Efficient Output Kernel Learning for Multiple Tasks

Pratik Jawanpuria[1] **, Maksim Lapin**[2] **, Matthias Hein**[1] **and Bernt Schiele**[2]
[1] Saarland University, Saarbrücken, Germany
[2] Max Planck Institute for Informatics, Saarbrücken, Germany

Abstract

The paradigm of multi-task learning is that one can achieve better generalization by learning tasks jointly and thus exploiting the similarity between the tasks rather than learning them independently of each other. While previously the relationship between tasks had to be user-defined in the form of an output kernel, recent approaches jointly learn the tasks and the output kernel. As the output kernel is a positive semidefinite matrix, the resulting optimization problems are not scalable in the number of tasks as an eigendecomposition is required in each step. Using the theory of positive semidefinite kernels we show in this paper that for a certain class of regularizers on the output kernel, the constraint of being positive semidefinite can be dropped as it is automatically satisfied for the relaxed problem. This leads to an unconstrained dual problem which can be solved efficiently. Experiments on several multi-task and multi-class data sets illustrate the efficacy of our approach in terms of computational efficiency as well as generalization performance.

1 Introduction

Multi-task learning (MTL) advocates sharing relevant information among several related tasks during the training stage. The advantage of MTL over learning tasks independently has been shown theoretically as well as empirically [1, 2, 3, 4, 5, 6, 7].

The focus of this paper is the question how the task relationships can be inferred from the data. It has been noted that naively grouping all the tasks together may be detrimental [8, 9, 10, 11]. In particular, outlier tasks may lead to worse performance. Hence, clustered multi-task learning algorithms [10, 12] aim to learn groups of closely related tasks. The information is then shared only within these clusters of tasks. This corresponds to learning the task covariance matrix, which we denote as the output kernel in this paper. Most of these approaches lead to non-convex problems.

In this work, we focus on the problem of directly learning the output kernel in the multi-task learning framework. The multi-task kernel on input and output is assumed to be decoupled as the product of a scalar kernel and the output kernel, which is a positive semidefinite matrix [1, 13, 14, 15]. In classical multi-task learning algorithms [1, 16], the degree of relatedness between distinct tasks is set to a constant and is optimized as a hyperparameter. However, constant similarity between tasks is a strong assumption and is unlikely to hold in practice. Thus recent approaches have tackled the problem of directly learning the output kernel. [17] solves a multi-task formulation in the framework of vector-valued reproducing kernel Hilbert spaces involving squared loss where they penalize the Frobenius norm of the output kernel as a regularizer. They formulate an invex optimization problem that they solve optimally. In comparison, [18] recently proposed an efficient barrier method to optimize a generic convex output kernel learning formulation. On the other hand, [9] proposes a convex formulation to learn low rank output kernel matrix by enforcing a trace constraint. The above approaches [9, 17, 18] solve the resulting optimization problem via alternate minimization between task parameters and the output kernel. Each step of the alternate minimization requires an eigenvalue decomposition of a matrix having as size the number of tasks and a problem corresponding to learning all tasks independently.

In this paper we study a similar formulation as [17]. However, we allow arbitrary convex loss functions and employ general p-norms for $p \in (1, 2]$ (including the Frobenius norm) as regularizer for the output kernel. Our problem is jointly convex over the task parameters and the output kernel. Small p leads to sparse output kernels which allows for an easier interpretation of the learned task relationships in the output kernel. Under certain conditions on p we show that one can drop the constraint that the output kernel should be positive definite as it is automatically satisfied for the unconstrained problem. This significantly simplifies the optimization and our result could also be of interest in other areas where one optimizes over the cone of positive definite matrices. The resulting unconstrained dual problem is amenable to efficient optimization methods such as stochastic dual coordinate ascent [19], which scale well to large data sets. Overall we do not require any eigenvalue decomposition operation at any stage of our algorithm and no alternate minimization is necessary, leading to a highly efficient methodology. Furthermore, we show that this trick not only applies to p-norms but also applies to a large class of regularizers for which we provide a characterization.

Our contributions are as follows: (a) we propose a generic p-norm regularized output kernel matrix learning formulation, which can be extended to a large class of regularizers; (b) we show that the constraint on the output kernel to be positive definite can be dropped as it is automatically satisfied, leading to an unconstrained dual problem; (c) we propose an efficient stochastic dual coordinate ascent based method for solving the dual formulation; (d) we empirically demonstrate the superiority of our approach in terms of generalization performance as well as significant reduction in training time compared to other methods learning the output kernel.

The paper is organized as follows. We introduce our formulation in Section 2. Our main technical result is discussed in Section 3. The proposed optimization algorithm is described in Section 4. In Section 5, we report the empirical results.

2 The Output Kernel Learning Formulation

We first introduce the setting considered in this paper. We denote the number of tasks by T. We assume that all tasks have a common input space $\mathcal{X}$ and a common positive definite kernel function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. We denote by $\psi(\cdot)$ the feature map and by H_k the reproducing kernel Hilbert space (RKHS) [20] associated with k. The training data is $(x_i, y_i, t_i)_{i=1}^n$, where $x_i \in \mathcal{X}$, t_i is the task the i-th instance belongs to and y_i is the corresponding label. Moreover, we have a positive definite matrix $\Theta \in S_+^T$ on the set of tasks $\{1, \ldots, T\}$, where S_+^T is the set of $T \times T$ symmetric and positive semidefinite (p.s.d.) matrices.

If one arranges the predictions of all tasks in a vector one can see multi-task learning as learning a vector-valued function in a RKHS [see 1, 13, 14, 15, 18, and references therein]. However, in this paper we use the one-to-one correspondence between real-valued and matrix-valued kernels, see [21], in order to limit the technical overhead. In this framework we define the joint kernel of input space and the set of tasks $M : (\mathcal{X} \times \{1, \ldots, T\}) \times (\mathcal{X} \times \{1, \ldots, T\}) \to \mathbb{R}$ as

$$M\big((x, s), (z, t)\big) = k(x, z)\Theta(s, t), \tag{1}$$

We denote the corresponding RKHS of functions on $\mathcal{X} \times \{1, \ldots, T\}$ as H_M and by $\|\cdot\|_{H_M}$ the corresponding norm. We formulate the output kernel learning problem for multiple tasks as

$$\min_{\Theta \in S_+^T, F \in H_M} C \sum_{i=1}^n L\big(y_i, F(x_i, t_i)\big) + \frac{1}{2} \|F\|_{H_M}^2 + \lambda V(\Theta) \tag{2}$$

where $L : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ is the convex loss function (convex in the second argument), $V(\Theta)$ is a convex regularizer penalizing the complexity of the output kernel Θ and $\lambda \in \mathbb{R}_+$ is the regularization parameter. Note that $\|F\|_{H_M}^2$ implicitly depends also on Θ. In the following we show that (2) can be reformulated into a jointly convex problem in the parameters of the prediction function and the output kernel Θ. In order to see this we first need the following representer theorem for fixed output kernel Θ.

Lemma 1 *The optimal solution $F^* \in H_M$ of the optimization problem*

$$\min_{F \in H_M} C \sum_{i=1}^{n} L\big(y_i, F(x_i, t_i)\big) + \frac{1}{2} \|F\|_{H_M}^2 \tag{3}$$

admits a representation of the form

$$F^*(x,t) = \sum_{s=1}^{T} \sum_{i=1}^{n} \gamma_{is} M\big((x_i, s), (x, t)\big) = \sum_{s=1}^{T} \sum_{i=1}^{n} \gamma_{is} k(x_i, x) \Theta(s, t),$$

where $F^*(x,t)$ is the prediction for instance x belonging to task t and $\gamma \in \mathbb{R}^{n \times T}$.

