Multi-task Multiple Kernel Learning

Pratik Jawanpuria* J. Saketha Nath†

Abstract

This paper presents two novel formulations for learning shared feature representations across multiple tasks. The idea is to pose the problem as that of learning a shared kernel, which is constructed from a given set of base kernels, leading to improved generalization in all the tasks. The first formulation employs a $(l_1, l_p), p \geq 2$ mixed norm regularizer promoting sparse combinations of the base kernels and unequal weightings across tasks — enabling the formulation to work with unequally reliable tasks. While this convex formulation can be solved using a suitable mirror-descent algorithm, it may not learn sparse feature representations which are shared across tasks. The second formulation extends these ideas for learning sparse feature representations constructed from multiple base kernels and shared across multiple tasks. The sparse feature representation learnt by this formulation is essentially a direct product of low-dimensional subspaces lying in the induced feature spaces of few base kernels. The formulation is posed as a $(l_1, l_q), q \geq 1$ mixed Schatten-norm regularized problem. One main contribution of this paper is a novel mirror-descent based algorithm for solving this problem which is not a standard set-up studied in the optimization literature. The proposed formulations can also be understood as generalizations of the framework of multiple kernel learning to the case of multiple tasks and hence are suitable for various learning applications. Simulation results on real-world datasets show that the proposed formulations generalize better than state-of-the-art. The results also illustrate the efficacy of the proposed mirror-descent based algorithms.

Keywords: multi-task feature learning, multiple kernel learning, mirror-descent, Schatten-norm regularization

1 Introduction

The problem of learning a shared feature representation across multiple related tasks (multi-task feature learning) is commonly encountered in many real-world applications. For instance, consider the application of object recognition. Consider each task as that of identifying the images containing a particular object. Note that images of different objects share a common set of features, perhaps those describing lines, shapes, textures etc., which are different from the input features describing the pixels. It is important to discover such shared feature representations leading to improved recognition in all the tasks [1, 2, 3]. Also note that in such applications, the high-level features shared across tasks are far fewer than the low-level input features. Thus it is also natural to seek for sparse feature representations which are shared across multiple tasks. This paper presents two novel formulations which simultaneously learn the shared feature representations as well as the corresponding optimal task predictors. In general, the former learns a non-sparse representation (henceforth denoted by **MK-MTFL**[1]); whereas the latter learns a sparse one (henceforth denoted by **MK-MTSFL**[2]).

The key idea is to pose the problem of multi-task feature learning as that of learning a shared kernel, constructed by combining a given set of base kernels, in order to improve the generalization in case of all the tasks simultaneously. Hence the work can also be understood as an extension of the Multiple Kernel Learning (MKL) framework [4, 5, 6], to the case of multiple tasks. The primary objective in MKL is indeed to learn a kernel best suited for a given task by optimally combining the base kernels.

The proposed **MK-MTFL** formulation employs a mixed $(l_1, l_p), p \geq 2$ norm regularizer over the RKHS norms of the feature loadings corresponding to the given tasks and base kernels. The l_p-norm regularizer is applied across tasks and promotes unequal weightings across task — by varying p one can achieve various schemes of unequal weightings for tasks. This may be useful in handling tasks with unequal reliability [7]. The l_1-norm regularizer is applied over the l_p norms leading to learning the shared kernel as a sparse combination of the given base kernels. The formulation is solved by adapting the mirror-descent [8, 9, 10] based algorithm proposed in [11] to the present case. While this formulation, in general, learns non-sparse feature representations, in some applications like object recognition, learning sparse feature representations may be useful.

In the recent past, several works have focused

*CSE, IIT-Bombay, INDIA. Email: pratik.j@cse.iitb.ac.in
†CSE, IIT-Bombay, INDIA. Email: saketh@cse.iitb.ac.in

[1] Abbreviation denotes multiple kernel multi-task feature learning

[2] Abbreviation denotes multiple kernel multi-task sparse feature learning

on this important problem of learning of sparse feature representations which are shared across multiple tasks [12, 13, 14, 15, 1, 16, 17]. A majority of these methods simultaneously learn a shared low-dimensional (sparse) feature representation as well as the corresponding optimal task predictors. Motivated from both computational as well as regularization perspectives, these methods restrict the search for the optimal feature space to low-dimensional subspaces in the input feature space (i.e., the final feature representation can be obtained from the input features by applying a suitable rotation and then neglecting unimportant features). Hence the quality of the final feature representation depends on the quality of the input features.

The proposed **MK-MTSFL** formulation aims at reducing this risk of employing low quality input features, by considering an enriched input space induced by multiple base kernels. **MK-MTSFL** essentially looks for low-dimensional subspaces in the induced feature spaces of the base kernels which are important for all the tasks at hand, while insisting on employing as few base kernels as possible. The latter constraint helps in eliminating kernels (and corresponding features) with overall low quality across the tasks. In the special case where number of base kernels is unity, the **MK-MTSFL** reduces to the formulation in [1] (henceforth denoted by **MTSFL**) — which is shown to achieve state-of-the-art performance on several benchmark multi-task datasets and hence is considered as a baseline for comparison in the simulations (refer section 5).

The **MK-MTSFL** formulation is posed as a $(l_1, l_q), q \geq 1$ mixed Schatten-norm regularized problem. Such problems are non-standard in literature and call for novel optimization methodologies. One main contribution of this paper is an efficient mirror-descent based algorithm for solving the **MK-MTSFL** formulation. Mirror-Descent (MD) is similar in spirit to the Projected Gradient Descent (PGD) [18, 19] algorithm. While the key computational step in PGD is projection onto the feasibility set, MD employs a suitable regularizer such that the projection problem is simple. MD has been studied in the context of two standard feasibility sets: i) simplex ii) spectrahedron. However the feasibility set with **MK-MTSFL** is a $(l_1, l_q), q \geq 1$ mixed Schatten-norm ball and is hence non-standard. In this paper we show that the entropy function can be employed as the regularizer in the context of MD leading to an efficient algorithm for solving **MK-MTSFL**.

Simulation results on real-world datasets clearly indicate that the proposed methods achieve better generalization than state-of-the-art and hence employing multiple base kernels is beneficial in the case of multiple tasks. The results also show that in the special case where number kernels is unity in the **MK-MTSFL** formulation, the proposed mirror-descent based algorithm converges faster (with similar per-iteration complexity) than the alternate minimization algorithm in [1].

In the subsequent section, we provide a brief introduction to the mirror-descent algorithm. Section 3 details the **MK-MTFL** formulation while in section 4, the details of the **MK-MTSFL** formulation are presented. The sections also describe the mirror-descent based algorithms for solving the formulations. The results of simulations are detailed in section 5. The paper concludes with a brief summary of the paper and directions for future work.

