$, $s > 0$ and $r_\lambda > 0$, there exist $R > 0$ and $a > 0$ such that Problem (26) and the following optimization problem are equivalent*

$$\begin{aligned} \min_{\boldsymbol{w}, \boldsymbol{\lambda}, \boldsymbol{\theta}} & \sum_{t=1}^{T} \lambda_t \Big(\sum_{m=1}^{M} \frac{\|\boldsymbol{w}_t^m\|^2}{2\theta_m} + C \sum_{i=1}^{N} \sum_{m=1}^{M} l(y_t^i \langle \boldsymbol{w}_t^m, \phi_m(\boldsymbol{x}_t^i)\rangle)\Big) \\ s.t.\ & \boldsymbol{w}_t^m \in \mathcal{H}_m, \forall t = 1, \cdots, T, m = 1, \cdots, M, \\ & \boldsymbol{\theta} \in \Omega(\boldsymbol{\theta}),\ \sum_{t=1}^{T} \frac{1}{\lambda_t} \leq a, \mathbf{1} \prec \boldsymbol{\lambda} \prec r_\lambda \mathbf{1}. \end{aligned} \tag{27}$$

where $\mathcal{H}_m$ is the RKHS defined by the kernel function k_m, and $\phi_m : \mathcal{X} \to \mathcal{H}_m$.

It is worth pointing out that, Problem (27) minimizes the generalization bound (21) for *any* $\Omega(\boldsymbol{\theta})$. A typical setting is to adapt the L_p-norm MKL method by letting $\Omega(\boldsymbol{\theta}) \triangleq \{\boldsymbol{\theta} = [\theta_1, \cdots, \theta_M]' : \boldsymbol{\theta} \succeq \mathbf{0}, \|\boldsymbol{\theta}\|_p \leq 1\}$, where $p \geq 1$. Alternatively, one may want to employ the *optimal neighborhood kernel* method [28] by letting $\Omega(\boldsymbol{\theta}) \triangleq \{\boldsymbol{\theta} = [\theta_1, \cdots, \theta_M]' : \sum_{t=1}^{T} \|\boldsymbol{K}_t - \hat{\boldsymbol{K}}_t\|_F \leq R_k, \boldsymbol{K}_t \triangleq \sum_{m=1}^{M} \theta_m \boldsymbol{K}_t^m\}$, where $\boldsymbol{K}_t^m \in \mathbb{R}^{N \times N}$ is the kernel matrix whose (i,j)-th element is calculated as $k_m(\boldsymbol{x}_t^i, \boldsymbol{x}_t^j)$, and $\hat{\boldsymbol{K}}_t$'s are the kernel matrices evaluated by a pre-defined kernel function on the training data of the t-th task.

By assuming $\Omega(\boldsymbol{\theta})$ to be a convex set and electing the loss function l to be convex in the model parameters (such as the hinge loss function), Problem (27) is jointly convex with respect to both $\boldsymbol{w}$ and $\boldsymbol{\theta}$. Also, it is separately convex with respect to $\boldsymbol{\lambda}$. Therefore, it is straightforward to employ a block-coordinate descent method to optimize Problem (27). Finally, it is worth mentioning that, by choosing to employ the hinge loss function, the generalization bound in Theorem 1 still holds, since the hinge loss upper-bounds the margin loss for $\rho = 1$. Therefore, our model still minimizes the generalization bound.

3.1 Incorporating L_p-Norm MKL

In this paper, we specifically consider endowing our MTL model with L_p-norm MKL, since it can be better analyzed theoretically, is usually easy to optimize and, often, yields good performance outcomes.

Although the upper bound in Theorem 6 is suitable for any $\Omega(\boldsymbol{\theta})$, it might be loose due to its generality. Another issue is that the expectation present in the bound is still hard to calculate. Therefore, as we consider L_p-norm MKL, it is of interest to derive a bound specifically for it, which is easier to calculate and is potentially tighter.