Proof: The proof is analogous to the standard representer theorem [20]. We denote by $U = \mathrm{Span}(M((x_i, s), (\cdot, \cdot)) \,|\, i = 1, \ldots, n,\ s = 1, \ldots, T)$ the subspace in H_M spanned by the training data. This induces the orthogonal decomposition of $H_M = U \oplus U^\perp$, where $U^\perp$ is the orthogonal subpace of U. Every function $F \in H_M$ can correspondingly decomposed into $F = F^\parallel + F^\perp$, where $F^\parallel \in U$ and $F^\perp \in U^\perp$. Then $\|F\|_{H_M}^2 = \|F^\parallel\|_{H_M}^2 + \|F^\perp\|_{H_M}^2$. As

$$F(x_i, t_i) = \langle F, M((x_i, t_i), (\cdot, \cdot)) \rangle = \left\langle F^\parallel, M((x_i, t_i), (\cdot, \cdot)) \right\rangle = F^\parallel(x_i, t_i). \tag{4}$$

As the loss only depends on $F^\parallel$ and we minimize the objective by having $\|F^\perp\|_{H_M} = 0$. This yields the result. □

With the explicit form of the prediction function one can rewrite the main problem (2) as

$$\min_{\Theta \in S_+^T, \gamma \in \mathbb{R}^{n \times T}} C \sum_{i=1}^{n} L\big(y_i, \sum_{s=1}^{T} \sum_{j=1}^{n} \gamma_{js} k_{ji} \Theta_{s\, t_i}\big) + \frac{1}{2} \sum_{r,s=1}^{T} \sum_{i,j=1}^{n} \gamma_{ir} \gamma_{js} k_{ij} \Theta_{rs} + \lambda\, V(\Theta), \tag{5}$$

where $\Theta_{rs} = \Theta(r, s)$ and $k_{ij} = k(x_i, x_j)$. Unfortunately, problem (5) is not jointly convex in Θ and γ due to the product in the second term. A similar problem has been analyzed in [17]. They could show that for the squared loss and $V(\Theta) = \|\Theta\|_F^2$ the corresponding optimization problem is invex and directly optimize it. For an invex function every stationary point is globally optimal [22].

We follow a different path which leads to a formulation similar to the one of [2] used for learning an input mapping (see also [9]). Our formulation for the output kernel learning problem is jointly convex in the task kernel Θ and the task parameters. We present a derivation for the general RKHS H_k, analogous to the linear case presented in [2, 9]. We use the following variable transformation,

$$\beta_{it} = \sum_{s=1}^{T} \Theta_{ts} \gamma_{is},\ i = 1, \ldots, n,\ s = 1, \ldots, T, \quad \text{resp.} \quad \gamma_{is} = \sum_{t=1}^{T} \left(\Theta^{-1}\right)_{st} \beta_{it}.$$

In the last expression Θ^{-1} has to be understood as the pseudo-inverse if Θ is not invertible. Note that this causes no problems as in case Θ is not invertible, we can without loss of generality restrict γ in (5) to the range of Θ. The transformation leads to our final problem formulation, where the prediction function F and its squared norm $\|F\|_{H_M}^2$ can be written as

$$F(x, t) = \sum_{i=1}^{n} \beta_{it} k(x_i, x), \qquad \|F\|_{H_M}^2 = \sum_{r,s=1}^{T} \sum_{i,j=1}^{n} \left(\Theta^{-1}\right)_{sr} \beta_{is} \beta_{jr} k(x_i, x_j). \tag{6}$$

This can be seen as follows

$$\begin{aligned}
\|F\|_{H_M}^2 &= \sum_{r,s=1}^{T} \sum_{i,j=1}^{n} \gamma_{ir} \gamma_{js} k(x_i, x_j) \Theta_{rs} && (7)\\
&= \sum_{t,u=1}^{T} \sum_{r,s=1}^{T} \sum_{i,j=1}^{n} \beta_{it} \beta_{ju} \left(\Theta^{-1}\right)_{tr} \left(\Theta^{-1}\right)_{us} k(x_i, x_j) \Theta_{rs} && (8)\\
&= \sum_{t,u=1}^{T} \sum_{i,j=1}^{n} \left(\Theta^{-1}\right)_{tu} \beta_{it} \beta_{ju} k(x_i, x_j). && (9)
\end{aligned}$$

We get our final primal optimization problem

$$\min_{\Theta \in S_+^T, \beta \in \mathbb{R}^{n \times T}} C \sum_{i=1}^n L\big(y_i, \sum_{j=1}^n \beta_{jt_i} k_{ji}\big) + \frac{1}{2} \sum_{r,s=1}^T \sum_{i,j=1}^n \left(\Theta^{-1}\right)_{sr} \beta_{is} \beta_{jr} k_{ij} + \lambda V(\Theta) \tag{10}$$

Before we analyze the convexity of this problem, we want to illustrate the connection to the formulations in [9, 17]. With the task weight vectors $w_t = \sum_{j=1}^n \beta_{jt} \psi(x_j) \in H_k$ we get predictions as $F(x,t) = \langle w_t, \psi(x) \rangle$ and one can rewrite

$$\|F\|_{H_M}^2 = \sum_{r,s=1}^T \sum_{i,j=1}^n \left(\Theta^{-1}\right)_{sr} \beta_{is} \beta_{jr} k(x_i, x_j) = \sum_{r,s=1}^T \left(\Theta^{-1}\right)_{sr} \langle w_s, w_t \rangle .$$

This identity is known for vector-valued RKHS, see [15] and references therein. When Θ is κ times the identity matrix, then $\|F\|_{H_M}^2 = \sum_{t=1}^T \frac{\|w_t\|^2}{\kappa}$ and thus (2) is learning the tasks independently. As mentioned before the convexity of the expression of $\|F\|_{H_M}^2$ is crucial for the convexity of the full problem (10). The following result has been shown in [2] (see also [9]).

Lemma 2 *Let $\mathcal{R}(\Theta)$ denote the range of $\Theta \in S_+^T$ and let $\Theta^\dagger$ be the pseudoinverse. The extended function $f : S_+^T \times \mathbb{R}^{n \times T} \to \mathbb{R} \cup \{\infty\}$ defined as*

$$f(\Theta, \beta) = \begin{cases} \sum_{r,s=1}^T \sum_{i,j=1}^n \left(\Theta^\dagger\right)_{sr} \beta_{is} \beta_{jr} k(x_i, x_j), & \textit{if } \beta_{i\cdot} \in \mathcal{R}(\Theta), \forall\, i = 1, \ldots, n, \\ \infty & \textit{else} . \end{cases} ,$$

is jointly convex.

Proof: It has been shown in [2] and [23][p. 223] that $\langle x, A^\dagger x \rangle$ is jointly convex on $S_+^T \times \mathcal{R}(A)$, where $\mathcal{R}(A)$ is the range of A and $A^\dagger$ is the pseudoinverse of $A \in S_+^T$. As $L := (k(x_i, x_j))_{i,j=1}^n$ is positive semi-definite we can compute the eigendecomposition as

$$L_{ij} = \sum_{l=1}^n \lambda_l u_{li} u_{lj},$$

where $\lambda_l \geq 0$, $l = 1, \ldots, n$ are the eigenvalues and $u_l \in \mathbb{R}^n$ the eigenvectors. Using this we get

$$\sum_{r,s=1}^T \sum_{i,j=1}^n \left(\Theta^{-1}\right)_{sr} \beta_{is} \beta_{jr} k(x_i, x_j) = \sum_{l=1}^n \lambda_l \sum_{r,s=1}^T \Big(\sum_{i=1}^n \beta_{is} u_{li}\Big)\Big(\sum_{j=1}^n \beta_{jr} u_{lj}\Big) \left(\Theta^{-1}\right)_{rs} \tag{11}$$

and thus we can write the function f as a positive combination of convex functions, where the arguments are composed with linear mappings which preserves convexity [24]. $\square$

The formulation in (10) is similar to [9, 17, 18]. [9] uses the constraint $\operatorname{Trace}(\Theta) \leq 1$ instead of a regularizer $V(\Theta)$ enforcing low rank of the output kernel. On the other hand, [17] employs squared Frobenius norm for $V(\Theta)$ with squared loss function. [18] proposed an efficient algorithm for convex $V(\Theta)$. Instead we think that sparsity of Θ is better to avoid the emergence of spurious relations between tasks and also leads to output kernels which are easier to interpret. Thus we propose to use the following regularization functional for the output kernel Θ:

$$V(\Theta) = \sum_{t,t'=1}^T |\Theta_{tt'}|^p = \|\Theta\|_p^p ,$$

for $p \in [1, 2]$. Several approaches [9, 17, 18] employ alternate minimization scheme, involving costly eigendecompositions of $T \times T$ matrix per iteration (as $\Theta \in S_+^T$). In the next section we show that for a certain set of values of p one can derive an unconstrained dual optimization problem which thus avoids the explicit minimization over the S_+^T cone. The resulting *unconstrained* dual problem can then be easily optimized by stochastic coordinate ascent. Having explicit expressions of the primal variables Θ and β in terms of the dual variables allows us to get back to the original problem.