2 Mirror-Descent Algorithm

Mirror-Descent (MD) algorithm is suitable for solving problems of the form $\min_{x \in X} f(x)$ where X is a convex compact set and f is a convex and Lipschitz continuous function on X. It is assumed that an oracle which can compute the sub-gradient (∇f) of f at any point in X is available (an oracle for computing f is not necessary). MD is close in spirit to the projected gradient descent algorithm where the update rule is $x^{(l+1)} = \Pi_X(x^{(l)} - s_l \nabla f(x^{(l)}))$ where Π_X denotes projection onto set X and $x^{(l)}, s_l$ are the iterate value and step-size in the l^{th} iteration respectively. Note that the update rule is equivalent to $x^{(l+1)} = \arg\min_{x \in X} x^\top \nabla f(x^{(l)}) + \frac{1}{2s_l}\|x - x^{(l)}\|_2^2$. The interpretation of this rule is: minimize a local linear approximation of the function while penalizing deviations from current iterate (as the linear approximation is valid only locally). The step-size takes the role of regularization parameter. The key idea in mirror-descent is to choose the regularization term which penalizes deviations from current iterate in such a way that this per-step optimization problem is easy to solve; leading to computationally efficient gradient descent procedures. For convergence guarantees to hold, the regularizer needs to be chosen as a Bregmann divergence i.e., $\frac{1}{2}\|x - x^{(l)}\|_2^2$ is replaced by some Bregmann divergence term: $\omega_{x^{(l)}}(x) = \omega(x) - \omega(x^{(l)}) - \nabla\omega(x^{(l)})^\top (x - x^{(l)})$ where ω is the (strongly convex) generating function of the Bregmann divergence employed. The step-sizes[3] can be chosen as any divergent series for e.g., $s_l = \frac{1}{l^p}, 0 \leq p \leq 1$. Note that in case ω is chosen as $\omega(x) = \frac{1}{2}\|x\|_2^2$, then the projected gradient descent algorithm is recovered. The per-step minimization problem mentioned above can now be re-written in terms of ω as:

$$x^{(l+1)} = \arg\min_{x \in X} x^\top \zeta^{(l)} + \omega(x) \tag{2.1}$$

where $\zeta^{(l)} = s_l \nabla f(x^{(l)}) - \nabla\omega(x^{(l)})$. As mentioned above, the strongly convex function ω is chosen cleverly

[3] Refer [10] for notes on choosing step-sizes optimally.

based on X such that (2.1) turns out to be an easy problem to solve. There are two well-known cases where the MD algorithm almost achieves the information theoretic optimal rates of convergence: i) X is a simplex in $\mathbb{R}^d$ (i.e., $X = \{\mathbf{x} \in \mathbb{R}^d \mid x_i \geq 0,\ \sum_{i=1}^d x_i = 1\}$) and ω is chosen as the negative entropy function: $\omega(\mathbf{x}) = \sum_{i=1}^d x_i \log(x_i)$ ii) X is a spectrahedron (i.e., set of all symmetric positive semi-definite matrices with unit trace) and $\omega(x) = \text{Trace}(x \log(x))$. Infact, in the former case, the per-step problem (2.1) has an analytical solution:

$$x_i^{(l+1)} = \frac{\exp\{-\zeta_i^{(l)}\}}{\sum_{j=1}^d \exp\{-\zeta_j^{(l)}\}} \tag{2.2}$$

Also for this case, the number of iterations can be shown to grow as $log(d)$ and hence nearly-independent of the dimensionality of the problem.

3 Multiple Kernel Multi-task Feature Learning

This section presents the **MK-MTFL** formulation and the mirror-descent based algorithm for solving it. The derivations are presented for the case where each task is a binary classification problem; however it is easy to extend the derivations to other learning tasks as well. Let $\mathcal{D} = \{(\mathbf{x}_{ti}, y_{ti}),\ i = 1, \ldots, m_t,\ t = 1, \ldots, T\}$ be the training dataset where $\mathbf{x}_{ti}$ represents the i^{th} example of the t^{th} task and y_{ti} is its label. Let $K_j, j = 1, \ldots, k$ be the given set of base kernels and let $\phi_j(\cdot)$ represent the feature vector induced by the j^{th} kernel. Let the linear discriminator for the t^{th} task be $\sum_{j=1}^k \mathbf{w}_{tj}^\top \phi_j(\mathbf{x}) - b_t = 0$. Low empirical risk over each task would imply minimizing the following hinge-loss: $\sum_{t=1}^T \sum_{i=1}^{m_t} \max\left(0, 1 - y_{ti}\left(\sum_{j=1}^k \mathbf{w}_{tj}^\top \phi_j(\mathbf{x}_{ti}) - b_t\right)\right)$. We employ the following mixed-norm based regularizer: $\left(\sum_{j=1}^k \left(\sum_{t=1}^T (\|\mathbf{w}_{tj}\|_2)^p\right)^{\frac{1}{p}}\right)^2$ where $p \geq 2$. This regularizer employs a mixed-norm over the RKHS norms (i.e., $\|\mathbf{w}_{tj}\|_2$). More specifically, it involves an l_1 norm across kernels and l_p norm across tasks and hence promotes sparsity across kernels and non-sparse combinations across tasks. With this regularizer, feature loadings for a particular kernel are encouraged to be either zero or non-zero across all the tasks (refer [20, 21] also). Hence the formulation does achieve a shared feature representation as desired. Mathematically, the proposed **MK-MTFL** formulation can be written as:

$$\begin{aligned} \min_{\mathbf{w},b,\xi} \quad & \tfrac{1}{2}\left(\sum_{j=1}^k \left(\sum_{t=1}^T \|\mathbf{w}_{tj}\|_2^p\right)^{\frac{1}{p}}\right)^2 + C\sum_{t=1}^T \sum_{i=1}^{m_t} \xi_{ti} \\ \text{s.t.} \quad & y_{ti}(\textstyle\sum_{j=1}^k \mathbf{w}_{tj}^\top \phi_j(\mathbf{x}_{ti}) - b_t) \geq 1 - \xi_{ti},\ \xi_{ti} \geq 0 \end{aligned}$$

The **MK-MTFL** formulation can also be understood as an extension of the standard MKL formulation [6] to the case of multiple tasks. Infact, in the special case of number of base kernels is unity, it reduces to the MKL formulation. Also, it can understood as an adaptation of the composite absolute penalties family studied in [22, 23, 11] to the current problem. We now rewrite this formulation in a convenient form which can be efficiently solved using mirror-descent based algorithms. We introduce some more notation: let $\Delta_{d,r} = \left\{\mathbf{z} \equiv [z_1 \ldots z_d]^\top \mid \sum_{i=1}^d z_i^r \leq 1, z_i \geq 0, i = 1, \ldots, d\right\}$ and with slight abuse of notation let $\Delta_{d,1} = \Delta_d$. Next we note the following lemma [24]:

LEMMA 3.1. *Let $a_i \geq 0, i = 1, \ldots, d$ and $1 \leq r < \infty$. Then, for $\Delta_{d,r}$ defined as before,*

$$\min_{\eta \in \Delta_{d,r}} \sum_i \frac{a_i}{\eta_i} = \left(\sum_{i=1}^d a_i^{\frac{r}{r+1}}\right)^{\frac{r+1}{r}}$$

and the minimum is attained at

$$\eta_i = \frac{a_i^{\frac{1}{r+1}}}{\left(\sum_{i=1}^d a_i^{\frac{r}{r+1}}\right)^{\frac{1}{r}}}$$

with the convention that $a/0$ is 0 if $a = 0$ and is ∞ if $a \neq 0$.

