Department of Computer Science
George Mason University Technical Reports

4400 University Drive MS#4A5
Fairfax, VA 22030-4444 USA
http://cs.gmu.edu/ 703-993-1530

Convex Multi-task Relationship Learning using Hinge Loss

Anveshi Charuvaka
acharuva@gmu.edu

Huzefa Rangwala
rangwala@cs.gmu.edu

Technical Report GMU-CS-TR-2014-8

Abstract

Multi-task learning improves generalization performance by learning several related tasks jointly. Several methods have been proposed for multi-task learning in recent years. Many methods make strong assumptions about symmetric task relationships while some are able to utilize externally provided task relationships. However, in many real world tasks the degree of relatedness among tasks is not known a priori. Methods which are able to extract the task relationships and exploit them while simultaneously learning models with good generalization performance can address this limitation. In the current work, we have extended a recently proposed method for learning task relationships using smooth squared loss for regression to classification problems using non-smooth hinge loss due to the demonstrated effectiveness of SVM classifier in single task classification settings. We have also developed an efficient optimization procedure using bundle methods for the proposed multi-task learning formulation. We have validated our method on one simulated and two real world datasets and have compared its performance to competitive baseline single-task and multi-task methods.

1 Introduction

In the standard settings of supervised machine learning, the objective is to learn a predictive function using example data. However, in real world settings, we often encounter situations with several related learning tasks. For example, in personalized email spam classification, the classification of spam for each user can be considered a separate task; in automated driving, steering and acceleration can be considered related tasks. Intuitively, it would seem that learning these related classification or regression tasks jointly should help us utilize common knowledge and improve generalization performance. In fact, this intuition is supported by empirical evidence provided by recent developments in transfer learning [26] and multi-task learning [10] [3] [31].

Multi-task learning (MTL) is a paradigm for learning several related tasks jointly. The generalization performance of the learned tasks is improved by utilizing inductive transfer across tasks. MTL achieves this by leveraging the training signal in related tasks [3, 10], and it has been empirically and theoretically [31] [4] shown to improve the generalization performance, especially when the training data is scarce. Some of the earliest models of multi-task learning were developed using multi-layer back-propagation neural networks [10]. Neural networks can be extended from single task to multiple tasks trivially with additional outputs for each task. Several tasks share the same input layer and one or more intermediate layers in this setting. By training multiple tasks simultaneously the back-propagation net prefers the inductive bias that helps multiple tasks [10].

Recently, there has been a significant progress in research in multi-task learning using Bayesian and regularized risk minimization framework (see [10, 31, 4, 2, 24, 12, 32, 7, 36, 29, 35, 1, 13] and the references therein). Various MTL models differ in the kinds of assumptions they make about the relatedness of the tasks and how these assumptions are incorporated into the learning algorithm. The simplest assumption could be that all the tasks share a similar set of parameters with some task specific variations [13]. This would be the case for personalized spam classification where generic spam is common for all users but different users might vary to some degree in individual spam. Some MTL formulations try to extract a good representation of the input features or a subset of features that are informative for all the tasks [15, 1]. Finally, we might also be interested in learning a good distance metric for all the predictive tasks [31].

In this work, we propose a convex formulation for multi-task relationship learning using non-smooth hinge loss. In section 2, we provide a brief review of the literature in regularized multi-task learning. Then, we motivate our MTL formulation and present an optimization method based on bundle methods in section 3. Finally in section 4, we evaluate the performance of our method on a simulated dataset and two real world datasets, and compare it to an STL model using SVM and two MTL models

2 Regularized Multi-task learning

Given a training set with n input-output example pairs, $\{(x_1, y_1), \ldots, (x_n, y_n)\}$, the objective of standard machine learning models is to learn a mapping function $f : \mathcal{X} \to \mathcal{Y}$ between the input domain $\mathcal{X}$ and the output domain $\mathcal{Y}$ which minimizes the loss on data not encountered in training. The training examples $(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$ are drawn from an unknown distribution. The regularized risk minimization framework achieves this goal by modeling an objective function as a trade-off between loss function, which minimizes the error, and regularization penalty, which controls the model complexity to discourage over-fitting.

$$\min_{w} \sum_{i=1}^{n} \underbrace{\mathcal{L}\left(w, x_i, y_i\right)}_{loss} + \lambda \underbrace{\mathcal{R}\left(w\right)}_{regularization} \tag{1}$$

This can be generically represented as (1) where w is the set of model parameters to be learned. This principle can be extended to MTL, where we have T tasks with training data for each of the $t = 1 \ldots T$ tasks, given by $\{(x_{it}, y_{it}) : i = 1 \ldots n_t\}$. The combined learning objective can be written as,

$$\min_{W} \sum_{t=1}^{T} \sum_{i=1}^{n_t} \underbrace{\mathcal{L}\left(w_t, x_{it}, y_{it}\right)}_{loss} + \lambda \underbrace{\mathcal{R}\left(W\right)}_{regularization} \tag{2}$$

where n_t is the number of training instances for the t^{th} task, w_t denotes the model parameters for the t^{th} task, and $W = \{w_t\}_{t=1}^{T}$ is the combined set of model parameters for all the tasks. Various multi-task learning methods take this general approach to build combined models for many related tasks, typically, by enforcing MTL assumptions through regularization term.