3 Unconstrained Dual Problem Avoiding Optimization over S_+^T

The primal formulation (10) is a convex multi-task output kernel learning problem. The next lemma derives the Fenchel dual function of (10). This still involves the optimization over the primal variable $\Theta \in S_+^T$. A main contribution of this paper is to show that this optimization problem over the S_+^T cone can be solved with an analytical solution for a certain class of regularizers $V(\Theta)$. In the following we denote by $\alpha^r := \{\alpha_i \,|\, t_i = r\}$ the dual variables corresponding to task r and by K_{rs} the kernel matrix $(k(x_i, x_j) \,|\, t_i = r, t_j = s)$ corresponding to the dual variables of tasks r and s.

Lemma 3 *Let L_i^* be the conjugate function of the loss $L_i : \mathbb{R} \to \mathbb{R}, u \mapsto L(y_i, u)$, then*

$$q : \mathbb{R}^n \to \mathbb{R}, q(\alpha) = -C \sum_{i=1}^{n} L_i^*\Big(-\frac{\alpha_i}{C}\Big) - \lambda \max_{\Theta \in S_+^T} \Big(\frac{1}{2\lambda} \sum_{r,s=1}^{T} \Theta_{rs} \left\langle \alpha^r, K_{rs}\alpha^s \right\rangle - V(\Theta)\Big) \tag{12}$$

is the dual function of (10)*, where $\alpha \in \mathbb{R}^n$ are the dual variables. The primal variable $\beta \in \mathbb{R}^{n\times T}$ in* (10) *and the prediction function F can be expressed in terms of Θ and α as $\beta_{is} = \alpha_i \Theta_{st_i}$ and $F(x, s) = \sum_{j=1}^{n} \alpha_j \Theta_{st_j} k(x_j, x)$ respectively, where t_j is the task of the j-th training example.*

Proof: We derive the Fenchel dual function of (10). For this purpose we introduce auxiliary variables $z \in \mathbb{R}^n$ which satisfy the constraint

$$z_i = \sum_{j=1}^{n} \beta_{jt_i} k(x_j, x_i) = F(x_i, t_i).$$

The Lagrangian L of the resulting problem (10) is given as:

$$\begin{aligned} L(\beta, \Theta, z, \alpha) = C \sum_{i=1}^{n} L(y_i, z_i) + \frac{1}{2} \sum_{r,s=1}^{T} \sum_{i,j=1}^{n} \left(\Theta^{-1}\right)_{sr} \beta_{is}\beta_{jr} k(x_i, x_j) \\ + \sum_{i=1}^{n} \alpha_i \Big(z_i - \sum_{j=1}^{n} \beta_{jt_i} k(x_j, x_i)\Big) + i_{S_+^T}(\Theta) + \lambda V(\Theta). \end{aligned} \tag{13}$$

where i_C is the indicator function of the set C. The dual function q is defined as

$$q(\alpha) = \min_{\beta \in \mathbb{R}^{n\times T},\, \Theta \in S_+^T,\, z \in \mathbb{R}^n} L(\beta, \Theta, z, \alpha). \tag{14}$$

Using the definition of the conjugate function [24], we get

$$\min_{z_i \in \mathbb{R}} C\, L(y_i, z_i) + \alpha_i z_i = C \min_{z_i \in \mathbb{R}} L(y_i, z_i) + \frac{\alpha_i}{C} z_i = -C \max_{z_i \in \mathbb{R}} \Big(-\frac{\alpha_i}{C} z_i - L(y_i, z_i)\Big) \tag{15}$$

$$= -C\, L_i^*\big(-\frac{\alpha_i}{C}\big), \tag{16}$$

where L_i^* is the conjugate function of $L_i : z \to L(y_i, z)$. Moreover, we compute the minimizer with respect to β, via

$$\begin{aligned} &\frac{\partial}{\partial \beta_{lu}} \Big(\frac{1}{2} \sum_{r,s=1}^{T} \sum_{i,j=1}^{n} \left(\Theta^{-1}\right)_{sr} \beta_{is}\beta_{jr} k(x_i, x_j) - \sum_{i=1}^{n} \alpha_i \big(\sum_{j=1}^{n} \beta_{jt_i} k(x_j, x_i)\big)\Big) \\ =& \sum_{r=1}^{T} \sum_{j=1}^{n} \beta_{jr} (\Theta^{-1})_{ur} k(x_l, x_j) - \sum_{i=1}^{n} \alpha_i \delta_{ut_i} k(x_l, x_i), \end{aligned} \tag{17}$$

where δ is the Kronecker symbol, that is $\delta_{ut_i} = \begin{cases} 1 & \text{if } u = t_i, \\ 0 & \text{else} \end{cases}$. Solving for the global minimizer β^* yields

$$\beta_{jr}^* = \alpha_j \Theta_{rt_j}. \tag{18}$$

Plugging β^* back into the above expressions yields

$$\begin{aligned}\sum_{r,s=1}^{T}\sum_{i,j=1}^{n}\left(\Theta^{-1}\right)_{sr}\beta_{is}\beta_{jr}k(x_i,x_j) &= \sum_{r,s=1}^{T}\sum_{i,j=1}^{n}\left(\Theta^{-1}\right)_{sr}\Theta_{st_i}\Theta_{rt_j}\alpha_i\alpha_j k(x_i,x_j)\\ &= \sum_{i,j=1}^{n}\Theta_{t_it_j}\alpha_i\alpha_j k(x_i,x_j), \qquad (19)\end{aligned}$$

$$\sum_{i,j=1}^{n}\alpha_i\beta_{jt_i}k(x_j,x_i) = \sum_{i,j=1}^{n}\alpha_i\alpha_j\Theta_{t_jt_i}k(x_j,x_i), \qquad (20)$$

Introducing $\alpha^r = (\alpha_i)_{t_i=r}$, $K_{rs} = \big(k(x_i,x_j)\big)_{t_i=r,t_j=s}$ and gathering the terms corresponding to the individual tasks we get

$$\sum_{i,j=1}^{n}\alpha_i\alpha_j\Theta_{t_jt_i}k(x_j,x_i) = \sum_{r,s=1}^{T}\langle\alpha^r, K_{rs}\alpha^s\rangle\,.$$

Plugging all the expressions back into (14), we get the dual function as

$$\begin{aligned}q(\alpha) &= -C\,L^*_{ti}(-\frac{\alpha_{ti}}{C}) + \min_{\Theta\in S^T_+}\lambda\,V(\Theta) - \frac{1}{2}\sum_{r,s=1}^{T}\Theta_{rs}\,\langle\alpha^r, K_{rs}\alpha^s\rangle \qquad (21)\\ &= -C\,L^*_{ti}(-\frac{\alpha_{ti}}{C}) + \lambda\min_{\Theta\in S^T_+}V(\Theta) - \langle\rho,\Theta\rangle \qquad (22)\\ &= -C\,L^*_{ti}(-\frac{\alpha_{ti}}{C}) - \lambda\max_{\Theta\in S^T_+}\langle\rho,\Theta\rangle - V(\Theta) \qquad (23)\end{aligned}$$

where we have introduced in the second step $\rho\in\mathbb{R}^{T\times T}$ with

$$\rho_{rs} = \frac{1}{2\lambda}\langle\alpha^r, K_{rs}\alpha^s\rangle\,, \quad r,s = 1,\ldots,T.$$

Note that ρ is a Gram matrix and thus positive semidefinite. The expression for the prediction function is obtained by plugging (18) into (6). $\square$

We now focus on the remaining maximization problem in the dual function in (12)

$$\max_{\Theta\in S^T_+}\frac{1}{2\lambda}\sum_{r,s=1}^{T}\Theta_{rs}\,\langle\alpha^r, K_{rs}\alpha^s\rangle - V(\Theta). \qquad (24)$$

This is a semidefinite program which is computationally expensive to solve and thus prohibits to scale the output kernel learning problem to a large number of tasks. However, we show in the following that this problem has an analytical solution for a subset of the regularizers $V(\Theta) = \frac{1}{2}\sum_{r,s=1}^{T}|\Theta_{rs}|^p$ for $p\geq 1$. For better readability we defer a more general result towards the end of the section. The basic idea is to relax the constraint on $\Theta\in R^{T\times T}$ in (24) so that it is equivalent to the computation of the conjugate V^* of V. If the maximizer of the relaxed problem is positive semi-definite, one has found the solution of the original problem.