Using the result of the lemma (with $r = 1$) and introducing variables $\gamma = [\gamma_1 \ldots \gamma_k]^\top$, we have:

$$\left(\sum_{j=1}^k \left(\sum_{t=1}^T (\|\mathbf{w}_{tj}\|_2)^p\right)^{\frac{1}{p}}\right)^2 = \min_{\gamma \in \Delta_k} \sum_{j=1}^k \frac{\left(\sum_{t=1}^T (\|\mathbf{w}_{tj}\|_2)^p\right)^{\frac{2}{p}}}{\gamma_j}$$

Now introducing dual variables $\lambda_j = [\lambda_{j1} \ldots \lambda_{jT}]^\top,\ j = 1, \ldots, k$ and using the notion of dual norm [25], we obtain:

$$\left(\sum_{t=1}^T \left(\|\mathbf{w}_{tj}\|_2^2\right)^{\frac{p}{2}}\right)^{\frac{2}{p}} = \max_{\lambda_j \in \Delta_{T,\bar{p}}} \sum_{t=1}^T \lambda_{jt} \|\mathbf{w}_{tj}\|_2^2$$

where $\bar{p} = \frac{p}{p-2}$. With this, the objective in the **MK-MTFL** formulation can now be written as:

$$\min_{\gamma \in \Delta_k} \min_{\mathbf{w},b,\xi} \max_{\lambda_j \in \Delta_{T,\bar{p}}} \frac{1}{2} \sum_{j=1}^k \frac{\sum_{t=1}^T \lambda_{jt} \|\mathbf{w}_{tj}\|_2^2}{\gamma_j} + C \sum_{t=1}^T \sum_{i=1}^{m_t} \xi_{ti}$$

After interchanging the min. and max. using min-max theorem [26] and writing the Lagrange dual of the resulting formulation wrt. the variables $\mathbf{w}, b, \xi$ alone, leads to the following:

$$\min_{\gamma \in \Delta_k} \max_{\lambda_j \in \Delta_{T,\bar{p}}} \max_{\alpha_t \in S_{m_t}(C)} g(\lambda, \alpha; \gamma) \tag{3.3}$$

where

$$g(\lambda, \alpha; \gamma) = \sum_{t=1}^{T} \left\{ \mathbf{1}^\top \alpha_t - \frac{1}{2} \alpha_t^\top \mathbf{Y}_t \left[\sum_{j=1}^{k} \frac{\gamma_j \mathbf{K}_{tj}}{\lambda_{jt}} \right] \mathbf{Y}_t \alpha_t \right\}$$

Here, α_t is a vector of Lagrange multipliers corresponding to the m_t classifications constraints corresponding to the t^{th} task in the **MK-MTFL** formulation, $S_{m_t}(C) \equiv \left\{ \alpha_t \mid \mathbf{0} \leq \alpha_t \leq C\mathbf{1}, \mathbf{y}_t^\top \alpha_t = 0 \right\}$, $\mathbf{0}, \mathbf{1}$ denote appropriate-sized vectors of zeros and ones respectively, $\mathbf{y}_t$ is vector of labels of the t^{th} task training data points, $\mathbf{Y}_t$ is a diagonal matrix with entries as $\mathbf{y}_t$ and $\mathbf{K}_{tj}$ represents the gram matrix of the t^{th} task training data points wrt. the j^{th} kernel.

The partial dual in (3.3) provides further insights into the formulation. Firstly, (3.3) is equivalent to solving regular SVM [27] problems one for each task with the effective kernel $\left[\sum_{j=1}^{k} \frac{\gamma_j \mathbf{K}_{tj}}{\lambda_{jt}}\right]$. The kernel weights (γ, λ) in the effective kernel are learned based on examples of all the tasks and in particular, γs are shared across all the tasks. Moreover, by construction, most of the γ_j are zero, leading to sparse combination of kernels. In other words, the effective features employed are learnt using examples of all the tasks whereas individual task feature loadings may differ across tasks; this is consistent with the context of multi-task feature learning. Note that in the special case $p = 2$ i.e., $\bar{p} = \infty$, all λs must be unity at optimality. In other words, even the non-zero weights for the selected kernels are also shared across tasks. Infact by varying p one can obtain various weighting schemes for the selected kernels and can be beneficial in cases where tasks are not equally reliable.

In the following we present an efficient mirror-descent based methodology for solving the **MK-MTFL** formulation which is inspired by the algorithm presented in [28]. Instead of solving the min-max problem in (3.3) we prefer to solve equivalently: $\min_{\gamma \in \Delta_k} f(\gamma)$ where $f(\gamma)$ is as defined as $\max_{\lambda_j \in \Delta_{T,\bar{p}}} \max_{\alpha_t \in S_{m_t}(C)} g(\lambda, \alpha; \gamma)$. It is easy to see that $f(\gamma)$ is convex and using Danskin's theorem [19] one can obtain ∇f provided $f(\gamma)$ can be computed efficiently for a given γ. Since the feasibility set for this problem is a simplex, mirror-descent outlined in section 2 can be employed to solve it very efficiently. As noted above, since the feasibility set is a simplex, the MD algorithm will be achieving almost the optimal rate of convergence and specifically, the number of iterations required will be nearly-independent of k (the no. base kernels). Now, evaluation of $f(\gamma)$ is equivalent to maximization of $g(\lambda, \alpha; \gamma)$ wrt. λ and α for the given γ. This maximization can be performed efficiently using an alternate minimization over the λ and α variables. For fixed values of λ, maximization over α is equivalent to solving T regular SVM problems. For fixed values of α the problem of maximization wrt. λ is equivalent to: $-\sum_{j=1}^{k} \min_{\lambda_j \in \Delta_{T,\bar{p}}} \sum_{t=1}^{T} \frac{D_{jt}}{\lambda_{jt}}$ where $D_{jt} = \frac{1}{2} \gamma_j \alpha_t^T \mathbf{Y}_t \mathbf{K}_{tj} \mathbf{Y}_t \alpha_t$. From lemma 1, we know that this problem has an analytical solution. The per-step computational complexity is dominated by the SVM computations and calculation of effective kernel: $O(k \sum_{t=1}^{T} m_t^2)$. As mentioned above, the number of iterations for the mirror-descent grows as $\log(k)$ and in our simulations we found that the alternate minimization typically converges in around 3-5 iterations. Hence the overall computational complexity is $O(k \log(k) \sum_{t=1}^{T} m_t^2)$. This formulation is thus expected to be far more scalable than most of the existing MTFL methodologies [12, 13, 14, 1, 16, 17] — which solve a Singular Value Decomposition (SVD) problem at every iteration. However, in general, unless the base kernels are carefully chosen, the optimal feature representation learnt by **MK-MTFL** is non-sparse. Whereas in some real-world multi-task applications it may be desirable to learn sparse feature-representations. Such a novel formulation for learning sparse feature representations constructed from multiple base kernels and shared across multiple tasks is presented in the subsequent section.

4 Multiple Kernel Multi-task Sparse Feature Learning

This section presents the details of the **MK-MTSFL** formulation. As discussed in section 1, most of the existing methods for shared learning of sparse feature representations look for low-dimensional subspaces in the input space which are beneficial to all the tasks. A short coming of this methodology, as noted above, is the strong dependency of the performance on the input features. One way to minimize risk of employing low quality input features is to enable the formulations working with multiple base kernels. A trivial way of generalizing the existing methodologies to work with multiple base kernels is just by employing a kernel (and hence its induced feature space) which is a simple sum or average of all the base kernels. This essentially is same as working with an enriched input space which is a concatenation of the individual feature spaces corresponding to the given base kernels [29]. This may be is fine provided all the base kernels are "good". In a more realistic situation it is desirable to employ the "best" few base kernels and the corresponding low-dimensional subspaces. In the following we present a formalism which implements this idea. Also, the simulation results on real-world datasets, detailed in section 5, confirm that the proposed **MK-MTSFL** formulation achieves better generalization than the trivial extension noted above.

