Multi-task Learning by Maximizing Statistical Dependence

Youssef A. Mejjati
University of Bath

Darren Cosker
University of Bath

Kwang In Kim
University of Bath

Abstract

We present a new multi-task learning (MTL) approach that can be applied to multiple heterogeneous task estimators. Our motivation is that the best task estimator could change depending on the task itself. For example, we may have a deep neural network for the first task and a Gaussian process for the second task. Classical MTL approaches cannot handle this case, as they require the same model or even the same parameter types for all tasks. We tackle this by considering task-specific estimators as random variables. Then, the task relationships are discovered by measuring the statistical dependence *between each pair of random variables. By doing so, our model is independent of the parametric nature of each task, and is even agnostic to the existence of such parametric formulation. We compare our algorithm with existing MTL approaches on challenging real world ranking and regression datasets, and show that our approach achieves comparable or better performance without knowing the parametric form.*

1. Introduction

Many real world visual learning applications involve learning several tasks. Instead of learning them separately, multi-task learning (MTL) [6] aims to learn them simultaneously with the hope of discovering inherent relationships or latent structures that help improve the overall generalization performance for each task. When applicable, MTL often outperforms single-task learning [3], even when the tasks are seemingly unrelated. This is due to the introduction of regularization techniques that help discover task relationships while avoiding negative transfer in an optimal way. These regularization methods are generally applied to the model parameters, for example, by enforcing a low rank or group sparsity structure. However, what happens if the parameters of the estimator are unknown, or if the parametric form of the estimator is different from one task to another?

Our motivation is that, depending on the task, the best learning strategy and estimator forms may differ. For example, the best estimator for the first task might be a Gaussian process (GP) regression, while for the second task it might be a deep neural network (DNNs). Classical MTL approaches cannot handle this situation as they would require the same estimator architecture or even shared parametric forms for all tasks. Furthermore, once the estimators are parametrized, the task relationships are measured only via these parameterizations. Therefore, strengthening the task relationships in MTL does not directly reflect the underlying data generating probability distributions. In general, such distributions are non-uniform and, therefore, MTL could benefit from selectively emphasizing the task dependence, e.g. in high-density regions in the data space.

We present a new MTL formulation to tackle these limitations. Our approach is agnostic to the parametric formulation of each task; it does not even assume the existence of such a parametric form. Further, it is inherently adaptive to the underlying data distributions. We approach this challenging case by considering the estimator of each task as a random variable and discovering task relationships via the *statistical* dependence estimated by evaluating these estimator random variables on datasets. We use a recently proposed statistically consistent dependence estimator to discover such relationships—the finite set independence criterion (FSIC)—and combine it with a lasso regularizer to enforce sparsity such that the dependencies are maximized only for related tasks. Our algorithm is instantiated as an energy functional that is optimized based on the efficient alternating direction method of multipliers (ADMM) approach.

We test our algorithm on several challenging visual ranking and benchmark regression datasets, and against state-of-the-art MTL techniques. We find that our approach achieves comparable or better performance than existing approaches.

2. Related work

Most existing MTL algorithms can be regarded as instances of an energy minimization framework:

$$\mathcal{O}(\{\mathbf{w}^i\}_{i=1}^L) = \sum_{i=1}^{L} \mathcal{L}^i(\mathbf{w}^i) + \lambda_1 \sum_{i=1}^{L} \mathcal{R}_1(\mathbf{w}^i) + \lambda_2 \mathcal{R}_2(\{\mathbf{w}^i\}_{i=1}^L), \tag{1}$$

where $\mathcal{L}^i$ is the task-specific training loss, e.g., mean classification, regression, or ranking errors on a training dataset, $\mathbf{w}^i$ is the parameter of the i-th estimator f^i, and $\mathcal{R}_1$ and $\mathcal{R}_2$ are regularizers with the corresponding regularization parameters $\lambda_1, \lambda_2 \geq 0$. For instance, for linear estimators, f^i is explicitly represented as $f^i(\mathbf{x}) = \mathbf{x}^\top \mathbf{w}^i$ and the individual regularizer $\mathcal{R}_1$ often takes the squared L^2 parameter norm $\mathcal{R}_1(\mathbf{w}^i) = \|\mathbf{w}^i\|^2$. Different MTL algorithms are specified based on the multi-task regularizer $\mathcal{R}_2$, which characterizes, enforces, and penalizes how different tasks are *related*. For instance, Evgeniou and Pontil's regularized

multi-task learning algorithm (regMTL) [9] assumes that all tasks are related in the sense that the corresponding parameters are *similar* to each other. This can be instantiated by a regularization function that measures the pairwise parameter differences:

$$\mathcal{R}_2(\{\mathbf{w}^i\}_{i=1}^L) = \sum_{i=1}^{L} \left\| \mathbf{w}^i - \sum_{j=1}^{L} \mathbf{w}^j \right\|^2. \quad (2)$$

As one of the first MTL approaches, this algorithm has been successfully applied to the cases when task dependences are known a priori. However, in many practical applications, such full dependence among all tasks is not guaranteed: Some tasks might be related while some others not. In this case, naïvely penalizing all pairwise deviations can deteriorate the performance of some tasks (called negative transfer).

One approach to address negative transfer is to group multiple related tasks. For instance, Jayaraman et al. [12] assume such grouping to be known a priori and enforce attributes from the same group to have features that uniquely describe this group. However, this grouping is not always available and often needs to be discovered. Passos et al. [18] addressed this by assuming that the model parameters of multiple estimators are generated from an underlying factor mixture. This leads to a Bayesian non-parametric approach to discover clusters within tasks. Gupta et al. [11] extended this to hierarchical factor analysis, a paradigm that can model the data from multiple groups into a subspace where some of the discovered bases are shared across the groups while others are specific to a group.

An alternative to clustering is to automatically identify latent task dependence and selectively enforce the dependence based on properly configured regularizers. For example, Argyriou et al.'s multi-task feature learning (MTFL) approach [1] adopts the group sparsity norm $L^{1,2}$ as a regularizer:

$$\mathcal{R}_2(\{\mathbf{w}^i\}_{i=1}^L) = \|A\|_{1,2}, \text{ where } W = UA \quad (3)$$

and $W = [\mathbf{w}^1, ..., \mathbf{w}^L]$. By enforcing the sparsity on the rows of A, this regularizer lets only few columns (*features*) of U contribute to the computation of W. However, by enforcing that all parameters are shared across tasks, this approach could suppress the contribution of some features (in W) that are characteristic to a specific single task, and degrade its final performance. Gong et al. addressed this problem by writing the parameter matrix W as the sum of two matrices P and Q [10], where the $L^{1,2}$ norm is applied to P and Q^T:

$$\mathcal{R}_2(\{\mathbf{w}^i\}_{i=1}^L) = \|P\|_{1,2} + \|Q^T\|_{1,2}, \quad W = P + Q. \quad (4)$$

The first term in the regularizer facilitates discovering a shared low dimensional space across tasks, while the second term identifies outlier tasks by enforcing sparsity on the column space of Q.

More recently, Lee et al.'s asymmetric MTL (AMTL) [16] addressed the negative transfer problem by penalizing a *weighted* pairwise parameter difference:

$$\mathcal{R}_2(\{\mathbf{w}^i\}_{i=1}^L) = \sum_{i=1}^{L} \left\| \mathbf{w}^i - \sum_{j=1}^{L} [B]_{ji} \mathbf{w}^j \right\|^2, \quad (5)$$

where the weight matrix B is trained to represent each estimator parameter based on a *sparse* combination of other estimator parameters. This automatically identifies relevant task groups and disregards outlier tasks.