Theorem 4 *Let $k\in\mathbb{N}$ and $p=\frac{2k}{2k-1}$, then with $\rho_{rs}=\frac{1}{2\lambda}\langle\alpha^r, K_{rs}\alpha^s\rangle$ we have*

$$\max_{\Theta\in S^T_+}\sum_{r,s=1}^{T}\Theta_{rs}\rho_{rs} - \frac{1}{2}\sum_{r,s=1}^{T}|\Theta_{rs}|^p \;=\; \frac{1}{4k-2}\Big(\frac{2k-1}{2k\lambda}\Big)^{2k}\sum_{r,s=1}^{T}\langle\alpha^r, K_{rs}\alpha^s\rangle^{2k}\,, \qquad (25)$$

and the maximizer is given by the positive semi-definite matrix

$$\Theta^*_{rs} = \Big(\frac{2k-1}{2k\lambda}\Big)^{2k-1}\langle\alpha^r, K_{rs}\alpha^s\rangle^{2k-1}\,, \quad r,s=1,\ldots,T. \qquad (26)$$

Proof: We relax the constraints and solve

$$\max_{\Theta \in \mathbb{R}^{T\times T}} \frac{1}{2\lambda} \sum_{r,s=1}^{T} \Theta_{rs} \left\langle \alpha^r, K_{rs}\alpha^s \right\rangle - \frac{1}{2}\sum_{r,s=1}^{T} |\Theta_{rs}|^p.$$

Note that the problem is separable and thus we can solve for each component separately,

$$\max_{\Theta_{rs} \in \mathbb{R}} \frac{1}{2\lambda} \Theta_{rs} \left\langle \alpha^r, K_{rs}\alpha^s \right\rangle - \frac{1}{2}|\Theta_{rs}|^p.$$

The optimality condition for Θ^*_{rs} becomes with $\rho_{rs} = \frac{1}{2\lambda}\left\langle \alpha^r, K_{rs}\alpha^s \right\rangle$,

$$0 = \rho_{rs} - \frac{p}{2}\,\mathrm{sign}(\Theta^*_{rs})|\Theta^*_{rs}|^{p-1} \implies \Theta^*_{rs} = \left(\frac{2}{p}\right)^{\frac{1}{p-1}} \mathrm{sign}(\rho_{rs})|\rho_{rs}|^{\frac{1}{p-1}}.$$

The solution of the relaxed problem is the solution of the original constrained problem, if we can show that the corresponding maximizer is positive semidefinite. Note that $\rho_{rs} = \frac{1}{2\lambda}\left\langle \alpha^r, K_{rs}\alpha^s \right\rangle$ is a positive semidefinite (p.s.d.) matrix as it is a Gram matrix. The factor $\left(\frac{2}{p}\right)^{\frac{1}{p-1}}$ is positive and thus the resulting matrix is p.s.d. if $\mathrm{sign}(\rho_{rs})|\rho_{rs}|^{\frac{1}{p-1}}$ is p.s.d.

It has been shown [25], that the elementwise power A^l_{rs} of a positive semidefinite matrix A is positive definite for all $A \in S^T_+$ and $T \in \mathbb{N}$ if and only if l is a positive integer. Note that we have an elementwise integer power of Θ if $\frac{1}{p-1}$ is an odd positive integer (the case of an even integer is ruled out by Theorem 5), that is $\frac{1}{p-1} = 2k-1$ for $k \in \mathbb{N}$ as in this case we have

$$\Theta^*_{rs} = \left(\frac{2}{p}\right)^{2k-1} \mathrm{sign}(\rho_{rs})|\rho_{rs}|^{2k-1} = \left(\frac{2}{p}\right)^{2k-1} \rho_{rs}^{2k-1} = \left(\frac{2k-1}{2k\lambda}\right)^{2k-1} \left\langle \alpha^r, K_{rs}\alpha^s \right\rangle^{2k-1}.$$

We get the admissible values of p as $p = \frac{2k}{2k-1}$, $k \in \mathbb{N}$ (resp. $2k = \frac{p}{p-1}$). We compute the optimal objective value as

$$\sum_{r,s=1}^{T} \rho_{rs}^{2k}\Big(\left(\frac{2}{p}\right)^{2k-1} - \frac{1}{2}\left(\frac{2}{p}\right)^{2k}\Big) = (p-1)\frac{1}{2}\left(\frac{2}{p}\right)^{2k}\sum_{r,s=1}^{T}\rho_{rs}^{2k} = \frac{1}{4k-2}\Big(\frac{2k-1}{k}\Big)^{2k}\sum_{r,s=1}^{T}\rho_{rs}^{2k} \tag{27}$$

$$= \frac{1}{4k-2}\Big(\frac{2k-1}{2\lambda\, k}\Big)^{2k}\sum_{r,s=1}^{T}\left\langle \alpha^r, K_{rs}\alpha^s \right\rangle^{2k} \tag{28}$$

□

Plugging the result of the previous theorem into the dual function of Lemma 3 we get for $k \in \mathbb{N}$ and $p = \frac{2k}{2k-1}$ with $V(\Theta) = \|\Theta\|_p^p$ the following unconstrained dual of our main problem (10):

$$\max_{\alpha \in \mathbb{R}^n} -C\sum_{i=1}^{n} L_i^*\Big(-\frac{\alpha_i}{C}\Big) - \frac{\lambda}{4k-2}\Big(\frac{2k-1}{2k\lambda}\Big)^{2k}\sum_{r,s=1}^{T}\left\langle \alpha^r, K_{rs}\alpha^s \right\rangle^{2k}. \tag{29}$$

Note that by doing the variable transformation $\kappa_i := \frac{\alpha_i}{C}$ we effectively have only one hyper-parameter in (29). This allows us to cross-validate more efficiently. The range of admissible values for p in Theorem 4 lies in the interval $(1, 2]$, where we get for $k = 1$ the value $p = 2$ and as $k \to \infty$ we have $p \to 1$. The regularizer for $p = 2$ together with the squared loss has been considered in the primal in [17, 18]. Our analytical expression of the dual is novel and allows us to employ stochastic dual coordinate ascent to solve the involved primal optimization problem. Please also note that by optimizing the dual, we have access to the duality gap and thus a well-defined stopping criterion. This is in contrast to the alternating scheme of [17, 18] for the primal problem which involves costly matrix operations. Our runtime experiments show that our solver for (29) outperforms the solvers of [17, 18]. Finally, note that even for suboptimal dual variables α, the corresponding Θ matrix in (26) is positive semidefinite. Thus we always get a feasible set of primal variables.

Table 1: Examples of regularizers $V(\Theta)$ together with their generating function ϕ and the explicit form of Θ^* in terms of the dual variables, $\rho_{rs} = \frac{1}{2\lambda}\langle \alpha^r, K_{rs}\alpha^s\rangle$. The optimal value of (24) is given in terms of ϕ as $\max_{\Theta\in\mathbb{R}^{T\times T}} \langle \rho, \Theta\rangle - V(\Theta) = \sum_{r,s=1}^{T} \phi(\rho_{rs})$.

$\phi(z)$	$V(\Theta)$	Θ^*_{rs}
$\frac{z^{2k}}{2k}$, $k \in \mathbb{N}$	$\frac{2k-1}{2k}\sum\limits_{r,s=1}^{T} \lvert\Theta_{rs}\rvert^{\frac{2k}{2k-1}}$	ρ_{rs}^{2k-1}
$e^z = \sum_{k=0}^{\infty}\frac{z^k}{k!}$	$\begin{cases}\sum\limits_{r,s=1}^{T}\Theta_{rs}\log(\Theta_{rs}) - \Theta_{rs} & \text{if } \Theta_{rs} > 0 \forall r,s \\ \infty & \text{else .}\end{cases}$	$e^{\rho_{rs}}$
$\cosh(z) - 1 = \sum_{k=1}^{\infty}\frac{z^{2k}}{(2k)!}$	$\sum\limits_{r,s=1}^{T}\left(\Theta_{rs}\operatorname{arcsinh}(\Theta_{rs}) - \sqrt{1+\Theta_{rs}^2}\right) + T^2$	$\operatorname{arcsinh}(\rho_{rs})$

Characterizing the set of convex regularizers V which allow an analytic expression for the dual function The previous theorem raises the question for which class of convex, separable regularizers we can get an analytical expression of the dual function by explicitly solving the optimization problem (24) over the positive semidefinite cone. A key element in the proof of the previous theorem is the characterization of functions $f : \mathbb{R} \to \mathbb{R}$ which when applied elementwise $f(A) = (f(a_{ij}))_{i,j=1}^{T}$ to a positive semidefinite matrix $A \in S_+^T$ result in a p.s.d. matrix, that is $f(A) \in S_+^T$. This set of functions has been characterized by Hiai [26].