One of the first MTL methods based on regularized risk minimization framework was proposed by Evegeniou and Pontil [13]. The key assumption of their model is that all the tasks are closely related and their model weights are similar. This assumption is incorporated into the method by a regularization term that penalizes the deviation of the model weights for each task from the mean of all tasks. However, the assumption of symmetric relationships between all tasks made by this formulation are not suitable for many real world problems where the degree of relatedness between different tasks can vary. Clustered MTL formulations [37, 14], on the other hand, assume that tasks are grouped into clusters such that tasks within each cluster share greater similarity with other tasks in the same cluster.

Some methods make try to constrain the model weights to a low dimension subspace. For instance, the MTL model proposed by Kumar *et al.* [23] represents the weight matrix as a product $W = LS$ where each column of the matrix L represents a latent task and S is a sparse matrix. The number of latent tasks is smaller than the number of tasks. In conjunction with sparsity of S, this formulation, therefore, enforces a constraint that the tasks are combinations of few latent tasks. Whereas, the trace norm based formulation [17, 28] tries to minimize the rank of the weight matrix W by enforcing the constraint that the tasks lie in a low dimensional sub-space.

In multi-task feature learning and feature selection methods [16, 2, 24, 25], sparse learning schemes, similar to lasso [33] type regularization, are used to select or learn a common set of features across many related tasks. A common assumption made by many methods [13, 1, 15] is that all tasks share a common sub-set of informative features. This may be a limitation in certain settings, which is addressed by the tree guided group lasso method proposed by Kim *et al.* [21] where external task relationships guide the feature selection by enforcing a group wise sparsity constraints.

Whenever the relationships between tasks are available, it is beneficial to take them into account. The MTL formulations proposed in [12, 19] incorporate externally provided task relationship into the regularization term and penalize the deviations of only related tasks. However, these relationships might not be available and may need to be determined from the data. Although clustered multi-task learning can extract related groups of tasks to some extent, one of their shortcomings is that the number of task clusters is not known beforehand and hence, needs to be determined through parameter tuning. Another set of approaches, mostly based on Gaussian Process models, learn the task co-variance structure [7, 36] and are able to take advantage of both positive and negative correlations between the tasks. The method proposed in this paper also falls into the task relationship inference paradigm and tries to uncover the task relationships using the training data.

3 Multi-task Relationship Learning

3.1 Notation

In this section we describe the commonly used notations in this paper. I denotes the identity matrix of appropriate size. $A \succeq 0$ signifies that A is a positive semidefinite matrix, $tr\left(A\right)$ denotes the trace of a symmetric matrix A, which is defined as the sum of the diagonal elements.

$|1 - yf|_+$ denotes the hinge loss between the prediction score f and the actual label y, where $|g|_+ \equiv \max(0, g)$. $[p : q]$ denotes the set of integers i such that $p \leq i \leq q$.

We use T for the number of tasks and D for the number of input dimensions. $W \in \mathbb{R}^{D\times T}$ denotes the matrix of task weights, where each column $w_t \in \mathbb{R}^D$ is the weight vector for tasks $t \in [1 : T]$, the rows of W are denoted by $\tilde{w}_j \in R^T$ for $j \in [1 : D]$. $\Omega \in R^{T\times T}$ denotes the task covariance matrix. To simplify notation, we also define $\Sigma \equiv \Omega^{-1}$; where Ω^{-1} is the inverse of Ω.

3.2 MTRL using hinge loss

The multi-task relationship learning (MTRL) for smooth squared loss was proposed by Zang and Yeung [36]. The key feature of this formulation is that it does not require the task relationships to be known beforehand and it can exploit both the positive and negative task correlations. In order to extend the formulation to classification problems, it can be noted that the same regularizer can be utilized in order to leverage its advantages while using a loss function well suited for classification. In STL settings, SVM has been empirically shown to perform well for a diverse array of tasks. Hence, we chose to extend the MTRL formulation using SVM hinge loss [9]. The new model for MTLR using hinge loss is given by the optimization problem in (3).

$$\begin{aligned}\min_{W,\Omega} J = &\sum_{t=1}^{T} \frac{1}{n_t} \sum_{i=1}^{n_t} \left|1 - y_i^t \left(w_t^T x_i^t\right)\right|_+ + \frac{\lambda_1}{2} tr\left(WW^T\right) \\ &+ \frac{\lambda_2}{2} tr\left(W\Omega^{-1}W^T\right) \\ s.t.\ &\Omega \succeq 0 \\ &tr\left(\Omega\right) = T\end{aligned} \tag{3}$$

In problem (3), the objective consists of the hinge loss term which decomposes over all the examples and all the tasks. The regularizer is composed of two parts — the first term $\frac{\lambda_1}{2} tr\left(WW^T\right)$ is the Frobenius norm of W, which penalizes the complexity of the weight matrix, and the second term $\frac{\lambda_2}{2} tr\left(W\Omega^{-1}W^T\right)$ penalizes the deviations between correlated tasks based on the task covariance matrix Ω. Due to the replacement of a smooth squared loss with non-smooth hinge loss the problem becomes significantly more difficult to solve. Hence, we propose the following optimization procedure to solve it efficiently.