Theorem 7. *Let $\Omega(\boldsymbol{\theta}) \triangleq \{\boldsymbol{\theta} = [\theta_1, \cdots, \theta_M]' : \boldsymbol{\theta} \succeq \mathbf{0}, \|\boldsymbol{\theta}\|_p \leq 1\}$, $p \geq 1$, and $\boldsymbol{K}_t^m \in \mathbb{R}^{N \times N}, t = 1, \cdots, T, m = 1, \cdots, M$ be the kernel matrix, whose (i,j)-th element is defined as $k_m(\boldsymbol{x}_t^i, \boldsymbol{x}_t^j)$. Also, define $\boldsymbol{v}_t \triangleq [tr(\boldsymbol{K}_t^1), \cdots, tr(\boldsymbol{K}_t^M)]' \in \mathbb{R}^M$. Then, we have*

$$R(\mathcal{F}_{\boldsymbol{\lambda}}) \leq \frac{2\sqrt{2Rp^*}}{TN} \sqrt{\sum_{t=1}^{T} \frac{1}{\lambda_t} \|\boldsymbol{v}_t\|_{p^*}} \tag{28}$$

where $p^* \triangleq \frac{p}{p-1}$.

Following a similar procedure to formulating our general model Problem (27), we arrive at the following L_p-norm MKL-based MTL problem

$$\begin{aligned} \min_{\boldsymbol{w},\boldsymbol{\lambda},\boldsymbol{\theta}} & \sum_{t=1}^{T} \lambda_t \Big(\sum_{m=1}^{M} \frac{\|\boldsymbol{w}_t^m\|^2}{2\theta_m} + C \sum_{i=1}^{N} \sum_{m=1}^{M} l(y_t^i \langle \boldsymbol{w}_t^m, \phi(\boldsymbol{x}_t^i) \rangle) \Big) \\ s.t. \ & \boldsymbol{w}_t^m \in \mathcal{H}_m, \forall t = 1, \cdots, T, m = 1, \cdots, M, \\ & \boldsymbol{\theta} \succeq \mathbf{0}, \|\boldsymbol{\theta}\|_p \leq 1, \\ & \sum_{t=1}^{T} \frac{\|\boldsymbol{v}_t\|_{p^*}}{\lambda_t} \leq a, \mathbf{1} \prec \boldsymbol{\lambda} \prec r_\lambda \mathbf{1}. \end{aligned} \tag{29}$$

which, based on (21) and (28), minimizes the generalization bound. Note that, due to the bound that is specifically derived for L_p-norm MKL, the constraint $\sum_{t=1}^{T} \frac{1}{\lambda_t} \leq a$ in Problem (27) is changed to $\sum_{t=1}^{T} \frac{\|\boldsymbol{v}_t\|_{p^*}}{\lambda_t} \leq a$ in the previous problem. However, when all kernel matrices $\boldsymbol{K}_t^m$'s have the same trace (as is the case, when all kernel functions are normalized, such that $k_m(\boldsymbol{x}, \boldsymbol{x}) = 1, \forall m = 1, \cdots, M, \boldsymbol{x} \in \mathcal{X}$), for a given $p \geq 1$, $\|\boldsymbol{v}_t\|_{p^*}$ has the same value for all $t = 1, \cdots, T$. In this case, Problem (29) is equivalent to Problem (27).

4 Experiments

In this section, we conduct a series of experiments with several data sets, in order to show the merit of our proposed MTL model by comparing it to a few other related methods.