A similar approach to enforcing the linear combination structure between parameter vectors is to penalize the rank of the parameter matrix W. Argyriou et al. [1] proposed minimizing the nuclear norm $\mathcal{R}_2(\{\mathbf{w}^i\}_{i=1}^L) = \|W\|_*$, which constitutes a tight convex envelop of the rank of W. Again, since not all tasks are related, uniformly penalizing the rank of W can degrade the performance in the presence of outlier tasks. To limit the negative transfer, matrix decomposition tricks similar to Eq. 4 have been applied [8, 7]: Chen et al. introduced a low rank structure and a group sparse structure on P and Q respectively [7], while Chen et al. applied a lasso penalty to Q [8].

While most existing MTL algorithms focus on the regularization based on the parameter matrix W, there are few non-parametric Gaussian Process (GP)-based algorithms that enables MTL without having to share parametric forms [25, 4]. These algorithms build upon the assumption that all tasks are instantiated based on GPs with shared kernels or a single latent kernel, which make those approaches difficult to apply to heterogeneous estimators (e.g., DNNs and GPs). In our experiments, we demonstrate that our algorithm is a competitive alternative to these non-parametric methods.

Our algorithm bypasses the limitations of existing parametric and non-parametric methods by casting individual estimators as random variables and measuring statistical dependence among them. One strongly-related work in this respect is Quadrianto et al.'s algorithm [21], from which our approach is motivated. Quadrianto et al. proposed to estimate the dependence present in multi-task labels based on mutual information. Unfortunately, estimating mutual information directly from continuous data is challenging (see Sec. 3 for details). Therefore, Quadrianto et al.'s algorithm focused on discrete classification problems where the underlying probability distributions are approximated based on discrete histograms. Furthermore, their algorithm assumes full dependence between multiple tasks, and therefore it is vulnerable to negative transfer which we explicitly tackle by introducing sparsity on the dependence estimates. Our use of FSIC does not require the estimation of any probability density, and no prior needs to be introduced. This makes our algorithm applicable to ranking and regression problems where the target variables take continuous values.

3. MTL by dependence maximization

Suppose we are given L different learning tasks $\mathcal{T} = \{T^i\}_{i=1}^L$, where the i-th task T^i is provided with $l(i)$ training data points $\{(\mathbf{x}_k^i, y_k^i)\}_{k=1}^{l(i)} \subset \mathcal{X}^i \times \mathcal{Y}^i \subset \mathbb{R}^{m(i)} \times \mathbb{R}$ sampled from an underlying probability distribution $P_{\mathcal{X}^i\mathcal{Y}^i}$. Our goal is to learn a task estimator set $\mathcal{F} = \{f^i\}_{i=1}^L$ where the i-th member $f^i : \mathcal{X}^i \to \mathcal{Y}^i$ specializes on T^i. Since the best form of the estimator depends on the specific task at hand, we do not assume that all estimators in $\mathcal{F}$ share the same parametric form. Therefore, it is possible that f^1 is a deep neural network (DNN) while f^2 is a Gaussian process (GP) estimator.

Since the parameter vectors of $\{f^i\}$ are not shared, or that finite-dimensional parameter vectors may not even exist (i.e., each member f^i is non-parametric), the classical MTL approach of penalizing the pairwise parameter deviations [9] or sharing a low-dimensional latent parameter vector space [1, 24] is infeasible. Instead, we propose to exploit the potential *statistical dependence* present in $\{f^i\}$: By evaluating f^i on the domain $\mathcal{X}^i$ equipped with the probability distribution $P_{\mathcal{X}^i}$, we can cast f^i into a random variable whose distribution $P_{\mathcal{Y}^i}$ (defined on $\mathcal{Y}^i$) is induced from $P_{\mathcal{X}^i}$.[1]

Measuring estimator dependence. The fundamental assumption of our approach is that some random variables in the class $\mathcal{F}$ (as estimators of $\mathcal{T}$) exhibit statistical dependence. We propose to exploit this to boost the performance of individual estimators. For instance, if we assume that the pair f^i and f^j have strong dependence, then we would expect that the underlying joint distribution $P_{\mathcal{Y}^i\mathcal{Y}^j}$ is significantly different from the marginal products $P_{\mathcal{Y}^i}P_{\mathcal{Y}^j}$. A well-established measure of such a discrepancy is the mutual information:

$$I(f^i,f^j)=\int_{\mathcal{Y}^i\times\mathcal{Y}^j} p_{\mathcal{Y}^i\mathcal{Y}^j}(f^i,f^j)\cdot \log\left(\frac{p_{\mathcal{Y}^i\mathcal{Y}^j}(f^i,f^j)}{p_{\mathcal{Y}^i}(f^i)p_{\mathcal{Y}^j}(f^j)}\right)dy^i dy^j, \tag{6}$$

where $p_{\mathcal{Y}^i\mathcal{Y}^j}$ denotes the joint probability density. The mutual information adopts the Kullback-Leibler divergence as the measure of deviation between the joint distribution $p_{\mathcal{Y}^i\mathcal{Y}^j}$ and the product of marginals $p_{\mathcal{Y}^i}p_{\mathcal{Y}^j}$.

This measure has been previously explored for multi-task learning (MTL). Quadrianto et al. [21] proposed a concave convex procedure that successively approximates the lower bounds on the mutual information constructed as the negation of the joint entropy $H(f^i,f^j)$. Unfortunately, estimating the joint entropy is a challenging problem in general, as it requires calculating the joint density $p_{\mathcal{X}^i\mathcal{X}^j}$. For the case of Quadrianto et al.'s approach, the challenge of estimating the joint density is avoided by 1) assuming the conditional independence of the target output variables y^i and y^j given the input variable $\mathbf{x}$, and 2) approximating the resulting factorized distributions $p(y^i|\mathbf{x})$ and $p(y^j|\mathbf{x})$ using the empirical histograms. This histogram-based approach is applicable only when the outputs variables represent discrete class indices. Therefore, extending it to regression or ranking problems is not straightforward. Furthermore, often even for classification problems, the estimators $\mathcal{F}$ generate continuous class estimates, which provides richer information than the predicted classification indices. For instance, the magnitude of $f^i(\mathbf{x})$ can be considered as confidence of the estimate made at $\mathbf{x}$.

Instead, we use the finite set independence criterion (FSIC): a kernel-based measure of statistical dependence that can be applied directly to continuous random variables and does not require us to estimate the joint density $p_{\mathcal{X}^i\mathcal{X}^j}$ [13]. To our knowledge, we are the first to apply FSIC for MTL.

[1]For exposition simplicity, we assume that each $\mathcal{F}$ member is deterministic.