3.3 Optimization Procedure

Theorem 1. *Problem (3) is jointly convex in both W and Ω*

Proof. To prove that (3) is convex we need to prove that the objective function is convex and the constraints define a convex set. Since the sum of convex functions preserves convexity, we only need to show that the individual terms of the objective function are convex. It is obvious that the first two terms in the objective function in (3) are convex in terms of W, the first being the sum of hinge losses and the second being a norm. The last term $tr\left(W\Omega^{-1}W^T\right)$ can be rewritten as $\sum_{j=1}^{D} \tilde{w}_j \Omega^{-1} \tilde{w}_j^T$ where $\tilde{w}_j$ denotes the j^{th} row of W. The function $f(x, Y) = x^T Y^{-1} x$, where $x \in R^n$ and $Y \succeq 0$, known as the *matrix fractional function*, is convex (for proof refer to [8] page 76). Once again, due to the convexity preserving nature of sum, the last term is also convex. Finally, the constraint $\Omega \succeq 0$ defines a positive semi-definite cone and the $tr(M)$ is just an affine function of the matrix M, and hence, the intersection of both the constraints defines a convex set. □

Although problem (3) is jointly convex in W and Ω, simultaneous optimization over both W and Ω is difficult. Therefore, we use an alternating optimization strategy [5] and iterate between optimizing W with Ω fixed and vice-versa. The complete optimization procedure is summarized in Algorithm 1.

Optimizing Ω with fixed W

With W fixed, we can ignore the terms which are not dependent on Ω. The reduced problem involving only terms dependent on Ω can be rewritten as (4).

$$\begin{aligned}\min_{W,\Omega}\ & tr\left(W\Omega^{-1}W^T\right) \\ s.t.\ &\Omega \succeq 0 \\ &tr\left(\Omega\right) = T\end{aligned} \tag{4}$$

The analytical solution to the above problem can be obtained in the closed form given by (5),

$$\Omega = \frac{T(W^TW)^{\frac{1}{2}}}{tr\left((W^TW)^{\frac{1}{2}}\right)} \tag{5}$$

This can be proved using the Cauchy-Schwarz inequality for the Frobenius norm. For a detailed proof, please refer to [36].

Optimizing W with fixed Ω

To solve this problem, we use the Bundle Methods for Regularized Risk Minimization (BMRM) proposed by Teo *et al.* [30]. BMRM is an optimization scheme for efficiently solving convex optimization problems commonly encountered in regularized risk minimization framework. This method converges in $\mathcal{O}(\epsilon)$ steps for general convex problems and in $\mathcal{O}(\log(1/\epsilon))$ steps for smooth convex problems.

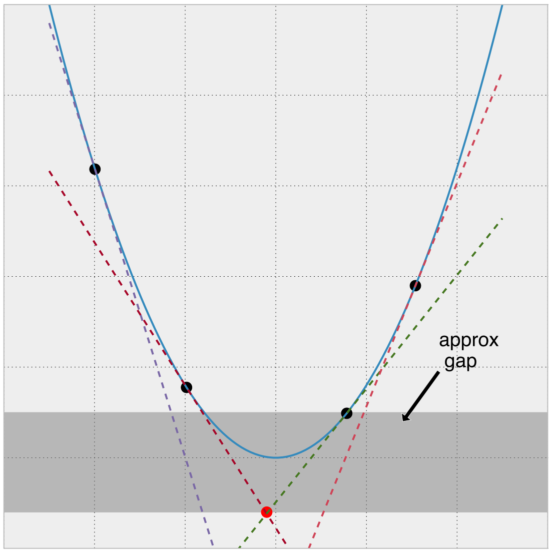


Figure 1: Illustration of convex function lower bounded by cutting planes. Cutting planes are tangents to the curve. The black dots represent the points at which the cutting planes are defined. The piecewise linear approximation is defined by the collective maxima of the cutting planes. The red dot represents the current minima of J_t^{CP}

The central idea of BMRM is based on the cutting plane and bundle methods, which is to bound a convex objective function using a piecewise linear approximations, denoted by J_t^{CP}. This is illustrated in Fig. 1 for a single variable convex function. The cutting planes are defined by the (sub)gradients to the curve at each of the points marked by black circles. The collective maximum of the cutting planes, $J_t^{CP} := \max_{1\le i\le t}\left\{tr\left(W^TA_i\right)+b_i\right\}$, provides an approximation of the objective function. Due to convexity of the objective function, the cutting planes, and as a consequence J_t^{CP}, always lower bound the objective. New cutting planes are added to the approximation as the algorithm progresses to make the lower bound progressively tighter. The highlighted gray area in Fig. 1 marks the approximation gap, which is defined as the difference between the current best minimum of the objective function and the minimum of the cutting plane approximation J_t^{CP}

Even though the cutting plane method is convergent [20], it generally suffers from instability and extremely slow convergence [22] when new iterates move far away from the previous ones. To mitigate this problem, bundle methods add a proximal term to prevent the zig-zag behavior caused by next iterates moving too far away from the current iterates. There are three popular variants of bundle methods [30].

proximal:

$$w_t = \mathbf{argmin}_w \left\{ \frac{\zeta_t}{2}\|w-\hat{w}_{t-1}\|^2 + J_t^{CP}(w)\right\} \tag{6a}$$

trust region:

$$w_t = \mathbf{argmin}_w \left\{ J_t^{CP}(w) \mid \frac{1}{2}\|w-\hat{w}_{t-1}\|^2 \le \kappa_t\right\} \tag{6b}$$

level set:

$$w_t = \mathbf{argmin}_w \left\{ \frac{1}{2}\|w-\hat{w}_{t-1}\|^2 \mid J_t^{CP}(w) \le \tau_t\right\} \tag{6c}$$

As it can be seen for (6a), (6b), and (6c), each of the variants penalize large steps in some form. However tuning the exact parameters for the proximal terms for achieving good rate of convergence can be difficult. The main insight of BMRM, which is based on *proximal* bundle methods, is to note that the objective function in regularized risk minimization consists of a loss term and a regularization term, as shown in (1), where the regularization term $\mathcal{R}(w)$ is typically a norm of some kind. Therefore, the regularization term can be used as the proximal term, obviating the need for a specialized proximal term and the additional inconvenience associated with its parameter tuning.