4.1 Experimental Settings

In our experiments, we specifically evaluate the L_p-norm MKL-based MTL model, *i.e.*, Problem (29), on classification problems using the hinge loss function. To solve Problem (29), we employed a block-coordinate descent algorithm, which optimizes each of the three variables $\boldsymbol{w}$, $\boldsymbol{\lambda}$ and $\boldsymbol{\theta}$ in succession by holding the remaining two variables fixed. Specifically, in each iteration, three optimization problems are solved. First, for fixed $\boldsymbol{\lambda}$ and $\boldsymbol{\theta}$, the optimization with respect to $\boldsymbol{w}$ can be split into T independent SVM problems, which are solved via `LIBSVM` [29]. Next, for fixed $\boldsymbol{w}$ and $\boldsymbol{\theta}$, the optimization with respect to $\boldsymbol{\lambda}$ is convex and is solved using `CVX` [30][31]. Finally, minimizing with respect to $\boldsymbol{\theta}$, while $\boldsymbol{w}$ and $\boldsymbol{\lambda}$ are held fixed, has a closed-form solution:

$$\boldsymbol{\theta}^* = \left(\frac{\boldsymbol{v}}{\|\boldsymbol{v}\|_{\frac{p}{p+1}}} \right)^{\frac{1}{p+1}} \tag{30}$$

where $\boldsymbol{v} \triangleq [v_1, \cdots, v_M]'$ and $v_m \triangleq \sum_{t=1}^{T} \|\boldsymbol{w}_t^m\|, \forall m = 1, \cdots, M$. Although more efficient algorithms may exist, we opted to use this simple and easy-to-implement algorithm, since the optimization strategy is not the focus of our paper[2].

[2] Our `MATLAB` implementation is located at `http://github.com/congliucf/ECML2014`

For all experiments, 11 kernels were selected for use: a Linear kernel, a 2^{nd}-order Polynomial kernel and Gaussian kernels with spread parameter values $\{2^{-7}, 2^{-5}, 2^{-3}, 2^{-1}, 2^0, 2^1, 2^3, 2^5, 2^7\}$. Parameters C, p and a were selected via cross-validation. Our model is evaluated on 6 data sets: 2 real-world data sets from the UCI repository [32], 2 handwritten digits data sets, and 2 multi-task data sets, which we detail below.

The Wall-Following Robot Navigation (*Robot*) and Vehicle Silhouettes (*Vehicle*) data sets were obtained from the UCI repository. The *Robot* data, consisting of 4 features per sample, describe the position of the robot, while it navigates through a room following the wall in a clockwise direction. Each sample is to be classified according to one of the following four classes: "Move-Forward", "Slight-Right-Turn", "Sharp-Right-Turn" and "Slight-Left-Turn". On the other hand, the *Vehicle* data set is a collection of 18-dimensional feature vectors extracted from images. Each datum should be classified into one of four classes: "4 Opel", "SAAB", "Bus" and "Van".

The two handwritten digit data sets, namely *MNIST*[3] and *USPS*[4], consist of grayscale images of handwritten digits from 0 to 9 with 784 and 256 features respectively. Each datum is labeled as one of ten classes, each of which represents a single digit. For these four multi-class data sets, an equal number of samples from each class were chosen for training. Also, we approached these multi-class problems as MTL problems using a one-vs.-one strategy and the averaged classification accuracy is calculated for each data set.

The last two data sets, namely *Letter*[5] and *Landmine*[6], correspond to pure multi-task problems. Specifically, the *Letter* data set involves 8 tasks: "C" vs. "E", "G" vs. "Y", "M" vs. "N", "A" vs. "G", "I" vs. "J", "A" vs. "O", "F" vs. "T" and "H" vs. "N". Each letter is represented by a 8×16 pixel image, which forms a 128-dimensional feature vector. The goal for this problem is to correctly recognize the letters in each task. On the other hand, the *Landmine* data set consists of 29 binary classification tasks. Each datum is a 9-dimensional feature vector extracted from radar images that capture a single region of landmine fields. The goal for each task is to detect landmines in specific regions. For the experiments involving these two data sets, we re-sampled the data such that, for each task, the two classes contain equal number of samples.

In all our experiments, we considered training set sizes of 10%, 20% and 50% of the original data set. As an exception, for the *Landmine* data set, we did not use the 10% of the original set for training due to its small size; instead, we used 20%, 30% and 50%.