We follow the notation introduced in section 3. As in case of the existing multi-task sparse feature learning algorithms and in particular **MTSFL**, we construct new features as orthogonal transforms of the given features i.e., $\mathbf{L}_j\phi_j(\mathbf{x})$ where $\mathbf{L}_j$ is an orthogonal matrix which is to be learnt. Now a linear discriminator for the t^{th} task with the newly constructed features is: $\sum_{j=1}^{k}\mathbf{w}_{tj}^\top\mathbf{L}_j^\top\phi_j(\mathbf{x}) - b_t = 0$. Again, low empirical risk over each task would imply minimizing the following hinge-loss: $\sum_{t=1}^{T}\sum_{i=1}^{m_t}\max\left(0, 1 - y_{ti}\left(\sum_{j=1}^{k}\mathbf{w}_{tj}^\top\mathbf{L}_j^\top\phi_j(\mathbf{x}_{ti}) - b_t\right)\right)$. Before describing the regularization term, we introduce some more notation: Let the entries of $\mathbf{w}_{tj}$ be $w_{tjl}, l = 1,\ldots,d_j$ where d_j is the dimensionality of the feature space induced by the j^{th} kernel. By $\mathbf{w}_{\cdot jl}$ we denote the vector with entries $w_{tjl}, t = 1,\ldots,T$. The regularization term we employ is $\left(\sum_{j=1}^{k}\left(\sum_{l=1}^{d_j}\|\mathbf{w}_{\cdot jl}\|_2\right)^q\right)^{\frac{2}{q}}, q \in [1,2]$. Though this regularizer looks similar to that in **MK-MTFL**, there are few fundamental differences: i) this regularizer employs an l_1 norm over feature loadings across tasks rather than an RKHS norm over feature loadings for each task. Hence the feature loadings will be sparse and the features not selected are common across the tasks. Thus a shared sparse feature representation is constructed ii) the l_1 norm over kernels in the former case is now generalized to an l_q norm over kernels. Since $q \in [1,2]$, we obtain various schemes of sparsely combining the base kernels by varying q. iii) the l_p norm over tasks is here restricted to l_2-norm. This is done to facilitate the kernel trick (as we shall understand later). In summary, the regularizer promotes use of few kernels (i.e., few kernel's feature loadings are non-zero) and promotes use of few learnt features in each kernel (i.e., achieves sparse feature representation) across the tasks. Thus it is suitable for constructing shared sparse feature representation across tasks given multiple base kernels. Mathematically, the **MK-MTSFL** formulation can be expressed as:

$$\begin{aligned}\min_{\mathbf{w},b,\xi,\mathbf{L}} \quad & \tfrac{1}{2}\left(\sum_{j=1}^{k}\left(\sum_{l=1}^{d_j}\|\mathbf{w}_{\cdot jl}\|_2\right)^q\right)^{\frac{2}{q}} + C\sum_{t=1}^{T}\sum_{i=1}^{m_t}\xi_{ti} \\ \text{s.t.} \quad & y_{ti}(\textstyle\sum_{j=1}^{k}\mathbf{w}_{tj}^\top\mathbf{L}_j^\top\phi_j(\mathbf{x}_{ti}) - b_t) \geq 1 - \xi_{ti} \\ & \xi_{ti} \geq 0, \mathbf{L}_j \in O^{d_j}\end{aligned}$$

where O^{d_j} represents the set of all orthogonal matrices of dimensionality d_j. Note that, in the special case $k = 1$, this formulation is same as the **MTSFL** formulation. In the following text we re-write this formulation in a form which is convenient to solve using an MD based algorithm.

Using lemma 1 and introducing new variables $\lambda = [\lambda_1 \ldots \lambda_k]^\top$, we have:

$$\left(\sum_{j=1}^{k}\left(\sum_{l=1}^{d_j}\|\mathbf{w}_{\cdot jl}\|_2\right)^q\right)^{\frac{2}{q}} = \min_{\lambda\in\Delta_{k,\bar{q}}}\sum_{j=1}^{k}\frac{\left(\sum_{l=1}^{d_j}\|\mathbf{w}_{\cdot jl}\|_2\right)^2}{\lambda_j}$$

where $\bar{q} = \frac{q}{2-q}$. Again using lemma 1 and introducing new variables $\gamma_j = [\gamma_{j1}\ldots\gamma_{jd_j}]^\top, j = 1,\ldots,k$, the regularizer can be written as: $\min_{\lambda\in\Delta_{k,\bar{q}}}\min_{\gamma_j\in\Delta_{d_j}}\sum_{t=1}^{T}\sum_{j=1}^{k}\sum_{l=1}^{d_j}\frac{w_{tjl}^2}{\gamma_{jk}\lambda_j}$. Now we perform a change of variables: $\frac{w_{tjl}}{\sqrt{\gamma_{jk}\lambda_j}} = \bar{w}_{tjl}$. Also, we define $\bar{\mathbf{w}}_{tj}$ as vector with entries as $\bar{w}_{tjl}, l = 1,\ldots,d_j$. Using this one can re-write the **MK-MTSFL** formulation as:

$$\begin{aligned}\min_{\lambda,\gamma_j,\mathbf{L}_j} \quad & \textstyle\sum_{t=1}^{T}\min_{\bar{\mathbf{w}}_t,b_t,\xi_t}\frac{1}{2}\sum_{j=1}^{k}\bar{\mathbf{w}}_{tj}^\top\bar{\mathbf{w}}_{tj} + C\sum_{i=1}^{m_t}\xi_{ti} \\ \text{s.t.} \quad & y_{ti}(\textstyle\sum_{j=1}^{k}\bar{\mathbf{w}}_{tj}^\top\Lambda_j^{\frac{1}{2}}\mathbf{L}_j^\top\phi_j(\mathbf{x}_{ti}) - b_t) \geq 1 - \xi_{ti},\ \xi_{ti} \geq 0 \\ & \lambda \in \Delta_{k,\bar{q}}, \gamma_j \in \Delta_{d_j}, \mathbf{L}_j \in O^{d_j}\end{aligned}$$

where Λ_j is a diagonal matrix with entries as $\lambda_j\gamma_{jl}, l = 1,\ldots,d_j$[4]. Now writing the Lagrange dual wrt. $\bar{\mathbf{w}}_t, b_t, \xi_t$ leads to the following form:

$$\begin{aligned}\min_{\lambda,\gamma_j,\mathbf{L}_j} \quad & \textstyle\sum_{t=1}^{T}\max_{\alpha_t}\mathbf{1}^\top\alpha_t - \frac{1}{2}\alpha_t^\top\mathbf{Y}_t\left(\sum_{j=1}^{k}\Phi_{tj}^\top\mathbf{L}_j^\top\Lambda_j\mathbf{L}_j\Phi_{tj}\right)\mathbf{Y}_t\alpha_t \\ \text{s.t.} \quad & \lambda \in \Delta_{k,\bar{q}}, \gamma_j \in \Delta_{d_j}, \mathbf{L}_j \in O^{d_j}, \alpha_t \in S_{m_t}(C)\end{aligned}$$