Suppose that we have a pair of random variables $Q\in\mathcal{Q}\subset\mathbb{R}^{d(q)}$ and $R\in\mathcal{R}\subset\mathbb{R}^{d(r)}$ equipped with the respective marginal distributions P_Q and P_R, as well as their joint distribution P_{QR}. Also, assume that we have positive definite kernels $k:\mathcal{Q}\times\mathcal{Q}\to\mathbb{R}$ and $l:\mathcal{R}\times\mathcal{R}\to\mathbb{R}$ associated with the reproducing kernel Hilbert spaces of functions $\mathcal{H}_Q$ and $\mathcal{H}_R$ defined on Q and R, respectively. Given this structure, the *mean embeddings* of P_Q and P_R into $\mathcal{H}_Q$ and $\mathcal{H}_R$ are respectively defined as the means of the kernel functions with respect to the corresponding distributions: $\mu_Q:=\mathbb{E}_Q[k(\cdot,q)]=\int_{\mathcal{Q}}k(q,\cdot)\mathrm{d}P_Q(q)$ and $\mu_R:=\mathbb{E}_R[l(\cdot,r)]=\int_{\mathcal{R}}l(r,\cdot)\mathrm{d}P_R(r)$. When the kernels k and l are *characteristic* on their respective domains, such as Gaussian kernels:

$$k(q,q')=\exp\left(-\frac{\|q-q'\|^2}{\sigma_k^2}\right),\ l(r,r')=\exp\left(-\frac{\|r-r'\|^2}{\sigma_r^2}\right), \tag{7}$$

the mean embeddings μ_Q and μ_R are injective in their respective domains [22]. Therefore, each distribution can be uniquely characterized in the corresponding RKHS embeddings. The mean embedding of the joint distribution P_{QR} is defined based on the tensor-product $k\otimes l$ of k and l:

$$\mu_{QR}=\int_{\mathcal{Q}\times\mathcal{R}}k(\cdot,q)\otimes l(\cdot,r)\mathrm{d}P_{QR}(q,r). \tag{8}$$

Similarly to mutual information, FSIC measures the statistical dependence of random variables Q and R by quantifying the deviation between their joint distributions and the product of marginals. Exploiting the injectivity of the mean embeddings μ_Q, μ_R, and μ_{QR}, these deviations can be measured based on their mean embeddings. An important advantage of this approach over mutual information is that it does not involve explicitly estimating the respective probability densities as an intermediate step. Therefore, it can be applied to any type of random variable, which facilitates MTL applications to ranking and regression. Specifically, FSIC quantifies these deviations based on an empirical measure ν evaluated over J test locations $S=\{(q_i,r_i)\}_{i=1}^J\subset\mathcal{Q}\times\mathcal{R}$: $\nu=\frac{1}{J}\sum_{i=1}^J\delta_{(q_i,r_i)}$ with $\delta_{(q,r)}$ being the Dirac measure centered on (q,r): The (squared) FSIC $\phi(Q,R)$ of Q and R is defined as the $L^2(\nu)$ distance of the mean embeddings [22]:

$$\begin{aligned}\phi(Q,R)&=\int_{\mathcal{Q}}\int_{\mathcal{R}}(\mu_Q(q)\mu_R(r)-\mu_{QR}(qr))^2\mathrm{d}\nu(q,r)\\&=\frac{1}{J}\|\mathbf{u}\|^2,\end{aligned} \tag{9}$$

where $[\mathbf{u}]_i=\mu_Q(q_i)\mu_R(r_i)-\mu_{QR}(q_ir_i)$. The rationale behind using the test locations S is that 1) even when S is finite, the statistical consistency of FSIC is guaranteed: Jitkrittum et al. [13] has shown that when the kernels $k(\cdot,\cdot)$ and $l(\cdot,\cdot)$ are characteristic, and $k(\cdot,q)$ and $l(\cdot,r)$ constitute analytic functions on $\mathcal{Q}$ and $\mathcal{R}$, respectively for any values of $q\in\mathcal{Q}$ and $r\in\mathcal{R}$, $\phi(Q,R)=0$ if and only if Q and R are independent. Therefore, for such kernels, FSIC provides a proper measure of independence. 2) as shown shortly, using the limited number J of test locations leads to a linear-time sample-based independence estimate (in the number

of data points N). Since Gaussian kernels satisfy the consistency requirements, henceforth we will use Gaussian kernels k and l with the respective width parameters σ_k^2 and σ_l^2 (Eq. 7). For the kernel parameters $\{\sigma_k^2\}$, we use the standard deviation heuristic: $\sigma_k = 4\,\text{mean}(\{\|q_i - q_j\|, 1 \leq i < j, \leq N\})$ for N sampled data points $\{q_i\}_{i=1}^N$.

FSIC-based MTL. Applying FSIC to our estimators $\mathcal{F} = \{f^i\}_{i=1}^L$, we define the FSIC matrix $\Phi(\mathcal{F})$ containing the pairwise FSIC evaluations $[\Phi(\mathcal{F})]_{i,j} = \phi(f^i, f^j)$. In practice, since the distributions $\{P_{\mathcal{X}^i}\}$ are unknown, we cannot explicitly construct random variables $\{f^i\}$. Instead, we evaluate them on a sample $X = X^1 \times \cdots \times X^L = \{(\mathbf{x}_k^1, ..., \mathbf{x}_k^L)\}_{k=1}^N \subset \mathcal{X}^1 \times \cdots \times \mathcal{X}^L$ generated from $P_{\mathcal{X}^1 \times ... \times \mathcal{X}^L}$.[2] In this case, a statistically consistent sample based estimate is obtained as [13]:

$$\hat{\phi}(f^i, f^j) = \frac{1}{J}\|\hat{\mathbf{u}}^{ij}\|^2,$$
$$\hat{\mathbf{u}}^{ij} = \frac{(K^i \circ K^j)\mathbf{1}}{N-1} - \frac{(K^i\mathbf{1}) \circ (K^j\mathbf{1})}{N(N-1)}, \qquad (10)$$

where $\circ$ denotes element-wise product, $\mathbf{1} = [1, ..., 1]^\top$, and $[K^i]_{(j,k)} = k^i(q_j^i, f^i(\mathbf{x}_k))$ for the test locations $\{q_j^i\}_{j=1}^J$ of i-th task.

FSIC and its sample-based estimates are bounded for bounded kernels. For instance, for Gaussian kernels which have an upper bound 1 (Eq. 7), FSIC evaluations are also upper bounded by 1. However, even when bounded kernels are used, the scaling behavior of FSIC varies per random variable pair depending on the choice of kernel parameters (e.g. σ_k^2 in Eq. 7). Therefore, FSICs evaluated over different pairs of random variables will not be directly comparable. This is crucial in our MTL applications, as we will simultaneously evaluate and strengthen FSIC values for all possible estimator pairs. Therefore, we normalize all pairwise FSIC values based on FSIC evaluated on marginal distributions, which suppresses the influence of such scaling variations:

$$[\Phi(F)]_{i,j} = \frac{\hat{\phi}(f^i, f^j)}{\sqrt{\hat{\phi}(f^i, f^i)}\sqrt{\hat{\phi}(f^j, f^j)}}. \qquad (11)$$

The accuracy of FSIC as a measure of (in)dependence relies on the selection of the test locations S. Ideally, these locations should be spread *evenly* over the joint space $\mathcal{Q} \times \mathcal{R}$ with respect to the underlying distribution P_{QR}. Identifying these data points involves a high-dimensional ($d_q + d_r$) non-linear optimization problem [13], which is computationally infeasible to incorporate into our MTL framework. Fortunately, for our MTL problem, all random variables $\{f^i\}$ are one-dimensional and, in this special case, a reasonably good set of test locations can be generated by first scaling each f^i into a unit interval, and then sampling points regularly within this interval. Throughout the entire experiments, we decide the test locations in this way with $J = 100$.