To apply BMRM to Problem (3) we split the objective into two parts — the loss term and the regularization term.

$$J = R_{emp} + \Psi$$

where

$$R_{emp} = \sum_{t=1}^{m}\frac{1}{n_t}\sum_{i=1}^{n_t}\left|1-y_i^t\left(w_t^Tx_i^t\right)\right|_+$$

$$\Psi = \frac{\lambda_1}{2}tr\left(WW^T\right) + \frac{\lambda_2}{2}tr\left(W\Sigma W^T\right)$$

We proceed by defining a cutting plane approximation of R_{emp} and use Ψ as the proximal term. The cutting plane approximation is defined by

$$R_t^{CP}(W) = \max_{1\le i\le t}\left\{tr\left(W^TA_i\right)+b_i\right\}$$

where $A_t \in \partial R_{emp}(W_{t-1})$ and $b_t = R_{emp}(W_{t-1}) - tr\left(W_{t-1}^TA_t\right)$. Therefore, we minimize problem (7) to get the next iterate

$$\begin{aligned} J_t(W) &= \Psi(W) + \max_{1\le i\le t}\left\{tr\left(W^TA_i\right)+b_i\right\} \\ W_t &= \arg\min_W J_t(W) \end{aligned} \tag{7}$$

The algorithm terminates when the gap between the approximation and the original objective function falls below a specified tolerance level ϵ, i.e. $min_{0\leq i\leq t} J\left(W_i\right) - J_t\left(W_t\right) \leq \epsilon$.

We solve the approximate problem (7) in its dual form using following result from [30] restated in a modified form for the current problem.

Theorem 2. *Let $\{A_i\}_{i=1}^{t}$ be the set of (sub)gradients and $b = \left(b_1, b_2, \ldots, b_t\right)^T$ be defined such tha $b_t = R_{emp}\left(W_{t-1}\right) - tr\left(W_{t-1}^T A_t\right)$. The dual problem of*

$$
\begin{aligned}
W_t &= \arg\min_{W} \left\{J_t\left(W\right)\right\} \\
J_t\left(W\right) &\equiv \Psi\left(W\right) + \max_{1\leq i\leq t}\left\{tr\left(W^T A_i\right) + b_i\right\} \\
&\quad is \\
\alpha_t &= \arg\max_{\alpha\in R^t}\left\{J_t^*\left(\alpha\right) \equiv \Psi^*\left(-\sum_i A_i\alpha_i\right) + \alpha^T b\right\}
\end{aligned}
$$

subject to:

$$
\begin{aligned}
\alpha &\geq 0 \\
\|\alpha\|_1 &= 1
\end{aligned}
$$

where Ψ^* denotes the Fenchel Dual of Ψ. Furthermore, W_t and α_t are related by the dual connection $W_t = \partial\Psi^*\left(-\sum_i A_i\alpha_i\right)$

Please note that our statement differs slightly, in notation, from the original theorem presented in [30] because the variables in the original theorem are presented in a vector form, whereas in our case the optimization variables are presented in matrix format.

Fenchel Dual of Ψ

In order to provide a concrete instantiation of the dual problem we have to compute the Fenchel dual of Ψ.

Definition 1. *Fenchel Dual: Let $\phi : \mathcal{W} \to \mathbb{R}$ be a convex function on a convex set $\mathcal{W}$. Then the dual ϕ^* of ϕ is defined as*

$$
\phi^*\left(\mu\right) := \sup_{w\in\mathcal{W}} w^T\mu - \phi\left(w\right) \tag{8}
$$

The matrix equivalent of dot product for vectors is the trace of the matrix product. Hence, the Fenchel dual of $\Psi\left(\cdot\right)$ is computed as follows.

$$
\begin{aligned}
\Psi^* &= \sup_{W} tr\left(U^T W\right) - \Psi\left(W\right) \\
&= \sup_{W} tr\left(U^T W\right) - \frac{\lambda_1}{2} tr\left(WW^T\right) - \frac{\lambda_2}{2} tr\left(W\Sigma W^T\right)
\end{aligned} \tag{9}
$$

To maximize R.H.S., we take its derivative w.r.t. W and equate it to zero to find the stationary point, noting that Σ is symmetric.

$$
\begin{aligned}
\left|\frac{\partial F\left(W, U\right)}{\partial W}\right|_{W=W*} &= 0 \\
U - \lambda_1 W^* - \lambda_2 W^*\Sigma &= 0 \\
\implies W^* = U\left(\lambda_1 I + \lambda_2\Sigma\right)^{-1} &\equiv UB,
\end{aligned}
$$

where we have defined $B = \left(\lambda_1 I + \lambda_2\Sigma\right)^{-1}$. Substituting back into (9), we get the closed form expression for Fenchel dual, which we further simplify using the cyclic property of trace [27].

$$
\begin{aligned}
\Psi^* &= tr\left(U^T UB\right) - \frac{\lambda_1}{2} tr\left(UBB^T U\right) - \frac{\lambda_2}{2} tr\left(UB\Sigma B^T U^T\right) \\
&\underline{= tr\left(UBU^T\right)} - \frac{\lambda_1}{2} tr\left(UBB^T U\right) - \frac{\lambda_2}{2} tr\left(UB\Sigma B^T U^T\right) \\
&= tr\left\{U\left(B - \frac{\lambda_1}{2} BB^T - \frac{\lambda_2}{2} B\Sigma B^T\right) U^T\right\} \\
&= tr\left\{UGU^T\right\},
\end{aligned}
$$

where we define $G := \left(B - \frac{\lambda_1}{2} BB^T - \frac{\lambda_2}{2} B\Sigma B^T\right)$