Theorem 5 ([26]) *Let $f : \mathbb{R} \to \mathbb{R}$ and $A \in S_+^T$. We denote by $f(A) = (f(a_{ij}))_{i,j=1}^{T}$ the elementwise application of f to A. It holds $\forall\, T \geq 2, \quad A \in S_+^T \implies f(A) \in S_+^T$ if and only if f is analytic and $f(x) = \sum_{k=0}^{\infty} a_k x^k$ with $a_k \geq 0$ for all $k \geq 0$.*

Note that in the previous theorem the condition on f is only necessary when we require the implication to hold for all T. If T is fixed, the set of functions is larger and includes even (large) fractional powers, see [25]. We use the stronger formulation as we want that the result holds without any restriction on the number of tasks T. Theorem 5 is the key element used in our following characterization of separable regularizers of Θ which allow an analytical expression of the dual function.

Theorem 6 *Let $\phi : \mathbb{R} \to \mathbb{R}$ be analytic on $\mathbb{R}$ and given as $\phi(z) = \sum_{k=0}^{\infty}\frac{a_k}{k+1}z^{k+1}$ where $a_k \geq 0\ \forall k \geq 0$. If ϕ is convex, then, $V(\Theta) := \sum_{r,s=1}^{T}\phi^*(\Theta_{rs})$, is a convex function $V : \mathbb{R}^{T\times T} \to \mathbb{R}$ and*

$$\max_{\Theta\in\mathbb{R}^{T\times T}} \langle \rho, \Theta\rangle - V(\Theta) \;=\; V^*(\rho) \;=\; \sum_{r,s=1}^{T}\phi\big(\rho_{rs}\big), \tag{30}$$

where the global maximizer fulfills $\Theta^* \in S_+^T$ if $\rho \in S_+^T$ and $\Theta^*_{rs} = \sum_{k=0}^{\infty} a_k\rho_{rs}^k$.

Proof: Note that ϕ is analytic on $\mathbb{R}$ and thus infinitely differentiable on $\mathbb{R}$. As ϕ is additionally convex, it is a proper, lower semi-continuous convex function and thus $(\phi^*)^* = \phi$ [27, Corollary 1.3.6]. As ϕ^* is convex, V is a convex function and using $(\phi^*)^* = \phi$ we get

$$\max_{\Theta\in\mathbb{R}^{T\times T}} \langle \rho, \Theta\rangle - V(\Theta) = V^*(\rho) = \sum_{r,s=1}^{T}\phi(\rho_{rs}). \tag{31}$$

Finally, we show that the global maximizer has the given form. Note that as ϕ is a proper, lower semi-continuous convex function it holds [27, Corollary 1.4.4]

$$\Theta_{rs} \in \partial\phi^*(\rho_{rs}) \quad\iff\quad \rho_{rs} \in \partial\phi(\Theta_{rs}).$$

Note that the maximizer Θ^*_{rs} of problem (31) fulfills $\rho_{rs} \in \frac{\partial\phi^*}{\partial\Theta_{rs}}(\Theta^*_{rs})$ and thus $\Theta^*_{rs} = \frac{\partial\phi}{\partial\rho_{rs}}(\rho_{rs})$, where we have used that ϕ is infinitely differentiable. These conditions allow us to express the maximizer of (30) in terms of $\partial\phi$. As ϕ is continuously differentiable, we get

$$\Theta^*_{rs} = \frac{\partial\phi}{\partial\rho_{rs}}(\rho_{rs}) = \sum_{k=0}^{\infty} a_k\rho_{rs}^k.$$

Note that the series has infinite convergence radius and $a_k \geq 0$ for all k and thus it is of the form provided in Theorem 5. Thus $\Theta^* \in S_+^T$ if $\rho \in S_+^T$. $\square$

Table 1 summarizes e.g. of functions ϕ, the corresponding $V(\Theta)$ and the maximizer Θ^* in (30).

Examples

- First we recover the results of Theorem 4. We use $\phi(x) = \frac{1}{2k}x^{2k}$ for $k \in \mathbb{N}$, which is convex. We compute

$$\phi^*(y) = \sup_{x\in\mathbb{R}} xy - \phi(x) = \sup_{x\in\mathbb{R}} xy - \frac{1}{2k}x^{2k} = \frac{2k-1}{2k}|y|^{\frac{2k}{2k-1}},$$

  where we have used $x^* = |y|^{\frac{1}{2k-1}}\operatorname{sign}(y)$. We recover

$$V(\Theta) = \sum_{r,s=1}^{T} \phi^*(\Theta_{rs}) = \frac{2k-1}{2k}\sum_{r,s=1}^{T} \Theta_{rs}^{\frac{2k}{2k-1}},$$

  which with $p = \frac{2k}{2k-1}$ yields up to a positive factor the family of regularizers employed in Theorem 4 together with

$$\Theta^*_{rs} = \rho_{rs}^{2k-1}$$

- In the second example we use $\phi(x) = e^x = \sum_{k=0}^{\infty}\frac{x^k}{k!}$, which is convex and the series has infinite convergence radius The conjugate ϕ^* is given as

$$\phi^*(y) = \sup_{x\in\mathbb{R}} xy - e^x = \begin{cases} y\log(y) - y & \text{if } y > 0 \\ \infty & \text{else.}\end{cases}$$

  so that the regularizer is given by,

$$V(\Theta) = \sum_{r,s=1}^{T}\phi^*(\Theta_{rs}) = \begin{cases}\sum_{r,s=1}^{T}\Theta_{rs}\log(\Theta_{rs}) - \Theta_{rs} & \text{if } \Theta_{rs} > 0 \ \forall r,s = 1,\dots,T \\ \infty & \text{else .}\end{cases}.$$

  This can be seen as a generalized KL-divergence between Θ and Θ_0, where $\Theta_0 \in S_+^T$ is the matrix of all ones

$$V(\Theta) = \sum_{r,s=1}^{T}\phi^*(\Theta_{rs}) = \begin{cases}\sum_{r,s=1}^{T}\Theta_{rs}\log\left(\frac{\Theta_{rs}}{\left(\Theta_0\right)_{rs}}\right) - \Theta_{rs} + \left(\Theta_0\right)_{rs} & \text{if } \Theta_{rs} > 0 \ \forall r,s \\ \infty & \text{else .}\end{cases}.$$

  Note that adding the constant term $\sum_{r,s=1}^{T}\left(\Theta_0\right)_{rs}$ does not change the optimization problem (10). The corresponding Θ^* is given by

$$\Theta^*_{rs} = \sum_{k=0}^{\infty}\frac{\rho_{rs}^k}{k!} = e^{\rho_{rs}}.$$

- Next we consider $\phi(x) = \cosh(x) - 1 = \sum_{k=1}^{\infty}\frac{x^{2k}}{(2k)!}$ which is obviously convex and the series has infinite convergence radius (e^x is majorant). The conjugate ϕ^* can be computed as

$$\phi^*(y) = \sup_{x\in\mathbb{R}} xy - \cosh(x) + 1 = y\operatorname{arcsinh}(y) - \sqrt{1+y^2} + 1 = y\log(y + \sqrt{y^2+1}) - \sqrt{1+y^2} + 1.$$

  so that the regularizer is given by

$$V(\Theta) = \sum_{r,s=1}^{T}\phi^*(\Theta_{rs}) = \sum_{r,s=1}^{T}\Big(\Theta_{rs}\operatorname{arcsinh}(\Theta_{rs}) - \sqrt{1+\Theta_{rs}^2} + 1\Big).$$

  The corresponding Θ^* is given by

$$\Theta^*_{rs} = \operatorname{arcsinh}(\rho_{rs}) = \log\left(\rho_{rs} + \sqrt{\rho_{rs}^2 + 1}\right).$$

  This regularizer is interpolating between a squared norm and a variant of 1-norm. One has

$$\lim_{y\to 0}\phi^*(y) = \frac{y^2}{2}, \quad \lim_{y\to\infty}\phi^*(y) = |y|(\log(2|y|) - 1) + 1.$$

Algorithm 1 Fast MTL-SDCA

```
Input: Gram matrix $K$, label vector $y$, regularization parameter and relative duality gap parameter $\epsilon$
Output: $\alpha$ ($\Theta$ is computed from $\alpha$ using our result in 26)
Initialize $\alpha = \alpha^{(0)}$
repeat
  Let $\{i_1, \ldots, i_n\}$ be a random permutation of $\{1, \ldots, n\}$
  for $j = 1, \ldots, n$ do
    Solve for $\Delta$ in (32) corresponding to $\alpha_{i_j}$
    $\alpha_{i_j} \leftarrow \alpha_{i_j} + \Delta$
  end for
until Relative duality gap is below $\epsilon$
```

4 Optimization Algorithm

The dual problem (29) can be efficiently solved via decomposition based methods like stochastic dual coordinate ascent algorithm (SDCA) [19]. SDCA enjoys low computational complexity per iteration and has been shown to scale effortlessly to large scale optimization problems.