[2]Often in MTL problems, the input spaces $\{\mathcal{X}^i\}$ are assumed to be identical, i.e., $\mathcal{X}^i := \mathcal{X}$. While our algorithm does not require this setting, we adopt this in all of our experiments to facilitate a fair comparison with other algorithms.

Our MTL algorithm strengthens pairwise dependence in $\mathcal{F}$ based on the empirical dependence estimate $\Phi(F)$: Constructing an $N \times T$-sized variable estimator matrix F where the i-th column $F_{(*,i)}$ stores the sample X-based observation of the random variable f^i, our initial algorithm minimizes the energy:

$$\mathcal{O}'(F) = \|F - S\|_F^2 - \lambda_1 \|\Phi(F)\|_F^2, \qquad (12)$$

where the matrix S stores the *initial* estimate constructed by independently solving the tasks $\mathcal{T}$.[3]

In general, not all tasks are related. Therefore, naïvely maximizing all pairwise FSIC (and tuning the hyper-parameter λ_1 accordingly) will not enable us to fully exploit the potential of MTL (Fig. 1). To automatically identify relevant tasks and selectively strengthen the dependence, we enforce the sparsity of $\Phi(F)$:

$$\mathcal{O}(F) = \|F - S\|_F^2 - \lambda_1 \|\Phi(F)\|_F^2 + \lambda_2 \|\Phi(F)\|_1, \qquad (13)$$

where $\lambda_1, \lambda_2 > 0$ are the hyper-parameters, and $\|A\|_F$ and $\|A\|_1$ denote the Frobenius norm and the vectorized L_1-norm ($\|A\|_1 := \|\text{vec}(A)\|_1$), respectively. It should be noted that the second term is the classical L^1-regularizer applied element-wise to the matrix $\Phi(F)$ instead of the nuclear norm $\|\cdot\|_*$ commonly used in MTL: The nuclear norm, as a convex envelop of the rank norm, is typically used to selectively enforce the linear (algebraic) dependence of the (shared) parameter vectors of estimators $\{f^i\}$ (Sec. 2) while our goal is to enforce selectively the statistical dependence.

Note that the combined objective of maximizing L^2 and minimizing L^1 has been previously used in different problems, e.g., in sparse principal component analysis the variance is maximized along each axis on which data are projected and, at the same time, sparsity is enforced to reduce the dimensionality of the relevant latent data space.

Since $\mathcal{O}$ includes a non-differentiable term $\|\Phi(F)\|_1$, it cannot be straightforwardly optimized based on the standard smooth (gradient-based) optimization methods. We minimize $\mathcal{O}$ by adopting the alternating direction method of multipliers (ADMM). First, we define the augmented Lagrangian of $\mathcal{O}$ that decouples $\|\Phi(F)\|_1$ from the other terms in $\mathcal{O}$:

$$L_\rho(F, H, Y) = \|F - S\|_F^2 - \lambda_1 \|\Phi(F)\|_F^2 + \lambda_2 \|H\|_1$$
$$+ \frac{\rho}{2}\|\Phi(F) - H\|_F^2 + \langle Y, \Phi(F) - H\rangle_F, \qquad (14)$$

where $\langle A, B\rangle_F = \text{tr}[A^\top B]$ with $\text{tr}[A]$ being the trace of A. Here, we introduce a constraint $\Phi(F) = H$ connecting $\Phi(F)$ and H, which is instantiated via the penalty term $\|\Phi(F) - H\|_F^2$ and the Lagrangian term $\langle Y, \Phi(F) - H\rangle_F$.

[3]Note that our goal is to construct the evaluation matrix F. An alternative to this approach is to explicitly reconstruct the estimators $\mathcal{F} = \{f^i\}_{i=1}^L$, e.g., by replacing $\|F - S\|_F^2$ in Eq. 12 by the individual training error $\sum_{i=1}^L \sum_{k=1}^N l(f^i(\mathbf{x}_k^i), y_k^i)$ (Eqs. 24-25). This typically requires employing an additional regularizer per task (see Eq. 1). Given that our algorithm has two hyper-parameters (see Eq. 13), tuning additional regularization hyper-parameters is challenging. As a design choice, we construct the initial estimates S by tuning these individual hyper-parameters independently, and penalize the deviation of F from S. When $\mathcal{F}$ needs to be explicitly constructed, it can be *trained* using (X, F) as a regression training data.

Based on $L_\rho(F, H, Y)$, each ADMM iteration can be constructed as updating the estimator variable F and two sets of auxiliary variables H and Y:

$$F^{k+1} := \underset{F}{\operatorname{argmin}} L_\rho(F, H^k, Y^k), \tag{15}$$

$$H^{k+1} := \underset{H}{\operatorname{argmin}} L_\rho(F^{k+1}, H, Y^k), \tag{16}$$

$$Y^{k+1} := Y^k + \rho(\Phi(F^{k+1}) - H^{k+1}). \tag{17}$$

The first update step (Eq. 15) minimizes an auxiliary energy functional:

$$\begin{aligned}\mathcal{C}(F) = &\|F - S\|_F^2 - \lambda_1 \|\Phi(F)\|_F^2 + \frac{\rho}{2}\|\Phi(F) - H^k\|_F^2 \\ &+ \langle Y, \Phi(F) - H^k \rangle_F,\end{aligned} \tag{18}$$

which is a differentiable function of F, and therefore, can be minimized by the gradient descent

$$\begin{aligned}\frac{\partial \mathcal{C}(F)}{\partial [F]_{m,n}} = &\left\langle Y - 2\lambda_1 \Phi(F) + \rho(\Phi(F) - H^k), \frac{\partial \Phi(F)}{\partial [F]_{m,n}} \right\rangle_F \\ &+ 2[F - S]_{m,n}.\end{aligned} \tag{19}$$

Calculating $\frac{\partial \Phi(F)}{\partial [F]_{m,n}}$ requires evaluating the derivative of $\hat{\mathbf{u}}^{ij}$ as an intermediate step (see Eqs. 10-11):

$$\frac{\partial [\hat{\mathbf{u}}^{ij}]_p}{\partial [F]_{(m,n)}} = \begin{cases} 0 & \text{if } i, j \neq n \\ \frac{\left(\frac{\partial [K^i]_{(p,*)}}{\partial [F]_{(m,n)}} \circ [K^j]_{(p,*)}\right)\mathbf{1}}{N-1} & \\ \quad - \frac{\left(\frac{\partial [K^i]_{(p,*)}}{\partial [F]_{(m,n)}}\mathbf{1}\right)([K^j]_{(p,*)}\mathbf{1})}{N(N-1)}, & \text{if } i = n, \end{cases} \tag{20}$$

where for Gaussian kernels, the kernel matrix derivatives are given as:

$$\frac{\partial [K^i]_{(j,k)}}{\partial [F]_{(m,n)}} = \begin{cases} 0 & \text{if } k \neq m \\ \frac{2}{\sigma_i^2}(q_j - f^i(\mathbf{x}_k)) k^i(q_j, f^i(\mathbf{x}_k)) & \text{otherwise.} \end{cases} \tag{21}$$

When $j = n$, the roles of i and j are interchanged in Eq. 20.