where Φ_{tj} is the data matrix with columns as $\phi_j(\mathbf{x}_{ti}), i = 1,\ldots,m_t$. Denoting $\mathbf{L}_j^\top\Lambda_j\mathbf{L}_j$ by $\bar{\mathbf{Q}}_j$ and eliminating variables $\lambda, \gamma, \mathbf{L}$s leads to:

$$\begin{aligned}\min_{\bar{\mathbf{Q}}}\sum_{t=1}^{T}\max_{\alpha_t\in S_{m_t}(C)} \quad & \mathbf{1}^\top\alpha_t - \tfrac{1}{2}\alpha_t^\top\mathbf{Y}_t\left(\textstyle\sum_{j=1}^{k}\Phi_{tj}^\top\bar{\mathbf{Q}}_j\Phi_{tj}\right)\mathbf{Y}_t\alpha_t \\ \text{s.t.} \quad & \bar{\mathbf{Q}}_j \succeq 0, \textstyle\sum_{j=1}^{k}(trace(\bar{\mathbf{Q}}_j))^{\bar{q}} \leq 1\end{aligned}$$

The difficulty in working with this formulation is that the explicit mappings ϕ_js are required. We now describe a way of overcoming this problem and efficiently kernelizing the formulation (refer [1] also). Let $\Phi_j \equiv [\Phi_{1j}\ldots\Phi_{Tj}]$ and the compact SVD of Φ_j be $\mathbf{U}_j\Sigma_j\mathbf{V}_j^\top$[5]. Now, introduce new variables $\mathbf{Q}_j$ such that $\bar{\mathbf{Q}}_j = \mathbf{U}_j\mathbf{Q}_j\mathbf{U}_j^\top$. Here, $\mathbf{Q}_j$ is a symmetric positive semi-definite matrix of size same as rank of Φ_j. Eliminating

[4]Note that, $\lambda_j\gamma_{jl}$ being zero does not cause a problem since then both w_{tjl} and $\bar{w}_{tjl}$ will be zero for all $t = 1,\ldots,T$ at optimality

[5]Since we perform a compact SVD, Σ_j is a square diagonal matrix of size equal to rank (which is atmost $\sum_{t=1}^{T}m_t$) of Φ_j and no. columns of $\mathbf{U}_j, \mathbf{V}j$ are the again equal to the rank.

variables $\bar{\mathbf{Q}}_j$, we can re-write the above problem using $\mathbf{Q}_j$ as:

$$\min_{\mathbf{Q}} \sum_{t=1}^{T} \max_{\alpha_t} \quad \mathbf{1}^\top \alpha_t - \tfrac{1}{2}\alpha_t^\top \mathbf{Y}_t \left(\textstyle\sum_{j=1}^{k} \mathbf{M}_{tj}^\top \mathbf{Q}_j \mathbf{M}_{tj}\right) \mathbf{Y}_t \alpha_t$$
$$\text{s.t.} \qquad \mathbf{Q}_j \succeq 0, \textstyle\sum_{j=1}^{k} (trace(\mathbf{Q}_j))^{\bar{q}} \leq 1,$$
$$\alpha_t \in S_{m_t}(C)$$

where $\mathbf{M}_{tj} = \Sigma_j^{-1} \mathbf{V}_j^\top \Phi_j^\top \Phi_{tj}$. Note that calculation of $\mathbf{M}_{tj}$ does not require the kernel-induced features explicitly and hence the formulation is kernelized.

The partial dual of **MK-MTSFL** noted above provides more insights into the original formulation. Given $\mathbf{Q}_j$s, the problem is equivalent to solving T SVM problems individually. The $\mathbf{Q}_j$s are learnt using training examples of all the tasks and are shared across the tasks. The trace constraints are a generalization of the trace-norm regularization and hence promote low ranked $\mathbf{Q}_j$s at optimality. Thus the formulation indeed constructs a shared sparse feature representation using multiple base kernels simultaneously. In the following text we describe an efficient MD based algorithm for solving this dual formulation.

Instead of solving the min-max problem in the dual, we prefer to equivalently solve: $\min_{\mathbf{Q} \in R(\bar{q})} f(\mathbf{Q})$ where $\mathbf{Q}$ is a block diagonal matrix with entries as $\mathbf{Q}_1 \dots \mathbf{Q}_k$, $R(\bar{q}) \equiv \left\{ \mathbf{Q} | \mathbf{Q}_j \succeq 0 \ \forall \ j = 1, \dots, k, \sum_{j=1}^{k} (trace(\mathbf{Q}_j))^{\bar{q}} = 1 \right\}$, $f(\mathbf{Q}) = \sum_{t=1}^{T} \mathbf{1}^\top \alpha_t - \frac{1}{2} trace(\mathbf{Q}\mathbf{B})$, and $\mathbf{B}$ is a block diagonal matrix with entries as $\mathbf{B}_j = \sum_{t=1}^{T} \mathbf{M}_{tj} \mathbf{Y}_t \alpha_t \alpha_t^\top \mathbf{Y}_t \mathbf{M}_{tj}^\top$. Note that this minimization problem is an instance of a mixed Schatten-norm based regularized problem. The idea is to solve min. of f using mirror-descent. Note that f is point-wise maximum over affine functions in $\mathbf{Q}$ and hence is convex in $\mathbf{Q}$. Also, the gradient of ∇f wrt. $\mathbf{Q}$ can be obtained using Danskin's theorem: $\nabla f(\mathbf{Q}^{(l)}) = -\frac{1}{2}\mathbf{B}^{(l)}$ where $\mathbf{B}^{(l)}$ is the value obtained using optimal α_t obtained while evaluating $f(\mathbf{Q}^{(l)})$. Also evaluation of $f(\mathbf{Q})$ is equivalent to solving T regular SVM problems.

In the following we show that the strongly convex function $\omega(x) = trace(x \log(x))$ can be employed as the Bregmann divergence generating function in the context of mirror-descent for solving the mixed Schatten-norm regularized problem at hand. With this choice, the per-step optimization problem (2.1) turns out to be:

$$\min_{\mathbf{Q} \in R(\bar{q})} trace(\zeta^{(l)} \mathbf{Q}) + trace(\mathbf{Q} \log(\mathbf{Q}))$$

where $\zeta^{(l)} = s_l \nabla f(\mathbf{Q}^{(l)}) - \nabla \omega(\mathbf{Q}^{(l)})$. We already noted how Danskin's theorem can be employed to obtain $\nabla f(\mathbf{Q}^{(l)})$. Also, $\nabla \omega(\mathbf{Q}^{(l)}) = \log(\mathbf{Q}^{(l)}) + \mathbf{I}$ where $\mathbf{I}$ is the identity matrix of appropriate size. Note that both $\mathbf{Q}$ and $\zeta^{(l)}$ share the same block diagonal structure. In the following we argue that at optimality, $\mathbf{Q}$ and $\zeta^{(l)}$ share the same eigen-vectors (also refer [10] for case of spectrahedron geometry). Let the EVD (Eigen Value Decomposition) of $\zeta^{(l)} = \mathbf{Z}\Pi\mathbf{Z}^\top$ (here Π is the diagonal matrix containing eigen values). Passing from variable $\mathbf{Q}$ to Θ according to $\mathbf{Q} = \mathbf{Z}\Theta\mathbf{Z}^\top$ we can re-write this problem as: $\min_{\Theta \in R(\bar{q})} trace(\Pi\Theta) + trace(\Theta \log(\Theta))$. It is easy to see that the unique optimal solution of this (strongly convex) problem is a diagonal matrix: for every diagonal matrix $\mathbf{D}$ with entries ± 1 and every feasible solution Θ, the quantity $\mathbf{D}\Theta\mathbf{D}$ will remain feasible and moreover achieves the same objective (since Π is diagonal). It follows that the optimal set of solutions must be invariant wrt. $\Theta \rightarrow \mathbf{D}\Theta\mathbf{D}$ transformations, which is possible if and only if Θ is also diagonal (else uniqueness of the solution breaks-down). If the entries in Π, Θ are $\pi_{jl}, \theta_{jl}, l = 1, \dots, r_j, j = 1, \dots, k$ (here r_j represents the dimension of matrix $\mathbf{Q}_j$) respectively, then the above problem is equivalent to:

$$(4.4) \quad \min_{\theta} \quad \textstyle\sum_{j=1}^{k} \sum_{l=1}^{r_j} (\theta_{jl} \log(\theta_{jl}) + \theta_{jl}\pi_{jl})$$
$$\text{s.t.} \quad \theta_{jl} \geq 0 \forall j, \textstyle\sum_{j=1}^{k} (\sum_{l=1}^{r_j} \theta_{jl})^{\bar{q}} \leq 1$$

Though this is a convex problem and involves vectorial variables, the number of variables can be as large as $k \sum_{t=1}^{T} m_t$. Hence it is not wise to employ standard optimization toolboxes to solve this problem. Actually one can reduce the number of variables to k by performing the following trick: introduce variables $\rho_j, j = 1, \dots, k$ and re-write problem (4.4) as:

$$(4.5) \min_{\rho} \quad \sum_{j=1}^{k} \min_{\theta_j} \quad \textstyle\sum_{l=1}^{r_j} (\theta_{jl} \log(\theta_{jl}) + \theta_{jl}\pi_{jl})$$
$$\text{s.t.} \quad \theta_{jl} \geq 0, \textstyle\sum_{l=1}^{r_j} \theta_{jl} = \rho_j$$
$$\text{s.t.} \quad \rho_j \geq 0, \textstyle\sum_{j=1}^{k} \rho_j^{\bar{q}} \leq 1$$

The minimization problem wrt. θ_j in the above problem has an analytical solution: $\theta_{jl} = \rho_j \bar{\pi}_{jl}$, where $\bar{\pi}_{jl} = \frac{\exp\{-\pi_{jl}\}}{\sum_{\bar{l}=1}^{r_j} \exp\{-\pi_{j\bar{l}}\}}$. Using this one can eliminate θ_j from (4.5) and re-write as:

$$\min_{\rho} \quad \textstyle\sum_{j=1}^{k} \left(\rho_j \log(\rho_j) + \rho_j \left(\sum_{l=1}^{r_j} (\pi_{jl}\bar{\pi}_{jl} + \bar{\pi}_{jl} \log(\pi_{jl}))\right)\right)$$
$$\text{s.t.} \quad \rho_j \geq 0, \textstyle\sum_{j=1}^{k} \rho_j^{\bar{q}} \leq 1$$

The size of this problem is k and can easily managed by standard solvers like `cvx`[6]. Apart from the com-

[6] Available at `http://cvxr.com/cvx/`

Table 1: Comparison of % explained variance with various methods on multi-task regression datasets

MTSFL	**MTSFL**$_{\text{avg}}$	**MKL**	**SVM**	**MK-MTFL**			MK-MTSFL		
				$p = 2$	$p = 6$	$p = 8.67$	$q = 1.01$	$q = 1.5$	$q = 1.99$
School									
13.88	12.35	-15.42	-8.97	-9.18	-6.60	-4.53	**14.02***	13.86	13.95
Sarcos									
-23.2839	-40.1259	-86.2684	-26.1611	-82.58	-71.70	-69.9437	**26.1477***	25.6644	25.52
Parkinsons									
57.81	42.72	14.16	44.35	-249.32	-455.36	-2156.34	45.37	58.11	**59.27***

putational cost of `cvx` (which is negligible for k in order of few hundreds), the dominant computations are i) EVD of $\zeta^{(l)}$ which involves EVDs of k matrices of size $r_j, j = 1, \ldots, k$ ii) solving T regular SVMs[7]. This is still reasonable as, in the special case number of base kernels is unity, `cvx` need not be employed and the dominant computation remains the same as that in the alternating minimization algorithm in [1]. Also, in the case $q = 1$, `cvx` need not be employed as the final problem again has an analytic solution.

5 Numerical experiments

This section summarizes results of simulations which illustrate the merits of the proposed algorithms. The experiments are aimed to show that: i) the proposed formulations achieve good generalization ii) the proposed MD based algorithms are efficient in solving them. We begin with results comparing the generalization of the various formulations[8]:

MK-MTFL The multiple kernel multi-task feature learning formulation presented in section (3). Three different values of p (the norm over tasks) were considered: 2, 6, 8.67.

MK-MTSFL The multiple kernel multi-task sparse feature learning formulation presented in section (4). Three different values of q (the norm over kernels) were considered: 1.01, 1.5, 1.99

MTSFL State-of-the-art multi-task sparse feature learning formulation [1]. The original code provided by authors[9] was employed for solving the formulation.

MTSFL$_{\text{avg}}$ Straight-forward extension of the **MTSFL** formulation to the case of multiple base kernels noted in section 4. It is same as the **MTSFL** formulation but with input kernel as the average of all the given base kernels.

SVM The baseline formulation where each task is learnt using an individual SVM [27] model. In case of binary classification tasks, we employed `libsvm`[10] for solving the SVM problem.

MKL The baseline formulation where each task is learnt using an individual MKL [6] formulation. Since this is a special case of **MK-MTFL** with $k = 1$, it can be solved using the same algorithm.

The following datasets were employed in our comparisons:

School A benchmark multi-task regression dataset [1][11]. The goal is to predict performance of students given their descriptions/past record. Data for 15362 students from 139 schools is available and each student is described using 28 features. The regression problem of predicting performances in each school is considered as a task. At random 15 examples in each task were taken as training data and the rest as test data[12].

Sarcos A multiple-output regression dataset [30][13]. The objective is to predict inverse dynamics corresponding to the seven degrees-of-freedom of a `SARCOS` anthropomorphic robot arm based on 21 input features. The dataset comprises of 48933 data points. Prediction of inverse dynamics for each degree-of-freedom is considered as a task. 2000 random examples were sampled in case of each task and 15 of them were used as training examples while the rest were kept aside as test examples.

[7] The computation of $\log(\mathbf{Q}_j)$ can be done efficiently (avoiding EVD) by book-keeping the EVD of $\mathbf{Q}_j$

[8] Code for the proposed formulations is available at: `http://www.cse.iitb.ac.in/saketh/research/MTFL.tgz`

[9] Available at `http://ttic.uchicago.edu/~argyriou/code/mtl_feat/mtl_feat.tar`

[10] Code available at `www.csie.ntu.edu.tw/~cjlin/libsvm/`

[11] Available at `http://ttic.uchicago.edu/~argyriou/code/mtl_feat/school_splits.tar`

[12] We employed a different training-test ratio than [1] in order to focus on data scarce regime

[13] Available at `http://www.gaussianprocess.org/gpml/data/`

Parkinsons A multi-task regression dataset[14]. The objective is to predict two Parkinsons disease symptom scores (motor UPDRS, total UPDRS) for patients based on 19 bio-medical features. The original dataset comprises of 5,875 recordings for 42 patients. The regression problem of predicting each symptom score for each patient is considered as a task. Thus the total number of tasks turns out to be $42 \times 2 = 84$. 15 random examples per task were used for training and the rest formed the test data.