Finally, the connection between primal and dual variables according to Theorem 2 is given by

$$
\begin{aligned}
W_t &= \partial\Psi^*\left(-\sum_i A_i\alpha_i\right) \\
&= \left|\frac{\partial tr\left\{UGU^T\right\}}{\partial U}\right|_{U=-\sum_i A_i\alpha_i} \\
&= \left|2UG\right|_{U=-\sum_i A_i\alpha_i} \\
&= -2\left(\sum_i A_i\alpha_i\right) G
\end{aligned} \tag{10}
$$

The dual problem in this form can be solved using any constrained quadratic solver. This optimization procedure is summarized in Algorithm 2.

4 Results

In this section we present the results of experimental evaluation of our method on a simulated dataset and two real world datasets commonly used in multi-task learning literature and compare the results with a single task model and two competitive multi-task learning models described below.

MTRL: Our method proposed in this paper.

Algorithm 1 Main Algorithm (MTRL-hinge)

```
Input: $X,\ Y,\ \lambda_1,\ \lambda_2$
Output: $W, \Omega$
$W_0 \leftarrow 0;\ \Omega_0 \leftarrow T * I$
$\epsilon \leftarrow 10^{-3}\ max\_iter \leftarrow 30;\ t \leftarrow 0;\ \epsilon_t \leftarrow \infty$

for $t = 1, \ldots, max\_iter$ do
    $W_t := optimizeW\left(X, Y, W, \Omega_t, \lambda_1, \lambda_2\right)$
    $\Omega_t := T * \frac{\left(W_t^T W_t\right)^{\frac{1}{2}}}{tr\left(\left(W_t^T W_t\right)^{\frac{1}{2}}\right)}$
    if $\frac{\|W_t - W_{t-1}\|_2}{\|W_t\|_2} > \epsilon$ then
        break;
    end
end
return $W_t, \Omega_t$
```

Algorithm 2 OptimizeW (BMRM)

```
Input: $W_1, X, Y, \Omega, \lambda_1, \lambda_2$
Output: $W_{best}$
$t \leftarrow 1, \epsilon \leftarrow 10^{-4}, max\_iter \leftarrow 100$
$B = \left(\lambda_1 I + \lambda_2 \Omega^{-1}\right)^{-1}$
$G = \left(B - \frac{\lambda_1}{2} B B^T - \frac{\lambda_2}{2} B \Omega^{-1} B^T\right)$

for $t = 1, \ldots, max\_iter$ do
    $A_t \in \partial R_{emp}\left(W_{t-1}\right)$
    $b_t := R_{emp}\left(W_{t-1}\right) - tr\left(W_{t-1}^T A_t\right)$
    $M$ such that $M_{ij} = tr\left\{A_i G A_j^T\right\}$
    $\alpha_t = \arg\min_{\alpha \in R^t}\left\{\alpha^T M \alpha - \alpha^T b \mid \alpha \geq 0, \mathbf{1}^T \alpha = 1\right\}$
    $W_t = -2\left(\sum_i A_i \left(\alpha_t\right)_i\right) G$
    $W_{best} = arg_{W_t}\left\{min_{0 \leq i \leq t} J\left(W_i\right)\right\}$
    $\epsilon_t \leftarrow J\left(W_{best}\right) - J_t\left(W_t\right)$
    if $\epsilon_t < \epsilon$ then
        break
    end
end
return $W_{best}$
```

SVM: STL baseline using support vector machines. SVM is the most competitive single task learning baseline because our method uses hinge loss in its objective.

TRACE: MTL formulation using Trace Norm minimization [17, 28] tries to enforce a low rank constraint on the combined weights matrix for all the tasks, thereby favoring the weight vectors for the tasks that lie in a low dimensional subspace. Due to the difficulties involved in directly minimizing the rank of a matrix, trace norm is used as a surrogate.

L21: MTL formulation uses L_{21} norm as the regularizer [1, 11]. This has the effect of joint feature selection on all the tasks.

Both TRACE and L21 MTL formulations use a smooth logistic loss which is different from the non-smooth hinge loss used in our method. For our experiments we used the publicly available implementation of libSVM for SVM [1]. TRACE and L21 are available in MALSAR package [18], which provides implementations for several well known multi-task methods using logistic loss and squared loss.

4.1 Simulated Toy Dataset

In this section, we discuss the results on a simulated dataset. The purpose of testing our method on this dataset is to verify that our method is indeed able to extract the task relationships as encoded in the data. By controlling the dataset generation process we know the ground truth about the task relationship, which can be then compared with the results of the method execution. To generate this dataset, we create two groups of four tasks each by first generating two orthonormal vectors which form the bases for the two groups. Then, we add random noise to the weight vectors of the group weight vectors to generate four independent task weight vectors in each group. Therefore, by construction we have one group of four tasks which has high correlation within the group and another group of four tasks with similarly high correlation within the group. Finally, we generate random examples and assign positive and negative classes using the generated weight vectors. Any correlation, positive or otherwise, with tasks from the other group are mere chance occurrences. We chose the number of input dimensions as 4. Each task has equal number of positive and negative training examples. We used a validation set for tuning the model parameters in the range $\left\{10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 10^{0}\right\}$ and measured the performance on an independent test set.