The second update step (Eq. 16) can be rewritten as:

$$\begin{aligned} H^{k+1} := &\underset{H}{\operatorname{argmin}} \lambda_2 \|H\|_1 + \frac{\rho}{2}\left\|H - \Phi(F^{k+1})\right\|_F^2 \\ &- \langle Y^k, H \rangle \\ = &\underset{H}{\operatorname{argmin}} \lambda_2 \|H\|_1 + \frac{\rho}{2}\left\|H - \Phi(F^{k+1}) - \frac{1}{\rho} Y^k\right\|_F^2. \end{aligned} \tag{22}$$

This is a typical least squares minimization with L^1 penalty. The minimum H^{k+1} in Eq. 22 can be obtained in a closed form based on the L^1-shrinkage operator [2]:

$$[H^{k+1}]_{i,j} = \max\left([\Phi(F^{k+1}) + 1/\rho Y^k - \lambda_2/\rho]_{i,j}, 0\right). \tag{23}$$

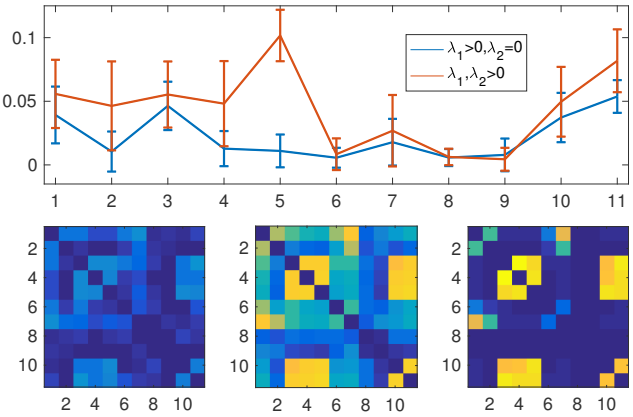

Figure 1. Results of our algorithm for Pubfig dataset (see Sec. 4 for details) with ($\lambda_1, \lambda_2 > 0$) and without ($\lambda_1 > 0, \lambda_2 = 0$) the sparsity regularizer $\|\Phi(F)\|_1$ (Eq. 13). (top) x-axis: target attributes; y-axis: mean accuracy improvements from the baseline independent estimators (see Table 1 for full results). (bottom) FSIC evaluation matrices Φ: (left) the initial FSIC matrix calculated from the baseline independent estimators S, (center) and (right) optimized FSIC matrices with and without the sparsity regularizer. FSIC values increases as the color varies from dark blue to bright red. The diagonal components (originally 1) are set to zero for better visibility.

Discussion. In our objective $\mathcal{O}$, the first regularizer $-\|\Phi(F)\|_F^2$ tends to uniformly strengthen the pairwise dependence. This is useful when all estimators in $\mathcal{F}$ are equally statistically dependent. The second regularizer—the sparsity regularizer $\|\Phi(F)\|_1$—helps to selectively strengthen the dependence addressing the case when not all $\mathcal{F}$ estimators are dependent.

Since practical applications exhibit both cases (as well as the case when no dependence is observed), both regularizers are important. Figure 1 demonstrates the influence of the second regularizer on the MTL performance based on the PubFig dataset (see Sec. 4 for details): Bottom left panel shows the pairwise FSIC values calculated from the initial independent estimators S demonstrating that the task dependence is indeed clustered, as Tasks $\{3,4,5,10,11\}$ show strong mutual dependence forming a cluster. Similarly, Tasks $\{1,7\}$ constitute another cluster. These two clusters are weakly connected via Task 1 and 3 dependence ((1,3)-th entry of the FSIC matrix). Tasks 6, 8 and 9 display no noticeable dependence with any other tasks. Uniformly strengthening dependence (bottom center: $\lambda_2 = 0$) somewhat improves the overall performance.

However, since strengthening the statistical dependence for tasks $\{6, 8, 9\}$ tends to degrade the performance, optimizing its regularization hyper-parameter λ_1 leads to only a moderate level of FSIC strengthening. Enforcing sparsity in FSIC enabled selectively strengthening only the relevant dependences and disregards the outlier tasks, thereby further significantly improving performance (Fig. 1 top).

The run-time of our algorithm depends on the number of total ADMM iterations and the cost of the gradient-based optimization step for F (Eqs. 15,18-19). The complexity of calculating the gradient $\partial \mathcal{C}(F)$ for F-update (Eq. 19) is linear in the numbers of data points N and test locations J, and quadratic in the

number T of tasks: $O(JNT^2)$. We observed in preliminary experiments that the ADMM converges quickly, typically within the first 20 iterations. Therefore, we set the maximum number K of ADMM iterations at 50 throughout the entire experiments: The iteration terminates when $K > 50$ or the reduction of the objective function values is smaller than the tolerance parameter $\epsilon = 10^{-5}$. For the OSR dataset with N=2,688 T=6, a single step of the ADMM optimization process took around 1 sec. Algorithm 1 summarizes the training process.

The convergence analysis of non-linear ADMM is an area of active research. A straightforward way to guarantee the convergence (to a local minimum) is to schedule $\rho \to \infty$. The price for this theoretical guarantee is numerical instability typically observed in the dual decomposition-type optimization methods [5]. Throughout the entire experiments, we fixed ρ at 10 which leaded to a steady decrease of the objective $\mathcal{O}$ values in preliminary experiments.

Algorithm 1 MTL by dependence maximization

Input: Initial estimate S; tolerance threshold ϵ; maximum iteration number K; regularization parameters $\lambda_1, \lambda_2 \geq 0$.

1: **while** $\mathcal{O}(F^k) - \mathcal{O}(F^{k-1}) \leq \epsilon$ (Eq. 13) and $k < K$ **do**
2: Update F^{k+1} (Eq. 18).
3: Update H^{k+1} (Eq. 23).
4: Update Y^{k+1} (Eq. 17).
5: $k = k+1$.
6: **end while**

Output: Optimized estimate F^*.

4. Experiments

Setup. We evaluate the performance of our algorithm on five ranking datasets and two regression datasets. For each dataset, we prepared two baseline estimator classes. The first baseline (Base_1) is constructed from homogeneous support vector machines (SVMs) while for the second baseline (Base_2), deep neural networks (DNNs), Gaussian process (GP) estimators,[4] and SVMs are evaluated and the best estimator was chosen per dataset and per target attribute (based on validation error, discussed shortly) as the baselines.

The **Outdoor Scene Recognition dataset (OSR)** contains 2,688 images of 6 visual target attributes computed from 8 image categories [17]. The **Public Figure Face dataset (PubFig)** contains 800 face images of 11 target attributes computed from 8 individuals [17, 15]. The **Shoes** dataset contains 14,658 images of 10 attributes extracted from 10 shoe categories [14]. We use 512-dimensional GIST descriptors, 542-dimensional features that combine GIST descriptors and color histograms, and 990-dimensional GIST and color histogram combinations for OSR, PubFig, and Shoes, respectively as shared by the authors of [17] and [14]. The **SUN** attribute dataset contains 16,656 scene images with 102 attributes calculated from 707 scene categories [19]. Here, the ground-truth rankings are calculated based on Mechanical Turk voting on the existence of attributes of interest per image.

[4]For ranking problems, we obtained the maximum a posteriori solutions using the rank loss (Eq. 24) as the (inverse) likelihood.

For each image in this dataset, a 4,096-dimensional feature vector is extracted based on VGG19 DNN feature extractor trained on *ImageNet* datasets. We selected 10 target attributes out of 102 that exhibit high pairwise correlations: We evaluated the outputs of individual baseline estimators per attribute and measured the pairwise Pearson correlations. The **Caltech-UCSD Birds (Birds)** dataset provides 11,788 bird images of 200 bird species provided with 312 binary attribute annotations. Each input image is represented based on 1000-dimensional VGG19 features. We selected 10 target attributes as elements of a single cluster formed via spectral task clustering using the FSIC as the similarity measure. Our supplemental provides details of task clustering (and visualization) on this dataset (see Sec. 5 for additional information).