Our algorithm for learning the output kernel matrix and task parameters is summarized in Algorithm 1. At each step of the iteration we optimize the dual objective over a randomly chosen α_i variable. Let $t_i = r$ be the task corresponding to α_i. We apply the update $\alpha_i \leftarrow \alpha_i + \Delta$. The optimization problem of solving (29) with respect to Δ is as follows:

$$\min_{\Delta \in \mathbb{R}} L_i^*\big((-\alpha_i - \Delta)/C\big) + \eta\big((a\Delta^2 + 2b_{rr}\Delta + c_{rr})^{2k} + 2\sum_{s \neq r}(b_{rs}\Delta + c_{rs})^{2k} + \sum_{s,z \neq r} c_{sz}^{2k}\big), \quad (32)$$

where $a = k_{ii}$, $b_{rs} = \sum_{j:t_j=s} k_{ij}\alpha_j \ \forall s$, $c_{sz} = \langle \alpha^s, K_{sz}\alpha^z \rangle \ \forall s, z$ and $\eta = \frac{\lambda}{C(4k-2)}\Big(\frac{2k-1}{2k\lambda}\Big)^{2k}$. This one-dimensional convex optimization problem is solved efficiently via Newton method. The complexity of the proposed algorithm is $O(T)$ per iteration . The proposed algorithm can also be employed for learning output kernels regularized by generic $V(\Theta)$, discussed in the previous section.

Special case $p = 2(k = 1)$: For certain loss functions such as the hinge loss, the squared loss, etc., $L_{ti}^*\big(-\frac{\alpha_{ti}+\Delta}{C}\big)$ yields a linear or a quadratic expression in Δ. In such cases problem (32) reduces to finding the roots of a cubic equation, which has a closed form expression. Hence, our algorithm is highly efficient with the above loss functions when Θ is regularized by the squared Frobenius norm.

5 Empirical Results

In this section, we present our results on benchmark data sets comparing our algorithm with existing approaches in terms of generalization accuracy as well as computational efficiency. In Section 5.1, we discuss generalization results in multi-task setting. We evaluate the performance of our algorithm against several recent multi-task methods that employ clustering, low-dimensional projection of input feature space or output kernel learning. Section 5.2 discusses multi-class experiment results. Single task learning (**STL**) is a common baseline in both these experiments, and it employs hinge loss and ϵ-SVR loss functions for classification and regression problems respectively. Finally, in Section 5.3, we discuss the results on the computational efficiency of our algorithm.

5.1 Multi-Task Data Sets

We begin with the generalization results in multi-task setups. The data sets are as follows:
Sarcos: A multi-task regression data set. The aim is to predict 7 degrees of freedom of a robotic arm [28].
Parkinson: A multi-task regression data set [29] where one needs to predict the Parkinson's disease symptom score for 42 patients.
Yale: A face recognition data set from the Yale face base with 28 binary classification tasks [30].
Landmine: A data set containing binary classification problems from 19 different landmines [30].
MHC-I: A bioinformatics data set having 10 binary classification tasks [12].

Table 2: Dataset statistics. T represents the number of tasks and m represents the average number of training examples per task.

Dataset	T	m	Dataset	T	m
Sarcos	7	15	Landmine	19	102
Parkinson	42	5	MHC-I	10	24
Yale	28	5	Letter	9	60
USPS	10	100	MNIST	10	100
MIT Indoor67	67	80	SUN397	397	5, 50

Table 3: Mean generalization performance and the standard deviation over ten train-test splits.

Data set	**STL**	**MTL**	**CMTL**	**MTFL**	**GMTL**	**MTRL**	**FMTL$_p$**		
							$p = 2$	$p = 4/3$	$p = 8/7$
Regression data sets: Explained Variance (%)									
Sarcos	40.5±7.6	34.5±10.2	33.0±13.4	49.9±6.3	45.8±10.6	41.6±7.1	46.7±6.9	**50.3±5.8**	48.4±5.8
Parkinson	2.8±7.5	4.9±20.0	2.7±3.6	16.8±10.8	**33.6±9.4**	12.0±6.8	27.0±4.4	27.0±4.4	27.0±4.4
Classification data sets: AUC (%)									
Yale	93.4±2.3	96.4±1.6	95.2±2.1	**97.0±1.6**	91.9±3.2	96.1±2.1	**97.0±1.2**	**97.0±1.4**	96.8±1.4
Landmine	74.6±1.6	76.4±0.8	75.9±0.7	76.4±1.0	76.7±1.2	76.1±1.0	**76.8±0.8**	76.7±1.0	76.4±0.9
MHC-I	69.3±2.1	72.3±1.9	**72.6±1.4**	71.7±2.2	72.5±2.7	71.5±1.7	71.7±1.9	70.8±2.1	70.7±1.9
Letter	61.2±0.8	61.0±1.6	60.5±1.1	60.5±1.8	61.2±0.9	60.3±1.4	61.4±0.7	**61.5±1.0**	61.4±1.0

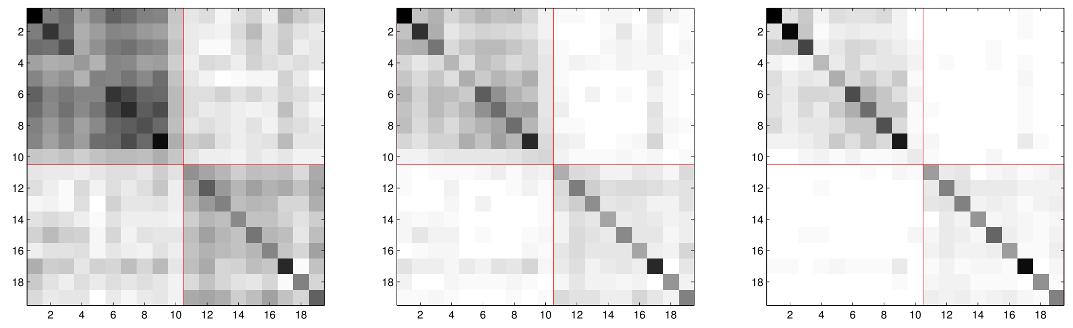
Letter: A data set containing handwritten letters from several writers and having 9 binary classification tasks [31].

Table 2 presents the data set statistics. We compare the following algorithms:
MTL [16]: A classical multi-task learning baseline. They define the Θ matrix as: $\Theta(t, t') = \frac{1}{\mu} + \delta_{tt'}$, where $\mu > 0$ is a hyper-parameter and $\delta_{tt'} = 1$ if $t = t'$ else $\delta_{tt'} = 0$. The hyper-parameter μ is cross-validated.
CMTL [12]: A clustered multi-task learning algorithm. Tasks within a cluster are assumed to be close to a mean vector. It requires the number of task clusters as a hyper-parameter.
MTFL [11]: Learns the input kernel and the output kernel matrix as a linear combination of base kernel matrices.
GMTL [10]: A clustered multi-task feature learning approach. Tasks within a cluster are assumed to share a low dimensional feature subspace [2]. Hence, it effectively learns both the input kernel as well as the output kernel.
MTRL [9]: A multi-task relationship learning approach. It learns a low rank output kernel matrix by enforcing a trace constraint on it.
FMTL$_p$: Our proposed multi-task learning formulation (29). We consider three different values for the p-norm: $p = 2$ $(k = 1)$, $p = 4/3$ $(k = 2)$ and $p = 8/7$ $(k = 4)$. Hinge and ϵ-SVR loss functions were used for classification and regression problems respectively.

We follow the experimental protocol[1] described in [11]. Three-fold cross validation was performed for parameter selection. Linear kernel was employed for all data sets. Also, note that GMTL [10] and MTFL [11] enjoy the advantage of both input and output kernel learning. Hence, their generalization results are not directly comparable to our method, which focuses solely on learning the output kernel matrix.

