Convex Multi-Task Learning using Squared and Hinge Loss

Aniket Deshmukh Naveen Murthy Aparna Garimella

Abstract

In this project, we explore and implement a machine learning method for learning sparse representations shared across multiple tasks by exploiting the inter-task relations. This method is a generalization of single-task L1 norm regularization which is non-convex and which controls the number of learned features shared across the various tasks. It is proved that this non-convex problem can be formulated as a convex optimization problem with an iterative algorithm, and hence can be shown to converge to an optimal solution. Though this framework is applicable to any loss function, the original authors have explored only squared loss function. We argue the robustness of hinge loss and extend this method by deriving the solution using an SVM like framework and random Fourier features for hinge loss. By implementing this algorithm on three real data sets, we show and conclude that the implemented method improves the performance relative to original work.

1 Introduction

Multi-task Learning (MTL) is a machine learning approach that simultaneously learns a task together with other related tasks, using a shared representation. The problem of learning data representations that are shared among multiple related supervised tasks is of primary interest to many research fields. This is particularly useful when few data samples per task are available, for which there is an advantage in using the data across many related tasks. For example, in computer vision, the problem of object detection could be treated as multiple related tasks where a shared representation of images could be beneficial. Another application is document classification, where the classification of each category is a task.

In recent literature, it has been frequently observed that it can be advantageous to learn all the tasks simultaneously instead of learning each task independently of the others [1, 5, 6, 9]. The notion of task relatedness has been modeled through assuming that all the learned functions are close to each other in some norm [3, 18]. Another way of modeling tasks' relatedness is that they share a common representation [4, 16]. For example, in modeling users' preferences in recommender systems, we could represent different choices of people using a common set of features describing those products.

Our project is based on the work done by Argyriou, Evgeniou and Pontil [5]. Agyriou et al. present a method for learning sparse representations shared across multiple related tasks. The authors provide a novel non-convex regularizer to control the number of learned features which is then solved using an equivalent convex optimization problem. The algorithm alternately learns task-specific functions and then, sparse representations for these functions which are common across the tasks.

The rest of the report is organized as follows. Section 2 gives the notation and general framework of the Multi-task learning problem. Section 3 describes the algorithm by Argyriou, Evgeniou and Pontil [5]. In section 4 we extend the work of Argyriou et al. using

hinge loss and changing the formulation of the problem, so that we can learn task specific functions using support vector machines and random Fourier features. We discuss the results of Argyriou's algorithm,the proposed algorithm and two other state of art algorithms on 3 benchmark datasets in section 5. We present our conclusions in section 6.

2 Problem Formulation

2.1 Notation

For the purpose of this project, we follow the following notations. Let $\mathbb{R}$ denote the set of real numbers and $\mathbb{R}_+$ and $\mathbb{R}_{++}$ denote subsets of non-negative and positive ones respectively. Let T denote the number of tasks and define $\mathbb{N}_T := \{1, ..., T\}$. For each task $t \in \mathbb{N}_T$, we have m input/output examples (x_{t1}, y_{t1}),...,$(x_{tm}, y_{tm}) \in \mathbb{R}^d \times \mathbb{R}$. Based on the above data, we aim to estimate T functions $f_t : \mathbb{R}^d \to \mathbb{R}$, $t \in \mathbb{N}_T$, such that they are statistically predictive and well approximate the data. For every $r, p \geq 1$, denote the (r,p)-norm of A as $\|A\|_{r,p} := (\sum_{i=1}^{d} \|a^i\|_r^p)^{\frac{1}{p}}$. Let the set of all $d \times d$ orthogonal matrices be denoted by $\mathbf{O}^d$.