We compared our method with five different Multi-Task MKL (MT-MKL) methods. The first one is Pareto-Path MTL, *i.e.*, Problem (9), which was originally proposed in [20]. One can expect our new method to outperform it in most cases, since our method selects $\boldsymbol{\lambda}$ by minimizing the generalization bound, while

[3] Available at: `http://yann.lecun.com/exdb/mnist/`
[4] Available at: `http://www.cs.nyu.edu/~roweis/data.html`
[5] Available at: `http://multitask.cs.berkeley.edu/`
[6] Available at: `http://people.ee.duke.edu/~lcarin/LandmineData.zip`

Pareto-Path MTL selects its value heuristically via Eq. (10). The second method we compared with is the L_p-norm MKL-based Average MTL, which is the same as our method for $\boldsymbol{\lambda} = \mathbf{1}$. As we argued earlier in the introduction, minimizing the averaged objective does not necessarily guarantee the best generalization performance. By comparing with Average MTL, we expect to verify our claim experimentally. Moreover, we compared with two other popular MT-MKL methods, namely Tang's Method [16] and Sparse MTL [17]. These two methods were outlined in Section 1. Finally, we considered the baseline approach, which trains each task individually via a traditional single-task L_p-norm MKL strategy.

4.2 Experimental Results

Table 1 provides the obtained experimental results based on the settings that were described in the previous sub-section. More specifically, in Table 1, we

Table 1. Comparison of Multi-task Classification Accuracy between Our Method and Five Other Methods. Averaged performances of 20 runs over randomly sampled training set are reported.

Robot	Our Method	Pareto	Average	Tang	Sparse	Baseline
10%	**95.83**	95.07	95.16	93.93	94.69	95.54
20%	**97.11**	96.11	95.90	96.36	96.56	95.75
50%	**98.41**	96.80	96.59	97.21	98.09	96.31
Vehicle	Our Method	Pareto	Average	Tang	Sparse	Baseline
10%	**80.10**	80.05	79.77	78.47	79.28	78.01
20%	84.69	**85.33**	85.22	83.98	84.44	84.37
50%	**89.90**	88.04	87.93	88.13	88.57	87.64
Letter	Our Method	Pareto	Average	Tang	Sparse	Baseline
10%	83.00	**83.95**	81.45	80.86	83.00	81.33
20%	87.13	**87.51**	86.42	82.95	87.09	86.39
50%	90.47	**90.61**	90.01	84.87	90.65	89.80
Landmine	Our Method	Pareto	Average	Tang	Sparse	Baseline
20%	**70.18**	69.59	67.24	66.60	58.89	66.64
30%	**74.52**	74.15	71.62	70.89	65.83	71.14
50%	**78.26**	77.42	76.96	76.08	75.82	76.29
MNIST	Our Method	Pareto	Average	Tang	Sparse	Baseline
10%	**93.59**	89.30	88.81	92.37	93.48	88.71
20%	**96.08**	95.02	94.95	95.94	95.96	94.81
50%	97.44	96.92	96.98	97.47	**97.53**	97.04
USPS	Our Method	Pareto	Average	Tang	Sparse	Baseline
10%	**94.61**	90.22	90.11	93.20	94.52	89.02
20%	**97.44**	96.26	96.25	97.37	97.53	96.17
50%	**98.98**	98.51	98.59	98.96	**98.98**	98.49

report the average classification accuracy of 20 runs over a randomly sampled training set. Moreover, the best performance among the 6 competing methods is highlighted in boldface. To test the statistical significance of the differences between our method and the 5 other methods, we employed a t-test to compare mean accuracies using a significance level of $\alpha = 0.05$. In the table, underlined numbers indicate the results that are statistically significantly worse than the ones produced by our method.

When analyzing the results in Table 1, first of all, we observe that the optimal result is almost always achieved by the two Conic MTL methods, namely our method and Pareto-Path MTL. This result not only shows the advantage of Conic MTL over Average MTL, but also demonstrates the benefit compared to other MTL methods, such as Tang's MTL and Sparse MTL. Secondly, it is obvious that our method can usually achieve better result than Pareto-Path MTL; as a matter of fact, in many cases the advantage is statistically significant. This observation validates the underlying rationale of our method, which chooses the coefficient $\boldsymbol{\lambda}$ by minimizing the generalization bound instead of using Eq. (10). Finally, when comparing our method against the five alternative methods, our results are statistically better most of the time, which further emphasizes the benefit of our method.