To facilitate a comparison with existing non-parametric GP-based MTL algorithms (originally designed for regression problems), we also use two benchmark regression datasets: The **river flow dataset (RF1)** contains 9,005 instances of 64-dimensional input features that represent the flows in river networks observed in time intervals. The 8 target attributes correspond to the predictions of the flows for 48 hours in the future at specific test locations [23]. The **Energy Building dataset (ENB)** contains 768 instances of 8 measured building parameters as input and the corresponding target heating load and cooling load attributes.

For comparison in ranking problems, we adopt four existing MTL algorithms. Evgeniou and Pontil's **regularized MTL algorithm (regMTL)** [9], Lee et al.'s **Asymmetric MTL algorithm (AMTL)** [16], Argyriou et al.'s multi-task feature learning algorithm (**MTFL**), and Chen et al.'s **Low-Rank MTL algorithm (LRMTL)**. Some of these algorithms were originally developed for classification but adapting them to ranking and regression is straightforward using rank loss: In training, an ordered training pair (i,j) implies that the ranking of $\mathbf{x}_i$ is higher than $\mathbf{x}_j$

$$l_{\text{rank}}(\mathbf{x}_i, \mathbf{x}_j; f) = \max(0, 1 - (f(\mathbf{x}_i) - f(\mathbf{x}_j)))^2. \quad (24)$$

or the squared L^2 regression loss:

$$l_{\text{regr}}(\mathbf{x}_i, y_i; f) = (f(\mathbf{x}_i) - y_j)^2. \quad (25)$$

In the supplemental, we also compare with an adaptation of Pentina et al.'s curriculum learning MTL algorithm [20]. Unlike other MTL algorithms in comparison, applying this algorithm to ranking and regression is not straightforward as it builds upon the estimated bounds for the classification error. Our adaptation is obtained as an algorithmic extension (see supplemental for details).

For regression datasets, we also compare with the non-parametric Gaussian process-based approaches of Bonilla et al. [4] (**GPMTL**) and Titsias and Lázaro-Gredilla [25] (**Spike and slab variational inference: SNS**). Both algorithms adopt fully Bayesian inference and therefore, the underlying hyper-parameters are automatically tuned. We tune only the latent dimensionality of the multiple task kernel functions based on a separate validation set.

The regMTL requires tuning two hyper-parameters: The regularization hyper-parameter for individual baseline estimators and the hyper-parameter for the multi-task regularizer that controls the strength of the task dependence. The LRMTL also has two

Table 1. Ranking performances of different MTL algorithms. Kendall's Tau correlations $\times 100 \pm$ std. $\times 100$ are presented (higher is better). The three best results are highlighted with **boldface blue**, *italic green*, and plain orange fonts, respectively.

Dataset	Target	Base_1	Base_2	regMTL [9]	MTFL [1]	LRMTL [7]	AMTL [16]	Ours
OSR	1	88.26±0.83	90.26±0.55	89.93±0.63	89.09±1.24	*90.67±1.06*	88.34±0.73	**91.95±0.74**
	2	81.30±0.62	*86.01±0.84*	85.84±1.02	84.15±1.13	81.28±1.15	81.47±0.58	**86.33±0.92**
	3	71.00±1.04	*75.01±1.42*	74.52±2.28	73.89±2.62	73.03±1.31	72.42±1.08	**76.30±1.03**
	4	72.39±1.77	*77.66±1.23*	77.41±0.99	75.10±2.08	73.74±2.05	73.35±1.54	**79.01±1.27**
	5	75.52±1.19	79.30±1.08	*79.42±1.20*	79.02±1.71	78.08±0.89	77.44±0.83	**82.52±1.13**
	6	76.12±1.27	80.04±1.73	*80.12±1.59*	78.23±1.30	76.97±1.74	77.15±1.02	**80.55±1.34**
PubFig	1	64.50±2.53	66.40±3.08	60.55±2.59	62.88±2.51	*71.60±1.28*	64.47±2.56	**71.98±2.89**
	2	57.10±3.08	60.07±3.53	53.12±3.39	54.12±4.07	*62.14±4.80*	57.10±3.08	**64.71±3.58**
	3	64.22±2.06	66.53±2.77	63.97±3.00	65.23±4.05	**72.52±1.43**	68.31±1.50	*72.06±2.70*
	4	61.91±1.37	64.33±2.44	63.01±2.00	60.69±2.99	**70.34±2.37**	*69.16±2.04*	69.15±4.53
	5	55.82±3.33	58.48±2.97	54.22±3.64	55.36±3.37	*65.08±1.28*	62.09±3.12	**68.65±2.26**
	6	75.12±1.54	77.34±2.54	74.65±1.53	72.86±3.40	*77.43±2.18*	75.13±1.55	**78.18±3.31**
	7	58.79±3.03	62.66±3.54	57.66±3.74	59.36±3.99	**66.53±1.54**	58.79±3.03	*65.34±4.76*
	8	60.05±1.69	61.91±2.68	60.13±1.31	58.08±2.65	**62.63±2.01**	60.05±1.68	*62.54±2.89*
	9	52.44±2.72	*57.09±3.43*	53.74±3.54	53.78±3.25	56.65±4.92	52.55±2.87	**57.53±3.80**
	10	58.27±3.86	61.51±2.77	59.15±2.18	58.25±3.36	**66.88±1.78**	64.21±3.45	*66.47±3.53*
	11	63.21±1.82	66.81±2.39	64.15±1.36	63.91±3.60	*74.05±1.25*	70.07±1.58	**74.99±1.15**
Shoes	1	68.09±1.47	67.87±0.70	69.08±1.76	68.43±1.06	*69.72±1.31*	68.84±1.90	**71.93±0.91**
	2	56.39±1.84	*60.27±2.71*	59.04±1.72	57.81±2.08	58.02±3.00	57.71±1.88	**60.37±2.52**
	3	30.50±3.65	**34.43±2.66**	32.39±3.60	32.55±3.39	30.55±2.60	31.09±3.47	*34.42±2.61*
	4	46.18±1.63	*48.41±2.31*	46.85±2.11	46.04±2.72	46.24±1.83	45.81±2.10	**48.47±2.26**
	5	61.44±1.99	61.64±1.98	*63.92±1.51*	62.21±2.09	62.77±1.81	63.06±1.40	**64.79±0.99**
	6	61.87±2.30	60.80±3.56	**64.40±2.36**	*62.79±2.31*	62.62±1.81	62.46±1.92	62.06±2.14
	7	52.58±1.51	*56.43±2.02*	53.64±1.65	52.40±2.54	52.58±2.85	52.95±1.28	**57.15±2.45**
	8	49.56±1.73	48.89±0.86	50.34±2.12	**51.98±1.92**	49.68±1.66	50.24±1.30	*50.78±1.50*
	9	61.57±1.97	62.78±2.59	62.07±2.44	*63.34±2.33*	62.89±1.81	61.63±1.90	**67.11±1.12**
	10	66.91±1.16	66.83±2.33	69.16±1.02	68.10±1.56	*69.34±1.43*	68.86±1.91	**72.09±1.08**
SUN	1	66.18±3.15	68.24±4.32	66.52±4.41	68.39±4.39	**72.01±5.65**	65.21±7.56	*70.62±3.35*
	2	71.24±3.11	*75.31±4.06*	75.04±3.08	73.78±5.97	**75.89±1.08**	72.60±4.50	73.74±4.53
	3	76.84±1.16	76.77±1.87	75.74±2.15	75.87±1.90	*77.73±2.07*	77.62±1.37	**78.78±1.82**
	4	79.03±1.21	80.40±1.32	79.19±3.37	79.58±1.52	**84.20±0.40**	79.75±3.30	*82.49±0.87*
	5	79.66±1.42	78.67±2.41	78.43±1.68	78.42±1.04	*80.64±1.63*	80.59±1.62	**80.66±1.55**
	6	80.75±0.81	79.76±1.79	78.78±3.35	79.47±0.95	**82.71±0.90**	81.44±0.79	*82.14±1.04*
	7	*79.76±0.65*	79.41±1.00	79.62±0.88	78.17±1.16	78.76±1.36	79.65±0.90	**80.03±0.81**
	8	83.91±0.53	84.12±0.67	**84.21±0.60**	*84.13±0.60*	83.39±1.46	84.08±0.77	83.83±0.72
	9	**63.34±1.10**	63.13±1.01	61.91±2.30	62.53±0.73	62.67±0.30	*63.23±1.22*	63.14±0.90
	10	*82.23±3.50*	80.85±5.77	78.79±6.76	77.30±7.76	**84.64±2.42**	80.60±5.41	81.31±2.91
Birds	1	56.78±3.06	56.76±3.04	**60.43±3.24**	52.51±5.00	55.07±2.48	*59.38±4.26*	58.70±3.69
	2	42.18±3.24	41.23±4.18	**54.15±4.33**	52.66±3.51	47.88±5.18	53.43±3.09	*53.88±2.46*
	3	57.74±1.84	57.56±2.48	**61.67±2.69**	59.30±2.15	55.45±2.89	60.07±4.44	*60.08±1.53*
	4	46.69±2.74	46.97±2.02	54.21±4.17	51.29±1.60	52.74±5.55	**56.48±1.76**	*54.36±1.29*
	5	49.94±3.95	48.56±6.42	**54.95±3.55**	*53.40±4.39*	51.41±6.44	51.86±6.61	53.16±2.09
	6	41.78±0.91	43.91±3.11	*54.00±3.51*	48.17±5.16	46.14±4.86	**55.75±1.09**	52.31±1.95
	7	46.46±5.68	45.70±5.90	48.99±10.49	49.19±1.43	*51.44±3.38*	50.82±5.46	**51.50±1.03**
	8	56.46±0.84	56.53±2.43	*62.96±2.32*	58.79±3.03	58.56±4.55	**62.99±1.02**	60.66±1.91
	9	49.89±2.23	49.66±2.16	53.92±5.16	53.26±1.64	49.61±3.99	*55.61±2.66*	**55.68±1.15**
	10	54.71±6.15	54.08±6.83	*60.27±4.21*	58.73±2.86	58.46±4.70	59.50±6.81	**60.99±1.70**