2.2 Formulation

The functions f_t are assumed to be related to each other so that they all share a small set of common features. More precisely, the functions f_t can be written as

$$f_t(x) = \sum_{i=1}^{d} a_{it} h_i(x), t \in \mathbb{N}_T \tag{1}$$

where $h_i(x) : \mathbb{R}^d \to \mathbb{R}$ are the features and $a_{it} \in \mathbb{R}$ are the regression parameters. The formulation of sparse feature representation would mean that all the features but a few have zero coefficients across all the tasks. Here, the focus is only on linear features, with $h_i(x) = \langle u_i, x \rangle$, where $u_i \in \mathbb{R}^d$ are orthonormal. Let $U \in \mathbf{O}^d$ with the vectors u_i as columns. The functions f_t are also linear, as $f_t(x) = \langle w_t, x \rangle$, with $w_t = \sum_i a_{it} u_i$. Let W denote the $d \times T$ matrix with vectors w_t as columns, and let A denote the $d \times T$ matrix with entries a_{it}, such that we have $W = UA$.

The goal of this work is to learn the feature vectors u_i and parameters a_{it}. For the case of only one task (say task t) and the features u_i fixed from data $\{(x_{ti}, y_{ti})\}_{i=1}^m$, the aim is to minimize the empirical error $\sum_{i=1}^m L(t_{ti}, \langle a_t, U^T x_{ti} \rangle)$ subject to an upper bound on the number of nonzero components of a_t, where $L : \mathbb{R} \times \mathbb{R} \to \mathbb{R}_+$ is a prescribed loss function which is assumed to be convex in the second argument. However, this problem is intractable, and is relaxed by requiring an upper bound on the 1-norm of a_t to an equivalent unconstrained problem

$$min\{\sum_{i=1}^{m} L(y_{ti}, \langle a_t, U^T x_{ti} \rangle) + \gamma \|a_t\|_1^2 : a_t \in \mathbb{R}^d\}, \tag{2}$$

where $\gamma > 0$ is the regularization parameter. Using 1-norm leads to sparse solutions, thus many components of the learned vector a_t are zero.

Generalizing to the multi-task case, the regularization error function is

$$\mathcal{E}(A, U) = \sum_{t=1}^{T} \sum_{i=1}^{m} L(y_{ti}, \langle a_t, U^T x_{ti} \rangle) + \gamma \|A\|_{2,1}^2 \tag{3}$$

Since we do not simply want to select the features but also learn them, $\mathcal{E}$ is further minimized over U, considering the following problem

$$min\{\mathcal{E}(A, U) : U \in \mathbf{O}^d, A \in \mathbb{R}^{d \times T}\} \tag{4}$$

This method learns a low dimensional representation that is shared across the tasks. The number of features will be non-increasing with the regularization parameter.

3 Convex Multi-task Learning

3.1 Equivalent Convex Optimization Formulation

Problem (2.4) is non-convex problem, though it is independently convex in each of the variables A and U. Also, the norm $\|A\|_{2,1}$ is not smooth which makes it more difficult to optimize. Due to these two reasons, it is difficult to be solved and the main contribution of the original paper is its transformation to an equivalent convex problem

$$min\{\mathcal{R}(W, D) : W \in \mathbb{R}^{d\times T}, D \in \mathbf{S}_+^d, trace(D) \leq 1, range(W) \subseteq range(D)\} \tag{5}$$

where

$$\mathcal{R}(W, D) = \sum_{t=1}^{T}\sum_{i=1}^{m} L(y_{ti}, \langle w_t, x_{ti}\rangle) + \gamma \sum_{t=1}^{T} \langle w_t, D^+ w_t\rangle \tag{6}$$

$(\hat{A}, \hat{U})$ is an optimal solution for (4) if and only if $(\hat{W}, \hat{D}) = (\hat{U}\hat{A}, \hat{U}Diag(\hat{\lambda})\hat{U}^T)$ is an optimal solution for (5), where

$$\hat{\lambda}_i := \frac{\|\hat{a}^i\|_2}{\|\hat{A}\|_{2,1}} \tag{7}$$

3.2 Learning Algorithm

Problem (5) can be solved by alternately minimizing the function $\mathcal{R}$ with respect to D and w_t. When D is fixed, the minimization with respect to w_t reduces to learning w_t independently by a regularization method, say by an SVM or ridge regression kind of method. For fixed values of w_t, D is learnt by solving the following minimization

$$min\{\sum_{t=1}^{T} \langle w_t, D^+ w_t\rangle : D \in \mathbf{S}_+^d, trace(D) \leq 1, range(W) \subseteq range(D)\} \tag{8}$$

D is learnt using Algorithm 1 (the original authors use squared loss).