5 Conclusions

In this paper, we considered the MTL problem that minimizes the conic combination of objectives with coefficients $\boldsymbol{\lambda}$, which we refer to as Conic MTL. The traditional MTL method, which minimizes the average of the task objectives (Average MTL), is only a special case of Conic MTL with $\boldsymbol{\lambda} = \mathbf{1}$. Intuitively, such a specific choice of $\boldsymbol{\lambda}$ should not necessarily lead to optimal generalization performance.

This intuition motivated the derivation of a Rademacher complexity-based generalization bound for Conic MTL in a MKL-based classification setting. The properties of the bound, as we have shown in Section 2, indicate that the optimal choice of $\boldsymbol{\lambda}$ is indeed not necessarily equal to $\mathbf{1}$. Therefore, it is important to consider different values for $\boldsymbol{\lambda}$ for Conic MTL, which may yield tighter generalization bounds and, hence, better performance. As a byproduct, our analysis also explains the reported superiority of Pareto-Path MTL [20] over Average MTL.

Moreover, we proposed a new Conic MTL model, which aims to directly minimize the derived generalization bound. Via a series of experiments on six widely utilized data sets, our new model demonstrated a statistically significant advantage over Pareto-Path MTL, Average MTL, and two other popular MT-MKL methods.

Acknowledgments. Cong Li acknowledges support from National Science Foundation (NSF) grants No. 0806931 and No. 0963146. Furthermore, Michael Georgiopoulos acknowledges support from NSF grants No. 0963146, No. 1200566, and

No. 1161228. Also, Georgios C. Anagnostopoulos acknowledges partial support from NSF grant No. 1263011. Note that any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF. Finally, the authors would like to thank the three anonymous reviewers, that reviewed this manuscript, for their constructive comments.

References

1. Caruana, R.: Multitask learning. Machine Learning 28, 41–75 (1997)
2. Evgeniou, T., Pontil, M.: Regularized multi–task learning. In: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 109–117. ACM (2004)
3. Zhong, L.W., Kwok, J.T.: Convex multitask learning with flexible task clusters. In: Proceedings of the 29th International Conference on Machine Learning, ICML 2012 (2012)
4. Zhou, J., Chen, J., Ye, J.: Clustered multi-task learning via alternating structure optimization. In: Advances in Neural Information Processing Systems, pp. 702–710 (2011)
5. Obozinski, G., Taskar, B., Jordan, M.I.: Joint covariate selection and joint subspace selection for multiple classification problems. Statistics and Computing 20(2), 231–252 (2010)
6. Jalali, A., Sanghavi, S., Ruan, C., Ravikumar, P.K.: A dirty model for multi-task learning. In: Advances in Neural Information Processing Systems, pp. 964–972 (2010)
7. Gong, P., Ye, J., Zhang, C.: Multi-stage multi-task feature learning. The Journal of Machine Learning Research 14(1), 2979–3010 (2013)
8. Liu, J., Ji, S., Ye, J.: Multi-task feature learning via efficient l 2, 1-norm minimization. In: Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, pp. 339–348. AUAI Press (2009)
9. Fei, H., Huan, J.: Structured feature selection and task relationship inference for multi-task learning. In: 2011 IEEE 11th International Conference on Data Mining (ICDM), pp. 171–180 (2011)
10. Argyriou, A., Evgeniou, T., Pontil, M.: Convex multi-task feature learning. Machine Learning 73, 243–272 (2008)
11. Kang, Z., Grauman, K., Sha, F.: Learning with whom to share in multi-task feature learning. In: Proceedings of the 28th International Conference on Machine Learning, ICML 2011 (2011)
12. Zhang, Y., Yeung, D.Y.: A convex formulation for learning task relationships in multi-task learning. ArXiv e-prints (2012)
13. Zhang, Y.: Heterogeneous-neighborhood-based multi-task local learning algorithms. In: Advances in Neural Information Processing Systems, pp. 1896–1904 (2013)
14. Romera-Paredes, B., Argyriou, A., Berthouze, N., Pontil, M.: Exploiting unrelated tasks in multi-task learning. In: International Conference on Artificial Intelligence and Statistics, pp. 951–959 (2012)
15. Pu, J., Jiang, Y.G., Wang, J., Xue, X.: Multiple task learning using iteratively reweighted least square. In: Proceedings of the 23rd International Joint Conference on Artificial Intelligence, pp. 1607–1613 (2013)