Caltech A benchmark multi-class object categorization dataset [31][15]. A collection of images from 102 categories of objects like faces, watches, ants etc., with 30 images per category. Each 1-vs-rest binary classification problem was considered as a task (no. tasks=102). The training-test splits and 25 base kernels are provided with the dataset.

Oxford A benchmark multi-class object categorization dataset [32][16]. Collection of images of 17 varieties of flowers. The number of images per category is 80. Each 1-vs-rest binary classification problem was considered as a task (no. tasks=17). Three predefined training-test splits consisting of 40 training, 20 validation and 20 test examples and seven base kernels are provided with the dataset.

In case of **SVM, MTSFL** a 3-fold nested cross-validation procedure is employed to tune the C and the kernel parameter. For the proposed methods and **MKL, MTSFL**$_{\text{avg}}$, a 3-fold cross-validation procedure is employed to tune the C parameter and for fairness in comparison, the base kernels are chosen to be exactly those provided to the other methods for nested cross-validation. The values of the C parameter considered for the regression datasets are in the set $\{5e-4, 5e-3, \ldots, 5e+2\}$ whereas for the classification datasets they are in set $\{5e-1, 5, \ldots, 5e+5\}$. Also, one linear and six Gaussian kernels, with parameter values in the set $\{1e-2, 1e-1, \ldots, 1e+3\}$, are employed in case of the regression datasets. Following [1], % explained variance [33] is employed as the measure of performance for regression problems. The explained variance per task can be computed as 1 minus the ratio of mean squared error and the variance of the true outputs on the test dataset. Note that a simple regressor which predicts every output as the mean of the outputs in the dataset achieves an explained variance of zero on the dataset. Thus if explained variance is negative a simple mean estimate is better than the corresponding regressor and hence the method is of no use. In general, higher the explained variance, better the method. However in our case we have multiple tasks and we compute the overall explained variance as the mean of explained variances over each task. Hence we can no longer claim that a methodology which achieves an overall negative score is useless. But still, the higher the explained variance, the better the method. In case of object categorization datasets we employ % accuracy as the measure of performance.

The results on regression (object categorization) datasets are summarized in table 1 (3) respectively. We report the mean % explained variance (% accuracy) achieved over 10 random splits (standard splits) for each dataset. The highest explained variance (accuracy) achieved in each dataset is highlighted. Also, if the improvement in explained variance (accuracy) is statistically significant ($\geq 94\%$ confidence with paired t-test), then the result is marked with a $*$. Table 1 clearly shows that the proposed **MK-MTSFL** formulation outperforms every baseline confirming that learning a shared feature representation is indeed beneficial. Moreover the improvement over state-of-the-art multi-task methodology is statistically significant. This illustrates the advantage of employing multiple kernels in the context of multi-task applications. Note that **MTSFL**$_{avg}$ which also employs multiple kernels (in a trivial manner) is infact achieving less generalization than its single kernel counter-part **MTSFL** — this shows the benefit of employing sparse combinations of kernels. The results highlight the importance of solving the **MK-MTSFL** formulation at various values of q (the norm for kernel regularization), and hence the utility of the MD based algorithm, as there seems to be no evident optimal value across datasets. Automatic tuning of q calls for further investigation.

From the results in table 1, the **MK-MTFL** formulation seems to be highly incompetent; whereas **MK-MTSFL**, which learns sparse feature representations, is highly efficient. However, **MK-MTFL** can also be employed for learning sparse feature representations provided suitable base kernels are employed. For instance, if base kernels are derived from individual features, then **MK-MTFL** may indeed learn sparse feature representations. In order to see if the poor performance of **MK-MTFL** is due to inherent limitations in **MK-MTFL** or due to restricted choice of base kernels, we repeated the simulations with base kernels including those derived from individual features. These results are reported in table 2. Due to the large increase in the number of base

[14]Available at `http://archive.ics.uci.edu/ml/datasets/Parkinsons+Telemonitoring`

[15]Available at `http://mkl.ucsd.edu/dataset/ucsd-mit-caltech-101-mkl-dataset`

[16]Available at `http://www.robots.ox.ac.uk/~vgg/data/flowers/17/index.html`

Table 2: Comparison of % explained variance with various methods on multi-task regression datasets with feature-wise kernels

Datasets	**MTSFL**	**MTSFL**$_{\text{avg}}$	**MKL**	**SVM**	**MK-MTFL**			MK-MTSFL
					$p=2$	$p=6$	$p=8.67$	$q=1.5$
School	-	9.76	-3.07	-4999.08	10.95	13.68	**14.35***	-
Sarcos	2.44	37.80	48.72	-14.64	**49.82**	39.23	30.12	38.82
Parkinsons	-	35.98	**60.66***	53.59	-19.58	-15.88	-418.86	-

kernels (increase by number of features times number of original base kernels) some formulations failed to execute with the available resources. This failure is indicated with a $-$ at the appropriate cell in the table.

Interestingly, the performance of **MK-MTFL** is improved by a great magnitude and infact in the case of School and Sarcos datasets it turned out to be the best performing method (with either setting of base kernels). In case of the Parkinsons dataset though the explained variance seems to be low, we observed that the mean squared error (mse) is the least among all the methods and infact the improvement in terms of mse is statistically significant. The results on object categorization are summarized in table 3. Since the training dataset sizes here are of the order of few tens of thousands, the SVD based algorithms failed to execute. Hence we report simulation results with the baselines and **MK-MTFL** alone. The proposed **MK-MTFL** outperforms the baselines confirming that it is indeed capable of discovering efficient shared feature representations in the context of multiple tasks. To summarize, in case large number of base kernels are available, then both from computational and generalization perspective **MK-MTFL** is the obvious choice. When a restricted set of base kernels are available, then **MK-MTSFL** is the best option.

The results showing the efficacy of the proposed MD methodologies are summarized in figure 1. The first two plots in figure 1 show the scaling of **MK-MTFL** with T and k respectively on the School dataset. The plots show that the method can easily scale to large number of base kernels as well as tasks. The subsequent plots compare convergence rate of the MD algorithm presented in section 4 for the special case $k = 1$ and alternate minimization in [1] on the School dataset with $T = 20,139$ respectively. It can be seen that the MD algorithm achieves faster convergence (with similar computational complexity) especially for large problems.

6 Conclusions

There are three main contributions in this paper: i) the **MK-MTFL** formulation: discovers shared feature representations by learning a common kernel constructed using multiple base kernels. Simulations show that this method is scalable as well as achieves good generalization when provided with an appropriate set of base kernels. ii) the **MK-MTSFL** formulation: learns sparse feature representations constructed from multiple base kernels and shared across multiple tasks. Simulations show that this method always achieves better generalization than state-of-the-art and outperforms various baselines. iii) the MD based algorithm for solving **MK-MTSFL**. We showed that the negative entropy function can be employed as an efficient Bregmann generating function in the context of mirror-descent for solving some mixed Schatten-norm regularized problems.