hyper-parameters: One for multi-task regularization and the other for outlier elimination. Similarly, AMTL has two parameters. The first parameter controls the overall strength of task dependence (equivalently the amount of transfer between tasks) while the other parameter controls the relative distribution of transfer strength across multiple tasks. MTFL on the other hand has one parameter that controls the number of shared features between tasks. Finally, our algorithm requires tuning two regularization hyper-parameters λ_1 and λ_2, which we show from Fig. 2 to be easily achievable as the performance varies smoothly with respect to changes in hyper-parameter values. For all datasets, we use 200 training and validation data points. All hyper-parameters were tuned based on validation sets. For ranking, pair-wise labels were induced from the labels of training data points. For evaluating the ranking performance, we measured the Kendall's Tau rank correlation coefficient bounded in $[-1,1]$ that counts the number of correct (and incorrect) rank predictions with respect to the number of total rank pairs. For the regression, we measured the mean squared error. All experiments were performed 10 times with different training and validation configurations and the results are averaged.

Table 2. Regression performances of different MTL algorithms. Mean squared error ± std. are presented (lower is better).

Dataset	Target	Base_1	Base_2	regMTL [9]	MTFL [1]	LRMTL [7]	AMTL [16]	GPMTL [4]	SNS [25]	Ours
RF1	1	21.93±17.51	*11.30±2.06*	12.73±0.69	11.71±0.82	11.67±0.64	13.46±4.68	26.28±9.75	15.26±4.68	**11.20±2.13**
	2	0.98±0.36	**0.80±0.20**	0.83±0.14	0.85±0.14	1.05±0.12	1.03±0.30	0.84±0.19	*0.81±0.18*	**0.80±0.20**
	3	23.21±25.28	15.82±1.89	15.82±0.67	*15.19±1.19*	15.38±0.56	16.43±3.52	25.72±8.56	17.09±4.05	**14.63±1.47**
	4	14.98±4.71	13.41±2.28	12.80±0.68	12.63±1.13	*12.58±0.37*	13.06±1.90	16.89±3.97	13.26±2.69	**12.51±1.37**
	5	7.80±0.26	*7.58±0.68*	8.40±0.97	7.80±0.53	7.75±0.25	7.77±0.25	10.47±3.97	8.24±1.16	**7.45±0.66**
	6	2.46±0.09	2.35±0.21	2.53±0.16	2.56±0.14	2.57±0.07	2.54±0.26	**2.28±0.70**	2.48±0.13	*2.32±0.18*
	7	5.88±0.62	*4.85±1.18*	4.90±0.74	4.98±0.81	5.46±0.15	6.05±1.14	7.25±3.35	5.13±1.25	**4.69±1.05**
	8	7.94±8.92	*4.79±0.44*	5.45±0.46	5.27±0.23	5.43±0.12	5.24±0.23	6.52±2.69	5.40±0.59	**4.67±0.33**
ENB	1	3.01±0.13	**0.92±0.04**	2.22±0.20	1.24±0.10	6.07±0.15	3.01±0.12	*0.96±0.12*	1.07±0.09	**0.92±0.04**
	2	3.26±0.15	1.85±0.15	2.44±0.12	1.95±0.20	6.19±0.18	3.26±0.13	**1.76±0.12**	2.01±0.28	*1.82±0.14*

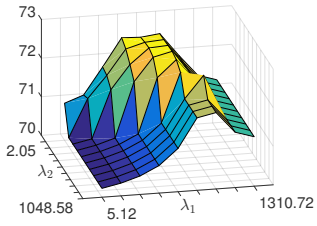

Figure 2. Mean accuracy (Kendall's Tau×100) of our algorithm on attribute 3 of PubFig dataset with hyper-parameters λ_1 and λ_2 varying in multiplicative intervals with factor 2.

Results. Overall, we improve upon the baseline independent estimator (Base_1) by adopting MTL approaches (Tables 1-2). While not all datasets and attributes show significant improvement, MTL approaches are on par with or outperform independent estimators. Since not all tasks are equally related (as suggested in Fig. 1), regMTL—which uniformly enforces pairwise task similarity—is further improved by allowing sparsity in task dependence (MTFL,AMTL) and/or task outliers (LRMTL). The performance variations of different MTL algorithms are significant (OSR, PubFig, Shoes, RF1, and ENB), while for SUN and Birds datasets the variation is less significant but noticeable. Among the three recent parametric MTL algorithms (MTFL, LRMTL, AMTL), LRMTL turned out to be the best followed by AMTL, but there was no clear winner indicating the complementary nature of different similarity-enforcing strategies. Also, for Birds dataset where by construction, all tasks are strongly related, the classical regMTL is competitive.