16. Tang, L., Chen, J., Ye, J.: On multiple kernel learning with multiple labels. In: Proceedings of the 21st International Joint Conference on Artifical Intelligence, pp. 1255–1260 (2009)
17. Rakotomamonjy, A., Flamary, R., Gasso, G., Canu, S.: $l_p - l_q$ penalty for sparse linear and sparse multiple kernel multitask learning. IEEE Transactions on Neural Networks 22, 1307–1320 (2011)
18. Samek, W., Binder, A., Kawanabe, M.: Multi-task learning via non-sparse multiple kernel learning. In: Real, P., Diaz-Pernil, D., Molina-Abril, H., Berciano, A., Kropatsch, W. (eds.) CAIP 2011, Part I. LNCS, vol. 6854, pp. 335–342. Springer, Heidelberg (2011)
19. Kloft, M., Brefeld, U., Sonnenburg, S., Zien, A.: l_p-norm multiple kernel learning. Journal of Machine Learning Research 12, 953–997 (2011)
20. Li, C., Georgiopoulos, M., Anagnostopoulos, G.C.: Pareto-Path Multi-Task Multiple Kernel Learning. ArXiv e-prints (April 2014)
21. Boyd, S., Vandenberghe, L.: Convex Optimization. Cambridge University Press (2004)
22. Ando, R.K., Zhang, T.: A framework for learning predictive structures from multiple tasks and unlabeled data. Journal of Machine Learning Research 6 (2005)
23. Maurer, A.: The rademacher complexity of linear transformation classes. In: Lugosi, G., Simon, H.U. (eds.) COLT 2006. LNCS (LNAI), vol. 4005, pp. 65–78. Springer, Heidelberg (2006)
24. Maurer, A.: Bounds for linear multi-task learning. Journal of Machine Learning Research 7, 117–139 (2006)
25. Kakade, S.M., Shalev-Shwartz, S., Tewari, A.: Regularization techniques for learning with matrices. Journal of Machine Learning Research 13, 1865–1890 (2012)
26. Maurer, A., Pontil, M.: Structured sparsity and generalization. Journal of Machine Learning Research 13, 671–690 (2012)
27. Pontil, M., Maurer, A.: Excess risk bounds for multitask learning with trace norm regularization. In: Conference on Learning Theory, pp. 55–76 (2013)
28. Liu, J., Chen, J., Chen, S., Ye, J.: Learning the optimal neighborhood kernel for classification. In: Proceedings of the 21st International Joint Conference on Artifical Intelligence, pp. 1144–1149 (2009)
29. Chang, C.C., Lin, C.J.: LIBSVM: A library for support vector machines. ACM Transactions on Intelligent Systems and Technology 2, 27:1–27:27 (2011), Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`
30. Grant, M.C., Boyd, S.P.: Graph implementations for nonsmooth convex programs. In: Blondel, V.D., Boyd, S.P., Kimura, H. (eds.) Recent Advances in Learning and Control. LNCIS, vol. 371, pp. 95–110. Springer, Heidelberg (2008), `http://stanford.edu/~boyd/graph_dcp.html`
31. Grant, M., Boyd, S.: CVX: Matlab software for disciplined convex programming, version 1.21 (April 2011)
32. Frank, A., Asuncion, A.: UCI machine learning repository (2010)