In Table 1, we compare the accuracy of our MTRL formulation with that of STL using SVM. Our MTRL model outperforms SVM by a huge margin when the amount of

[1] http://www.csie.ntu.edu.tw/~cjlin/libsvm/

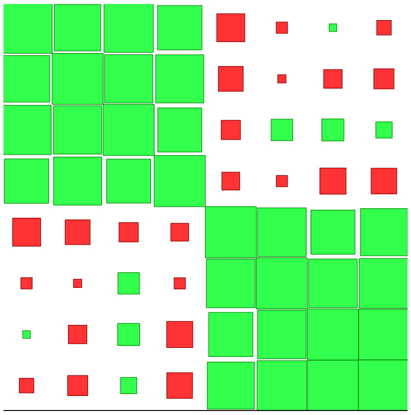


Figure 2: Learned task correlations for simulated data using 10 training examples. Each square represents the correlation between a pair of tasks. The size of the squares represents the magnitude of the value with positive values shown in green and negative values in red.

Table 1: Simulated Dataset Accuracy

Train Size (# Examples)	SVM	MTRL
5	0.8158	0.8785
10	0.8979	0.9186
30	0.9815	0.9822
50	0.9941	0.9948

training data per task is small. Given sufficient amount of training data, SVM performs just as well as our MTRL formulation. In Fig. 2 we show the task correlations for our MTRL model learned with 10 training examples. The figure shows two distinct groups of tasks uncovered by our method.

4.2 Landmine Detection

The landmine detection dataset[2] consists of 29 tasks. Each task is a binary classification problem of learning a classification model to discriminate between instances of landmines and clutters using features extracted from radar images. The 9 features in this dataset consist of four moment-based features, three correlation-based features, one energy ratio feature and one spatial variance feature [34]. Tasks 1-15 are taken from regions with high foliage and the rest are taken from bare earth or desert regions. Therefore, it is reasonable to assume that the tasks form two distinct groups. The number of examples per task vary between 445 and 690, whereas the number of positive examples per task vary in the range from 15 to 48.

We selected 25% of the examples for test set and 25% for validation set and using the remaining 50% of the examples we created datasets of varying sizes to evaluate classification performance at different training sizes. We used the validation set to select the best parameters for all the methods in the range $\{10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 10^{0}\}$.

Table 2: Landmine AUC Scores

Train Size (%)	SVM	MTRL	TRACE	L21
10	0.6902	0.6708	0.7512	0.7533
20	0.6697	0.6688	0.7393	0.7384
30	0.6841	0.6914	0.7711	0.7676
40	0.6983	0.7107	0.7708	0.7646
50	0.7016	0.7225	0.7724	0.7682

Due to the imbalance in the number of positive and negative examples in this dataset, accuracy is not a good measure of performance. Hence, we evaluate the performance of different models using area under the receiver operating characteristic curve (AUC). The results for this dataset are provided in Table 2. In general, with increasing training size the performance of the classifier improves, as can be expected. Our MTRL model performs better than the STL using SVM, but for this dataset, the performance using hinge loss based models was considerably worse than the models using logistic loss.

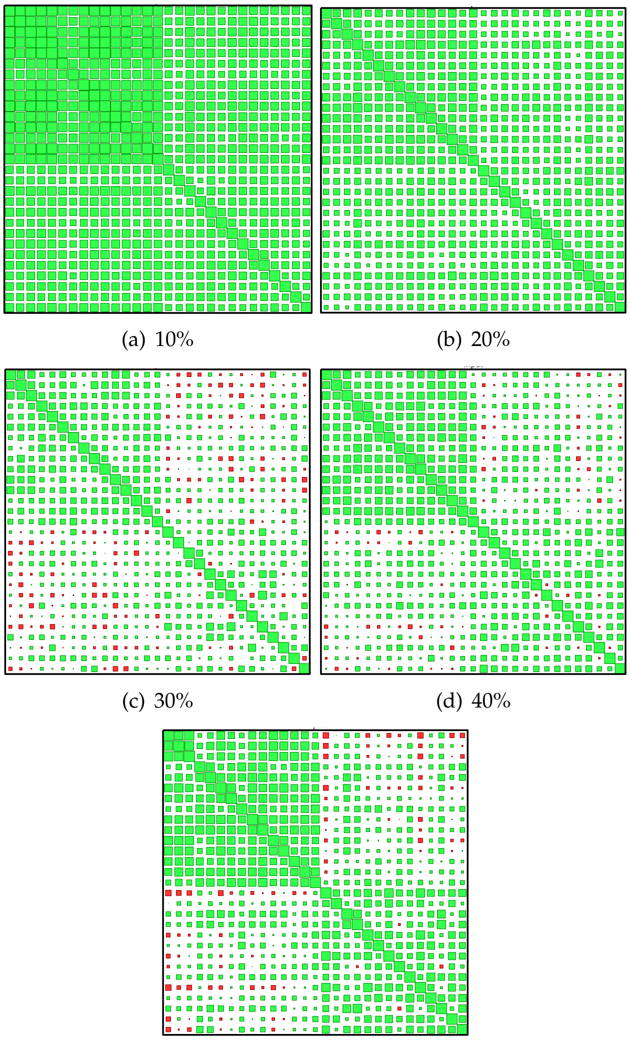
As mentioned earlier, the tasks in this dataset are naturally clustered into two groups. In order to observe our method's ability to extract these groups at various training sizes, we plotted the correlations of the weight vectors of the models extracted from the learned model parameter Ω in Fig. 3, for different training sizes. We observed that the first group of tasks, consisting of tasks 1-15, is more strongly correlated than the second group of tasks, consisting of task 16-29. With sufficient training sample sizes, the correlation pattern between tasks recovered by MTRL converges to the expected pattern consisting of two distinct groups.