Table 3: Comparison of performance on various object categorization datasets

Dataset	**MKL**	**SVM**	**MK-MTFL**		
			$p=2$	$p=6$	$p=8.67$
Caltech	52.75	54.35	65.12	65.15	**65.27***
Oxford	83.73	81.67	85.29	**85.98***	85.88

References

[1] A. Argyriou, T. Evgeniou, and A. Pontil. Convex Multi-task Feature Learning. *Machine Learning*, 73:243–272, 2008.

[2] B. Heisele, T. Serre, M. Pontil, T. Vetter, and T. Poggio. Categorization by Learning and Combining Object Parts. In *Advances in Neural Information Processing Systems*, volume 14, pages 1239–1245, 2002.

[3] A. Torralba, K. P. Murphy, and W. T. Freeman. Sharing Features: Efficient Boosting Procedures for Multiclass Object Detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 762–769, 2004.

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

[5] F. Bach, G. R. G. Lanckriet, and M. I. Jordan. Multiple Kernel Learning, Conic Duality, and the SMO

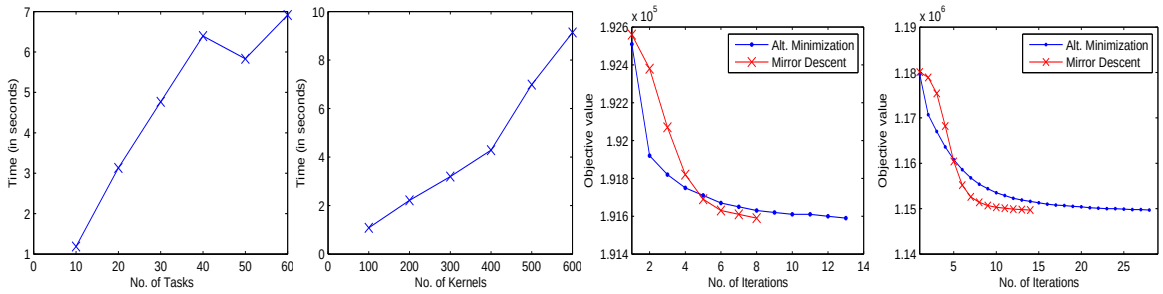

Figure 1: First two plots present scaling results with **MK-MTFL**. The next two compare convergence of MD and alt.min. with $T = 20, 139$ respectively. All comparisons on School dataset

Algorithm. In *International Conference on Machine Learning*, 2004.

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

[7] Xiaolin Yang, Seyoung Kim, and Eric P. Xing. Heterogeneous Multitask Learning with Joint Sparsity Constraints. In *Advances in Neural Information Processing Systems*, 2009.

[8] A.Ben-Tal, T.Margalit, and A.Nemirovski. The Ordered Subsets Mirror Descent Optimization Method with Applications to Tomography. *SIAM J. of Opt.*, 12(1):79–108, 2001.

[9] Amir Beck and Marc Teboulle. Mirror Descent and Nonlinear Projected Subgradient Methods for Convex Optimization. *Operations Research Letters*, 31:167–175, 2003.

[10] Arkadi Nemirovski. Lectures on modern convex optimization (chp.5.4-5.5). Available at `http://www2.isye.gatech.edu/~nemirovs/Lect_ModConvOpt.pdf`, 2005.

[11] J. Saketha Nath, G. Dinesh, S. Raman, C. Bhattacharyya, A. Ben-Tal, and K. R. Ramakrishnan. On the Algorithmics and Applications of a Mixed-norm based Kernel Learning Formulation. In *Advances in Neural Information Processing Systems*, 2009.

[12] R. K. Ando and T. Zhang. A Framework for Learning Predictive Structures from Multiple Tasks and Unlabeled Data. *Journal of Machine Learning Research*, 6:1817–1853, 2005.

[13] Jianhui Chen, Lei Tang, Jun Liu, and Jieping Ye. A Convex Formulation for Learning Shared Structures from Multiple Tasks. In *Proceedings of the International Conference on Machine Learning*, 2009.

[14] Y. Amit, M. Fink, N. Srebro, and S. Ullman. Uncovering Shared Structures in Multiclass Classification. In *International Conference on Machine Learning*, pages 17–24, 2007.

[15] Jian Zhang, Zoubin Ghahramani, and Yiming Yang. Flexible Latent Variable Models for Multi-Task Learning. *Machine Learning*, 73(3):221–242, 2008.

[16] A. Argyriou, C. A. Micchelli, M. Pontil, and Y. Ying. A Spectral Regularization Framework for Multi-task Structure Learning. In *Advances in Neural Information Processing Systems*, 2007.

[17] Laurent Jacob, Francis Bach, and Jean-Philippe Vert. Clustered Multi-Task Learning: A Convex Formulation. In *Advances in Nueral Information Processing Systems*, 2008.

[18] A. Ben-Tal and A. Nemirovski. Lectures on Modern Convex Optimization: Analysis, Algorithms and Engineering Applications. *MPS/ SIAM Series on Optimization*, 1, 2001.

[19] D. Bertsekas. *Non-linear Programming*. Athena Scientific, 1999.

[20] F. Bach. Exploring Large Feature Spaces with Hierarchical Multiple Kernel Learning. In *Advances in Neural Information Processing Systems (NIPS)*, 2008.

[21] R. Jenatton, J.-Y. Audibert, and F. Bach. Structured Variable Selection with Sparsity-inducing Norms. Technical report, 2009.

[22] P. Zhao, G. Rocha, and B. Yu. Grouped and Hierarchical Model Selection through Composite Absolute Penalties. *Annals of Statistics*, 37(6A):3468–3497, 2009.

[23] M. Szafranski, Y. Grandvalet, and A. Rakotomamonjy. Composite Kernel Learning. In *Proceedings of the Twenty-fifth International Conference on Machine Learning (ICML)*, 2008.

[24] Charles A. Micchelli and Massimiliano Pontil. Learning the Kernel Function via Regularization. *Journal of Machine Learning Research*, 6:1099–1125, 2005.

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

[26] R. T. Rockafellar. *Convex Analysis*. Princeton University Press, 1970.

[27] Vladimir Vapnik. *Statistical Learning Theory*. Wiley-Interscience, 1998.

[28] J. Saketha Nath, G. Dinesh, S. Raman, C. Bhattacharyya, A. Ben-Tal, and K. R. Ramakrishnan. On the Algorithmics and Applications of a Mixed-norm Regularization based Kernel Learning Formulation. In *Advances in Neural Information Processing Systems (NIPS)*, 2009.

[29] Bernhard Schölkopf and Alex Smola. *Learning with Kernels*. MIT press, Cambridge, 2002.

[30] Yu Zhang and Dit-Yan Yeung. Semi-Supervised Multi-Task Regression. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 617–631, 2009.

[31] R. Fergus L. Fei-Fei and P. Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In *IEEE. CVPR 2004, Workshop on Generative-Model Based Vision.*, 2004.

[32] M-E. Nilsback and A. Zisserman. A visual vocabulary for flower classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2006.

[33] B. Bakker and T. Heskes. Task clustering and gating for bayesian multi-task learning. *Journal of Machine Learning Research*, 4:83–99, 2003.