All four existing algorithms use shared parametric forms and extending them for the multiple heterogeneous estimator case is not straightforward. The importance of breaking this limitation can be clearly seen by comparing the results with the heterogeneous baselines (Base_2): Especially, for the OSR and RF1 datasets, by simply adopting heterogeneous estimators including DNNs, GPs and SVMs, even independent training already significantly improved performance. Being able to apply the MTL to these heterogeneous baselines, our algorithm further improves the performance and is consistently ranked as the best three. In particular, our algorithm constantly improves upon the initial Base_2. Bonilla et al.'s non-parametric Gaussian process-based MTL approach (GPMTL) [4] produced the best results for the second target attribute of the ENB dataset, improving the baseline with a large margin. However, their results on RF1 indicates that the performance varies significantly across different target attributes. The Spike and slab variational inference (SNS) demonstrated a similar behavior.

5. Conclusions

The best selection of task estimators depends on the problem of interest and the nature of the target attributes. However, existing MTL algorithms do not provide the ability to combine multiple heterogeneous estimator combinations. We demonstrated that heterogeneous combinations leads to improved learning performance. We build upon the idea of casting individual estimators into random variables and measuring their statistical dependence on the recently proposed kernel-based dependence measure (FSIC). Our approach is agnostic to the architecture of estimators and so enables us to discover and (selectively) enforce task dependence independently of the specific forms of individual estimators.

Evaluated on seven ranking and regression datasets, our algorithm is on par with or improves upon the baseline independent learning algorithms, Evgeniou and Pontil's classical uniform MTL algorithm, non-parametric GP-based algorithms, and four state-of-the-art parametric MTL algorithms.

One limitation of our approach is that it relies on the existence of the dataset X. While this is a moderate assumption, when X is not available or it does not fairly reflect the underlying probability distribution $P_{\mathcal{X}}$, the performance of our algorithm will degrade. Also, since our algorithm relies on the initial estimators S (see Eq. 13), the final performance will depend on their performance. One way to remove this dependence is to replace the first term in $\mathcal{O}$ (Eq. 13) by the task-dependent training loss.

The normalized FSIC criteria (Eq. 11) can be considered as a measure of similarity between estimators $\mathbf{f}^i$ and $\mathbf{f}^j$ as *data points*: Inspecting Eq. 10 reveals that, evaluated at columns of F, the empirical FSIC estimate $\hat{\phi}$ constitutes a positive definite kernel and, therefore, it induces a distance measure on $\mathbb{R}^N$ (N: the number of data points). This facilitates embedding individual estimations into a graph where the graph Laplacian L is constructed based on the kernel matrix $\Phi(F)$. Using the graph Laplacian, one could embed all tasks into a low-dimensional visualization space. Visualizing task dependence not only helps us to understand the nature of the problem, but also gives an insight into identifying potential subsets of tasks that can benefit from MTL. In the accompanying supplemental, we provide such a visualization and details of task clustering on the Birds dataset [26].

Acknowledgements: Y. Mejjati thanks Marie Sklodowska-Curie grant 665992. D. Cosker and K. I. Kim thank RCUK EP/M023281/1, and K. I. Kim thanks EPSRC EP/M00533X/2.

References

[1] A. Argyriou, T. Evgeniou, and M. Pontil. Convex multi-task feature learning. *Machine Learning*, 73(3), 2008.

[2] F. Bach, R. Jenatton, J. Mairal, and G. Obozinski. Optimization with sparsity-inducing penalties. *Foundations and Trends in Machine Learning*, 4(1):1–106, 2012.

[3] S. Ben-David and R. Schuller. Exploiting task relatedness for multiple task learning. In *COLT*, pages 567–580, 2003.

[4] E. V. Bonilla, K. M. Chai, and C. Williams. Multi-task Gaussian process prediction. In *NIPS*, pages 153–160, 2008.

[5] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2010.

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

[7] J. Chen, J. Liu, and J. Ye. Learning incoherent sparse and low-rank patterns from multiple tasks. *ACM TKDD*, 5(4):22:1–28, 2012.

[8] J. Chen, J. Zhou, and J. Ye. Integrating low-rank and group-sparse structures for robust multi-task learning. In *KDD*, pages 42–50, 2011.

[9] T. Evgeniou and M. Pontil. Regularized multi–task learning. In *KDD*, pages 109–117, 2004.

[10] P. Gong, J. Ye, and C. Zhang. Robust multi-task feature learning. In *KDD*, pages 895–903, 2012.

[11] S. K. Gupta, D. Phung, and S. Venkatesh. Factorial multi-task learning: a Bayesian nonparametric approach. In *ICML*, pages 657–665, 2013.

[12] D. Jayaraman, F. Sha, and K.Grauman. Decorrelating semantic visual attributes by resisting the urge to share. In *CVPR*, pages 1629–1636, 2014.

[13] W. Jitkrittum, Z. Szabó, and A. Gretton. An adaptive test of independence with analytic kernel embeddings. In *PMLR (Proc. ICML)*, pages 1742–1751, 2017.

[14] A. Kovashka, D. Parikh, and K. Grauman. Whittlesearch: Image search with relative attribute feedback. In *CVPR*, pages 2973–2980, 2012.

[15] N. Kumar, A. C. Berg, P. N. Belhumeur, and S. K. Nayar. Attribute and simile classifiers for face verification. In *ICCV*, pages 365–372, 2009.

[16] G. Lee, E. Yang, and S. J. Hwang. Asymmetric multi-task learning based on task relatedness and loss. In *PMLR (Proc. ICML)*, pages 230–238, 2016.

[17] D. Parikh and K. Grauman. Relative attributes. In *ICCV*, pages 503–510, 2011.

[18] A. Passos, P. Rai, J. Wainer, and H. Daumé III. Flexible modeling of latent task structures in multitask learning. In *ICML*, pages 1103–1110, 2012.

[19] G. Patterson and J. Hays. SUN attribute database: discovering, annotating, and recognizing scene attributes. In *CVPR*, pages 2751–2758, 2012.

[20] A. Pentina, V. Sharmanska, and C. H. Lampert. Curriculum learning of multiple tasks. In *CVPR*, pages 5492–5500, 2015.

[21] N. Quadrianto, A. Smola, T. Caetano, S. V. N. Vishwanathan, and J. Petterson. Multitask learning without label correspondences. In *NIPS*, pages 1957–1965, 2010.

[22] A. Smola, A. Gretton, L. Song, and B. Schölkopf. A Hilbert space embedding for distributions. In *ALT*, pages 13–31, 2007.

[23] E. Spyromitros-Xioufis, G. Tsoumakas, W. Groves, and I. Vlahavas. Multi-target regression via input space expansion: treating targets as inputs. *Machine Learning*, 104(1):55–98, 2016.

[24] C. Su, F. Yang, S. Zhang, Q. Tian, L. S. Davis, and W. Gao. Multi-task learning with low rank attribute embedding for person re-identification. In *ICCV*, pages 3739–3747, 2015.

[25] M. K. Titsias and M. Lázaro-Gredilla. Spike and slab variational inference for multi-task and multiple kernel learning. In *NIPS*, pages 2339–2347, 2011.

[26] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011.