4.3 Amazon Sentiment Classification

The tasks in Amazon sentiment classification dataset [3] are to classify the the polarity of different product reviews using their text. This dataset was originally provided by Blitzer *et al.* [6] and used in the context of domain adaptation. The instances consist of product reviews, and ratings, which are provided in terms of 1 to 5 stars. The ratings are converted into positive and negative reviews for classification tasks. Each task corresponds to a particular product category - books, DVDs, electronics, and kitchen appliances. 1000 positive and 1000 negative examples are available for each task. The instances are represented by a term fre-

[2] http://people.ee.duke.edu/~lcarin/LandmineData.zip

[3] http://www.cs.jhu.edu/~mdredze/datasets/sentiment/



(a) 10%

(b) 20%

(c) 30%

(d) 40%

(e) 50%

Figure 3: Task correlations of best models for Landmine dataset. Sub-figures correspond to different percentages of data used for training. Each square represents the correlation between a pair of tasks. The size of the squares represents the magnitude of the value with positive values shown in green and negative values in red.

Table 3: Per Task Accuracy - Amazon Sentiment Classification

Train Size (%)	Method	Books	DVDs	Electronic	Kitchen Appliances
10	SVM	0.6360	0.6800	0.7720	0.7560
	MTRL	**0.7380**	0.7500	**0.8280**	**0.8100**
	TRACE	0.7000	**0.7620**	0.8180	0.7920
	L21	0.6380	0.6960	0.7640	0.7620
20	SVM	0.7040	0.7680	0.8080	0.8020
	MTRL	0.7580	**0.7860**	0.8640	**0.8420**
	TRACE	**0.7600**	0.7680	**0.8660**	**0.8420**
	L21	0.7000	0.7400	0.8200	0.7620
30	SVM	0.7540	0.7880	0.8060	0.8160
	MTRL	0.7860	**0.8020**	0.8600	**0.8520**
	TRACE	**0.7940**	0.7960	**0.8620**	0.8420
	L21	0.7280	0.7440	0.8320	0.8320
40	SVM	0.7420	0.8180	0.8200	0.8240
	MTRL	**0.7980**	**0.8380**	**0.8640**	**0.8340**
	TRACE	0.7860	0.8260	0.8560	0.8280
	L21	0.7320	0.8020	0.8160	0.8220
50	SVM	0.7740	0.8160	0.8240	0.8540
	MTRL	**0.8080**	**0.8440**	**0.8820**	**0.8640**
	TRACE	0.7940	0.8300	0.8720	0.8460
	L21	0.7360	0.8140	0.8420	0.8400

quency vector of 473856 feature dimensions. We selected 25% examples for testing and 25% examples for validation and to assess the performance with different training sizes, we created different training split with 10%, 20%, 30%, 40%, and 50% of the original data. We validated the models using parameters in the range $\{10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 10^{0}\}$ on a validation set and selected the best model for final test on a held out test set.

Accuracies on the test set for various methods are reported in Table 3. Both MTRL and Trace perform considerably better than STL and L21. We expected L21 to perform well, given the high dimensionality of this dataset, but, it performed worse than STL for this dataset. The overall performance of our model was better than all other models.

In Table 4, we show the task correlations recovered by our model for different tasks. We observed a strong correlation between the Books-DVDs and Electronics-Kitchen Appliances task pairs.

5 Conclusion

In this work, we have presented a multi-task model using non-smooth hinge loss which is capable of learning the task relationships between different tasks. Our method can not only use positive task relationships, but it can also leverage negative task relationships. We demonstrated the effectiveness of the method in im-

Table 4: Task Correlation - Amazon Sentiment Classification

	Correlation Tables			
Train Size(%)	Books	DVDs	Electronic	Kitchen Appliances
10	1.0000	0.7830	0.6192	0.5629
	0.7830	1.0000	0.7291	0.7000
	0.6192	0.7291	1.0000	0.9413
	0.5629	0.7000	0.9413	1.0000
20	1.0000	0.5848	0.5492	0.5516
	0.5848	1.0000	0.4589	0.5263
	0.5492	0.4589	1.0000	0.7397
	0.5516	0.5263	0.7397	1.0000
30	1.0000	0.5898	0.5774	0.6009
	0.5898	1.0000	0.5557	0.5885
	0.5774	0.5557	1.0000	0.7925
	0.6009	0.5885	0.7925	1.0000
40	1.0000	0.6084	0.4683	0.5578
	0.6084	1.0000	0.4847	0.5593
	0.4683	0.4847	1.0000	0.7450
	0.5578	0.5593	0.7450	1.0000
50	1.0000	0.6835	0.5638	0.6386
	0.6835	1.0000	0.5782	0.6214
	0.5638	0.5782	1.0000	0.8179
	0.6386	0.6214	0.8179	1.0000

proving generalization performance using a simulated dataset and two real world datasets. Our method compared favorably with competing multi-task learning methods and consistently outperformed them on one of the datasets. We also compared the performance of our method to competing STL and MTL methods at different training sizes and found that the MTL methods significantly outperform STL methods and the performance gains are more pronounced for smaller training sizes. With respect to extracting task correlations we observed that some amount of training data is required to extract robust task correlations.