Algorithm 1 Multi-Task Feature Learning

1: **Input:** training sets $\{(x_{ti}, y_{ti})\}_{i=1}^m, t \in \mathbb{N}_T$
2: **Parameters:** regularization parameter γ, tolerance tol**Output:**d $\times d$ matrix D, $d \times T$ regression matrix $W = [w_1, ..., w_T]$
3: **Initialization:** set $D = \frac{I}{d}$
4: **while** $\|W - W_{prev}\| > tol$ **do**
5: **for** t = 1,...,T **do**
6: compute $w_t = argmin\{\sum_{i=1}^m L(y_{ti}, \langle w, x_{ti}\rangle) + \gamma\langle w, D^+ w\rangle : w \in \mathbb{R}^d\}$
7: **end for**
8: set $D = \frac{(WW^T)^{\frac{1}{2}}}{trace(WW^T)^{\frac{1}{2}}}$
9: **end while**

In the extended version of [5], the authors come up with an equivalent problem [2] where D^+ is replaced with D^{-1} by introducing a perturbation of the objective function in Eqn. 6.

4 Extensions - Hinge Loss

Rosasco et al. [12] conclude that the built-in statistical robustness of loss functions like the hinge or the logistic loss for classification leads to better convergence rates than the classic squared loss. So, we extend the results of our reference paper [5] when hinge loss is used instead of squared loss. We reformulate the problem statement so that it can be solved using SVM, especially using liblinear toolbox [7]. We also extend it to non linear feature space using random Fourier features (RFF) [10].

4.1 Updating W

For each task t, we have the optimization problem

$$\min_{w_t} \sum_{i=1}^{m} max\left(0, 1 - y_i w_t^T x_i\right) \; + \; \lambda w_t^T D^{-1} w_t \tag{9}$$

$$Let\ w_t \; = D^{\frac{1}{2}}\ v_t, \;\; v_t \; = D^{-\frac{1}{2}}\ w_t \;\; and \;\; z \; = D^{\frac{1}{2}}\ x \;\; (since\ D\ is\ PSD)$$

$$Then, \; w_t^T D^{-1} w_t \; = \; w_t^T D^{-\frac{1}{2}} D^{-\frac{1}{2}} w_t \; = \; v_t^T v_t \;\; and \;\; w_t^T x_i \; = \; w_t^T D^{-\frac{1}{2}} D^{\frac{1}{2}} x_i \; = \; v_t^T z_i$$

The new formulation of the problem becomes

$$\sum_{i=1}^{m} max \; \left(0, 1 - y_i v_t^T z_i\right) \; + \; \lambda \left\| v_t \right\|_2^2 \tag{10}$$

This is similar to an SVM problem with no bias term. The dual optimization problem with α being the dual variable, is

$$\max_{\alpha} \; -\frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j \left\langle z_i, z_j \right\rangle \; + \; \sum_i \alpha_i \tag{11}$$

$$s.t. \;\; 0 \; \leq \; \alpha_i \; \leq \; k$$

where k is a positive constant. Since the optimization problem is convex and has affine constraints, we can use KKT conditions to obtain

$$v_t \; = \; \sum_i \alpha_i y_i z_i \;\; and \;\; w_t \; = \; D^{\frac{1}{2}} v_t \; = \; D^{\frac{1}{2}} \sum_i \; \alpha_i y_i z_i \tag{12}$$

4.2 Updating D

$$D \; = \; \frac{\left(W W^T\right)^{\frac{1}{2}}}{Tr((W W^T)^{\frac{1}{2}})} \tag{13}$$

4.3 Implementation

Theorem 1 (Bochner's Theorem): A continuous kernel $k(x, y) = k(x - y)$ on R^d is positive definite iff $k(\delta)$ is the Fourier transform of a non-negative measure [13].