Table 3 reports the performance of the algorithms averaged over ten random train-test splits. The proposed FMTL$_p$ attains the best generalization accuracy in general. It outperforms the baseline MTL as well as MTRL and CMTL, which solely learns the output kernel matrix. Moreover, it achieves an overall better performance than GMTL and MTFL. The FMTL$_{p=4/3,8/7}$ give comparable generalization to $p = 2$ case, with the additional benefit of learning sparser and more interpretable output kernel matrix (see Figure 1).

[1]The performance of STL, MTL, CMTL and MTFL are reported from [11].



$(p = 2)$ $(p = 4/3)$ $(p = 8/7)$

Figure 1: Plots of $|\Theta|$ matrices (rescaled to [0,1] and averaged over ten splits) computed by our solver FMTL$_p$ for the Landmine data set for different p-norms, with cross-validated hyper-parameter values. The darker regions indicate higher value. Tasks (landmines) numbered 1-10 correspond to highly foliated regions and those numbered 11-19 correspond to bare earth or desert regions. Hence, we expect two groups of tasks (indicated by the red squares). We can observe that the learned Θ matrix at $p = 2$ depicts much more spurious task relationships than the ones at $p = 4/3$ and $p = 8/7$. Thus, our sparsifying regularizer improves interpretability.

Table 4: Mean accuracy and the standard deviation over five train-test splits.

Data set	**STL**	**MTL-SDCA**	**GMTL**	**MTRL**	**FMTL$_p$-H** $p=2$	**FMTL$_p$-H** $p=8/7$	**FMTL$_p$-S** $p=2$	**FMTL$_p$-S** $p=8/7$	**FMTL$_{kl}$**
MNIST	84.1±0.3	86.0±0.2	84.8±0.3	85.6±0.4	86.1±0.4	**86.2±0.4**	82.3±0.6	82.4±0.3	82.5±0.5
USPS	90.5±0.3	90.6±0.2	91.6±0.3	92.4±0.2	92.4±0.2	**92.6±0.1**	87.2±0.4	87.5±0.3	87.0±0.4

5.2 Multi-Class Data Sets

The multi-class setup is cast as T one-vs-all binary classification tasks, corresponding to T classes. In this section we experimented with two loss functions: a) FMTL$_p$-H – the hinge loss employed in SVMs, and b) FMTL$_p$-S – the squared loss employed in OKL [17]. In these experiments, we also compare our results with **MTL-SDCA**, a state-of-the-art multi-task feature learning method [32]. In addition, we report results from our KL-divergence regularized formulation with squared loss (denoted by FMTL$_{kl}$).

Handwritten Digit Recognition: We consider the following two data sets and follow the experimental protocol detailed in [10].
USPS: A handwritten digit data sets with 10 classes [33]. We process the images using PCA and reduce the dimensionality to 87. This retains almost 87% of variance.
MNIST: Another handwritten digit data set with 10 classes [34]. PCA is employed to reduce the dimensionality to 64.

We use 1000, 500 and 500 examples for training, validation and test respectively. Table 4 reports the average accuracy achieved by various methods on both data sets over 5 splits. Our approach FMTL$_p$-H obtains better accuracy than GMTL, MTRL and MTL-SDCA [32] on both data sets.

MIT Indoor67 Experiments: We also report results on the MIT Indoor67 benchmark [35] which covers 67 indoor scene categories with over 100 images per class. We use the train/test split (80/20 images per class) provided by the authors. FMTL$_p$-S achieved the accuracy of 73.1%, 73.1% and 73.3% with $p = 2, 4/3$ and $8/7$ respectively. Our KL-divergence regularized approach FMTL$_{kl}$ obtained 73.1%. Note that these are better than the ones reported in [36] (70.1%) and [35] (68.24%).

SUN397 Experiments: SUN397 [37] is a challenging scene classification benchmark [35] with 397 scene classes and more than 100 images per class. We use $m = 5, 50$ images per class for training, 50 images per class for testing and report the average accuracy over the 10 standard splits. We employed the CNN features extracted with the convolutional neural network (CNN) provided by [35] using Places 205 database. We resized the images directly to 227×227 pixels, which is the

Table 5: Mean accuracy and the standard deviation over ten train-test splits on SUN397.

m	**STL**	**MTL**	**MTL-SDCA**	**FMTL$_p$-H** $p=2$	$p=4/3$	$p=8/7$	**FMTL$_p$-S** $p=2$	$p=4/3$	$p=8/7$	**FMTL$_{kl}$**
5	40.5±0.9	42.0±1.4	41.2±1.3	41.5±1.1	41.6±1.3	41.6±1.2	**44.1±1.3**	**44.1±1.1**	44.0±1.2	**44.1±1.3**
50	55.0±0.4	57.0±0.2	54.8±0.3	55.1±0.2	55.6±0.3	55.1±0.3	**58.6±0.1**	58.5±0.1	**58.6±0.2**	58.4±0.1

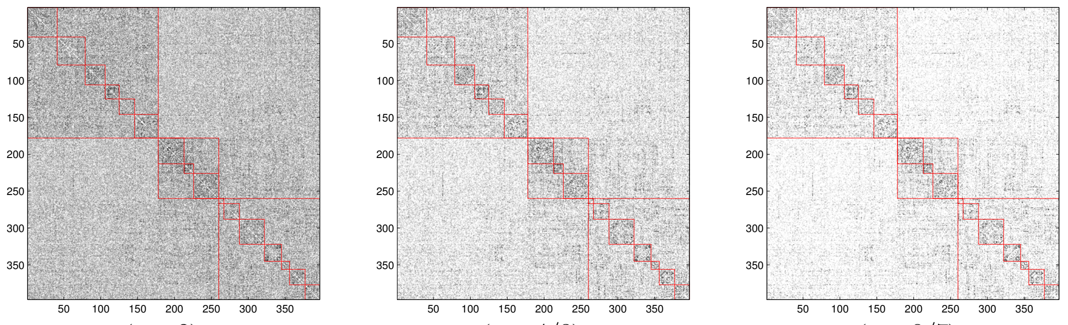


Figure 2: Plots of matrices $\log(1 + |\Theta|)$ (rescaled to [0,1] and diagonal entries removed since they reflect high similarity of a task with itself, which is obvious) computed by our solver FMTL$_p$-S for the SUN397 data set for different p-norms, with cross-validated hyper-parameter values. The hierarchical block structure indicated by the red squares corresponds to the groups of classes available in SUN397, e.g., the top 3 super-classes are *indoor*, *outdoor-natural*, and *outdoor-man-made*, which in turn contain subgroups of classes. Note that this information was not used in experiments. We can observe that the learned Θ matrix at $p = 2$ depicts much more spurious task relationships than the one at $p = 8/7$. Thus, our sparsifying regularizer improves interpretability. Best viewed in color.

size of the receptive field of that network. The parameters were set by 2-fold cross-validation. The results are tabulated in Table 5.

Figure 2 offers a qualitative assessment of the proposed method by showing the output kernel matrices Θ computed by our formulation FMTL$_p$-S for various p-norms. We can observe that the Θ matrix becomes sparser as the p-norm decreases from 2 towards one. Enforcing sparsity helps to detect the hierarchical structure of the tasks (see caption for more details).