Acknowledgments

This work was supported by NSF IIS-0905117, IIS-1447489, and NSF career award to Huzefa Rangwala (IIS-1252318). The experiments were run on ARGO, a research computing cluster provided by the Office of Research Computing at George Mason University, VA. (http://orc.gmu.edu)

References

[1] A. Argyriou, T. Evgeniou, and M. Pontil. Multi-task feature learning. *NIPS*, 19:41, 2007.

[2] A. Argyriou, T. Evgeniou, and M. Pontil. Convex multi-task feature learning. *Machine Learning*, 73(3):243–272, 2008.

[3] J. Baxter. A model of inductive bias learning. *JAIR*, 12:149–198, 2000.

[4] S. Ben-David and R. Schuller. Exploiting task relatedness for multiple task learning. *Learning Theory and Kernel Machines*, pages 567–580, 2003.

[5] James Bezdek and Richard Hathaway. Some notes on alternating optimization. *Advances in Soft Computing—AFSS 2002*, pages 187–195, 2002.

[6] J. Blitzer, M. Dredze, and F. Pereira. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Annual Meeting-Association For Computational Linguistics*, volume 45, page 440, 2007.

[7] E. Bonilla, K.M. Chai, and C. Williams. Multi-task gaussian process prediction. *NIPS*, 20(October), 2008.

[8] S.P. Boyd and L. Vandenberghe. *Convex optimization - Boyd*. Cambridge Univ Pr, 2004.

[9] C.J.C. Burges. A tutorial on support vector machines for pattern recognition. *Data mining and knowledge discovery*, 2(2):121–167, 1998.

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

[11] X. Chen, W. Pan, J.T. Kwok, and J.G. Carbonell. Accelerated gradient method for multi-task sparse learning problem. In *Data Mining, 2009. ICDM'09. Ninth IEEE International Conference on*, pages 746–751. IEEE, 2009.

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

[13] T. Evgeniou and M. Pontil. Regularized multitask learning. *KDD*, pages 109–117, 2004.

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

[15] T. Jebara. Multi-task feature and kernel selection for SVMs. *ICML*, page 55, 2004.

[16] T. Jebara. Multitask sparsity via maximum entropy discrimination. *JMLR*, 12:75–110, 2011.

[17] S. Ji and J. Ye. An accelerated gradient method for trace norm minimization. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 457–464. ACM, 2009.

[18] J.Zhou, J.Chen, and J. Ye. *MALSAR: Multi-tAsk Learning via StructurAl Regularization*. Arizona State University, 2011.

[19] T. Kato, H. Kashima, M. Sugiyama, and K. Asai. Multi-task learning via conic programming. *NIPS*, 20:737–744, 2008.

[20] James E Kelley, Jr. The cutting-plane method for solving convex programs. *Journal of the Society for Industrial & Applied Mathematics*, 8(4):703–712, 1960.

[21] Seyoung Kim and Eric P. Xing. Tree-guided group lasso for multi-task regression with structured sparsity. In *ICML*, pages 543–550, 2010.

[22] Krzysztof C Kiwiel. Proximity control in bundle methods for convex nondifferentiable minimization. *Mathematical Programming*, 46(1-3):105–122, 1990.

[23] Abhishek Kumar and Hal Daume III. Learning task grouping and overlap in multi-task learning. *arXiv preprint arXiv:1206.6417*, 2012.

[24] J. Liu, S. Ji, and J. Ye. Multi-task feature learning via efficient l 2, 1-norm minimization. *UIA*, pages 339–348, 2009.

[25] G. Obozinski, B. Taskar, and M. Jordan. Multi-task feature selection. *ICML*, 2006.

[26] S.J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 2009.

[27] K.B. Petersen and M.S. Pedersen. The matrix cookbook. *Technical University of Denmark*, 2006.

[28] T.K. Pong, P. Tseng, S. Ji, and J. Ye. Trace norm regularization: Reformulations, algorithms, and multi-task learning. *SIAM Journal on Optimization*, 2009.

[29] Y. Qi, D. Liu, D. Dunson, and L. Carin. Multi-task compressive sensing with dirichlet process priors. In *Proceedings of the 25th international conference on Machine learning*, pages 768–775. ACM, 2008.

[30] Choon Hui Teo, SVN Vishwanthan, Alex J Smola, and Quoc V Le. Bundle methods for regularized risk minimization. *The Journal of Machine Learning Research*, 11:311–365, 2010.

[31] S. Thrun. Is learning the n-th thing any easier than learning the first? *NIPS*, pages 640–646, 1996.

[32] S. Thrun and J. O'Sullivan. Clustering learning tasks and the selective cross-task transfer of knowledge. *Learning to learn*, pages 181–209, 1998.

[33] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

[34] Y. Xue, X. Liao, L. Carin, and B. Krishnapuram. Multi-task learning for classification with dirichlet process priors. *The Journal of Machine Learning Research*, 8:35–63, 2007.

[35] Jian Zhang, Zoubin Ghahramani, and Yiming Yang. Flexible latent variable models for multi-task learning. *Machine Learning*, 73(3):221–242, 2008.

[36] Y. Zhang and D.Y. Yeung. A convex formulation for learning task relationships in multi-task learning. In *Proceedings of the Twenty-fourth Conference on Uncertainty in AI (UAI)*, 2010.

[37] J. Zhou, J. Chen, and J. Ye. Clustered multi-task learning via alternating structure optimization. *NIPS*, 2011.