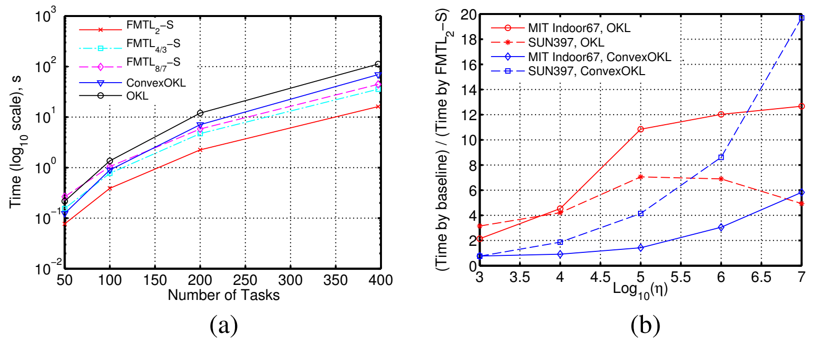
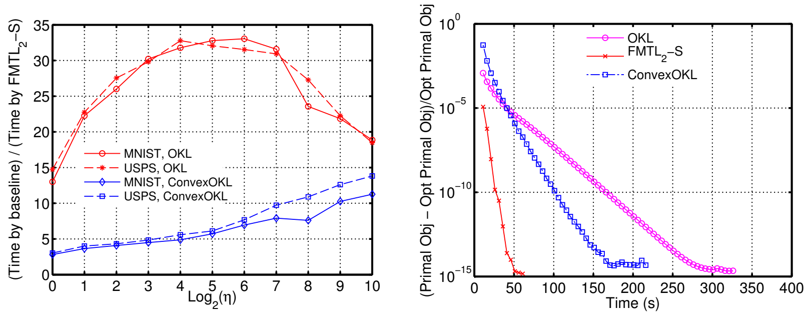


Figure 3: (a) Plot compares the runtime of various algorithms with varying number of tasks on SUN397. Our approach FMTL$_2$-S is 7 times faster that OKL [17] and 4.3 times faster than ConvexOKL [18] when the number of tasks is maximum. It can be observed that FMTL$_2$-S also has the best computational complexity in terms of number of tasks. (b) Plot showing the factor by which FMTL$_2$-S outperforms OKL and ConvexOKL over the hyper-parameter range on various data sets. On SUN397, we outperform OKL and ConvexOKL by factors of 5.2 and 7 respectively. On MIT Indoor67, we are better than OKL and ConvexOKL by factors of 8.4 and 2.4 respectively.



(a) (b)

Figure 4: (a)Plot showing the factor by which FMTL$_2$-S outperforms OKL and ConvexOKL over the hyper-parameter range on MNIST and USPS data sets. On MNIST, we outperform OKL and ConvexOKL by factors of 25.5 and 6.3 respectively. On USPS, we are better than OKL and ConvexOKL by factors of 26.2 and 7.4 respectively. (b) Plot comparing the average rate at which the three algorithms achieve the optimal primal objective (on SUN397 data set). FMTL$_2$-S was run with a duality gap of 10^{-15} and its primal objective value was taken to be the optimal primal objective.

5.3 Scaling Experiment

We compare the runtime of our solver for FMTL$_2$-S with the OKL solver of [17] and the ConvexOKL solver of [18] on several data sets. All the three methods solve the same optimization problem. Figure 3a shows the result of the scaling experiment where we vary the number of tasks (classes). The parameters employed are the ones obtained via cross-validation. Note that both OKL and ConvexOKL algorithms do not have a well defined stopping criterion whereas our approach can easily compute the relative duality gap (set as 10^{-3}). We terminate them when they reach the primal objective value achieved by FMTL$_2$-S . Our optimization approach is 7 times and 4.3 times faster than the alternate minimization based OKL and ConvexOKL, respectively, when the number of tasks is maximal. The generic FMTL$_{p=4/3,8/7}$ are also considerably faster than OKL and ConvexOKL.

Figure 3b compares the average runtime of our FMTL$_p$-S with OKL and ConvexOKL on the cross-validated range of hyper-parameter values. The hyper-parameter value chosen by cross-validation for SUN397 and MIT Indoor67 data sets was around 10^5. FMTL$_p$-S outperform them on both MIT Indoor67 and SUN397 data sets. Figure 4 shows the same comparison on MNIST and USPS data sets.

6 Conclusion

We proposed a novel formulation for learning the positive semi-definite output kernel matrix for multiple tasks. Our main technical contribution is our analysis of a certain class of regularizers on the output kernel matrix where one may drop the positive semi-definite constraint from the optimization problem, but still solve the problem optimally. This leads to a dual formulation that can be efficiently solved using stochastic dual coordinate ascent algorithm. Results on benchmark multi-task and multi-class data sets demonstrates the effectiveness of the proposed multi-task algorithm in terms of runtime as well as generalization accuracy.

Acknowledgments. P.J. and M.H. acknowledge the support by the Cluster of Excellence (MMCI).

References

[1] T. Evgeniou, C. A. Micchelli, and M. Pontil. Learning multiple tasks with kernel methods. *JMLR*, 6:615–637, 2005.

[2] A. Argyriou, T. Evgeniou, and M. Pontil. Convex multi-task feature learning. *ML*, 73:243–272, 2008.

[3] K. Lounici, M. Pontil, A. B. Tsybakov, and S. van de Geer. Taking advantage of sparsity in multi-task learning. In *COLT*, 2009.

[4] A. Jalali, P. Ravikumar, S. Sanghavi, and C. Ruan. A dirty model for multi-task learning. In *NIPS*, 2010.

[5] P. Jawanpuria and J. S. Nath. Multi-task multiple kernel learning. In *SDM*, 2011.

[6] A. Maurer, M. Pontil, and B. Romera-paredes. Sparse coding for multitask and transfer learning. In *ICML*, 2013.

[7] P. Jawanpuria, J. S. Nath, and G. Ramakrishnan. Generalized hierarchical kernel learning. *JMLR*, 16:617–652, 2015.

[8] R. Caruana. Multitask learning. *ML*, 28:41–75, 1997.

[9] Y. Zhang and D. Y. Yeung. A convex formulation for learning task relationships in multi-task learning. In *UAI*, 2010.

[10] Z. Kang, K. Grauman, and F. Sha. Learning with whom to share in multi-task feature learning. In *ICML*, 2011.

[11] P. Jawanpuria and J. S. Nath. A convex feature learning formulation for latent task structure discovery. In *ICML*, 2012.

[12] L. Jacob, F. Bach, and J. P. Vert. Clustered multi-task learning: A convex formulation. In *NIPS*, 2008.

[13] C. A. Micchelli and M. Pontil. Kernels for multitask learning. In *NIPS*, 2005.

[14] A. Caponnetto, C. A. Micchelli, M. Pontil, and Y. Ying. Universal multi-task kernels. *JMLR*, 9:1615–1646, 2008.

[15] M. A. Álvarez, L. Rosasco, and N. D. Lawrence. Kernels for vector-valued functions: a review. *Foundations and Trends in Machine Learning*, 4:195–266, 2012.

[16] T. Evgeniou and M. Pontil. Regularized multi–task learning. In *KDD*, 2004.

[17] F. Dinuzzo, C. S. Ong, P. Gehler, and G. Pillonetto. Learning output kernels with block coordinate descent. In *ICML*, 2011.

[18] C. Ciliberto, Y. Mroueh, T. Poggio, and L. Rosasco. Convex learning of multiple tasks and their structure. In *ICML*, 2015.

[19] S. Shalev-Shwartz and T. Zhang. Stochastic dual coordinate ascent methods for regularized loss. *JMLR*, 14(1):567–599, 2013.

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

[21] M. Hein and O. Bousquet. Kernels, associated structures and generalizations. Technical Report TR-127, Max Planck Institute for Biological Cybernetics, 2004.

[22] A. Ben-Israel and B. Mond. What is invexity ? *J. Austral. Math. Soc. Ser. B*, 28:1–9, 1986.

[23] J. Dattorro. *Convex Optimization and Euclidean Distance Geometry*. Meboo Publishing, 2015. version from 15.4.2015 available at `https://ccrma.stanford.edu/~dattorro/`.

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

[25] R. A. Horn. The theory of infinitely divisible matrices and kernels. *Trans. Amer. Math. Soc.*, 136:269–286, 1969.

[26] F. Hiai. Monotonicity for entrywise functions of matrices. *Linear Algebra and its Applications*, 431(8):1125 – 1146, 2009.

[27] J.-B. Hiriart-Urruty and C. Lemaréchal. *Fundamentals of convex analysis*. Springer, 2001.

[28] Y. Zhang and D. Y. Yeung. Semi-supervised multi-task regression. 2009.

[29] A. Frank and A. Asuncion. UCI machine learning repository, 2010.

[30] Y. Zhang and J. Schneider. Learning multiple tasks with a sparse matrix-normal penalty. In *NIPS*, 2010.

[31] S. Ji and J. Ye. An accelerated gradient method for trace norm minimization. In *ICML*, 2009.

[32] M. Lapin, B. Schiele, and M. Hein. Scalable multitask representation learning for scene classification. In *CVPR*, 2014.

[33] J. J. Hull. A database for handwritten text recognition research. *IEEE PAMI*, 16:550–554, 1994.

[34] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[35] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva. Learning deep features for scene recognition using places database. In *NIPS*, 2014.

[36] M. Koskela and J. Laaksonen. Convolutional network features for scene recognition. In *Proceedings of the ACM International Conference on Multimedia*, 2014.

[37] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba. SUN database: Large-scale scene recognition from abbey to zoo. In *CVPR*, 2010.