If a shift variant kernel $k(\delta)$ is properly scaled then Bochner's theorem guarantees that $p(w)$ in the following equation is a proper probability distribution.

$$k(x - y) = \int_{R^d} p(w) e^{jw'(x-y)} dw = E_w(\zeta_w(x) \zeta_w(y)^*) \; \approx \; \phi(x)^T \phi(y)$$

where w is drawn from p. Hence, we get an approximated non-linear feature mapping using RFF [10].

Theorem 2: For any $X \in R^{n \times T}$, if $U \Sigma U^T$ is the eigen-decomposition of $X^T X$, then $X X^T$ has the eigen-decomposition $\tilde{U} \Sigma \tilde{U}^T$, where $\tilde{U}^T = X U \Sigma^{-1/2}$ [17].

This is the principle used in kernelization of Fisher Linear Discriminant and PCA [15, 14].

The proposed algorithm in the above section is not completely kernelized. But we can certainly use random Fourier features. Eqn. 11 can be solved using any SVM toolbox

which allows us to set zero bias. There are two ways to solve this. In one method, we obtain dual variables α and update D using theorem 2. Another way of doing this is to directly get W and update D using Eqn. 13.

In our current implementation, we obtain W using liblinear. The advantage of using liblinear is the scalability of the algorithm. Also, we can extend it to a non-linear feature space using random Fourier features. We have two extensions of the standard convex MTL: one which uses just liblinear (referred to as *Linear kernel*) and the other which uses random Fourier features on top of liblinear (referred to as *RFF - Gaussian kernel*).

5 Experiments

5.1 Datasets

USPS: This is a popular dataset for handwritten digit recognition. Its dimensionality is reduced to 87 using PCA (95% of total variance). We use 1000, 500 and 500 samples for training, validation and testing respectively.

MNIST: MNIST is another dataset for digit recognition. We perform identical preprocessing as for the USPS dataset except that the dimensionality is reduced to 64.

UCI Dermatology Dataset: This is a multi-class dataset and deals with diagnosing one of 6 dermatological diseases. It consists of a total of 366 samples having 33 attributes.

5.2 Results

In this Multi-task setting, each task is a one vs. all classifier. For all three datasets described above, we compare convex Multi-task Learning (MTL) method proposed by Argyriou et al. [5] with the extended versions, one with a linear kernel and the other with a Gaussian kernel (RFF). We also compare convex MTL, with Multi-Task Lasso and a Dirty Model for Multi-Task Learning with the Least Squared Loss [8]. We implemented convex-MTL, linear kernel version and RFF version. We use MALSAR toolbox [19] for implementing Multi-task Lasso model and a Dirty Model for Multi-Task Learning. The parameters for all experiments were obtained using 5-fold cross validation and are given in Table 1. In our implementation with RFF, we have used 100 features to obtain the results shown in Figs. 1a, 1b and 1c. The impact of varying number of random Fourier features on the error rate is shown in Fig. 1d.

All methods are compared for different number of training points as shown in Fig. 1. As shown in Figs. 1a and 1b, the proposed versions, which use hinge loss instead of squared loss outperform every other method for MNIST and USPS datasets. Among all versions, the one with RFF performs the best followed by the linear kernel. This could be due to two reasons - one is we are mapping data into a higher dimensional space using RFF for a Gaussian kernel and also, we are using the hinge loss. The behavior of classifiers for UCI Dermatology dataset, shown in Fig. 1c is not clear. The versions with linear kernel and random Fourier features perform poorly when the number of training points is less. Eventually they do perform as well as the other classifiers when the number of training points is sufficiently large.

Table 1: Cross validation parameters for datasets.

	Convex MTL	Lasso	Dirty Model		Linear Kernel	RFF - Gaussian kernel	
	γ	λ	ρ_1	ρ_2	c	c	σ
MNIST	11	10	100	10	0.05	11	11
USPS	10	10	100	10	0.2	8	7
Dermatology	10	1	10	100	0.1	7	10

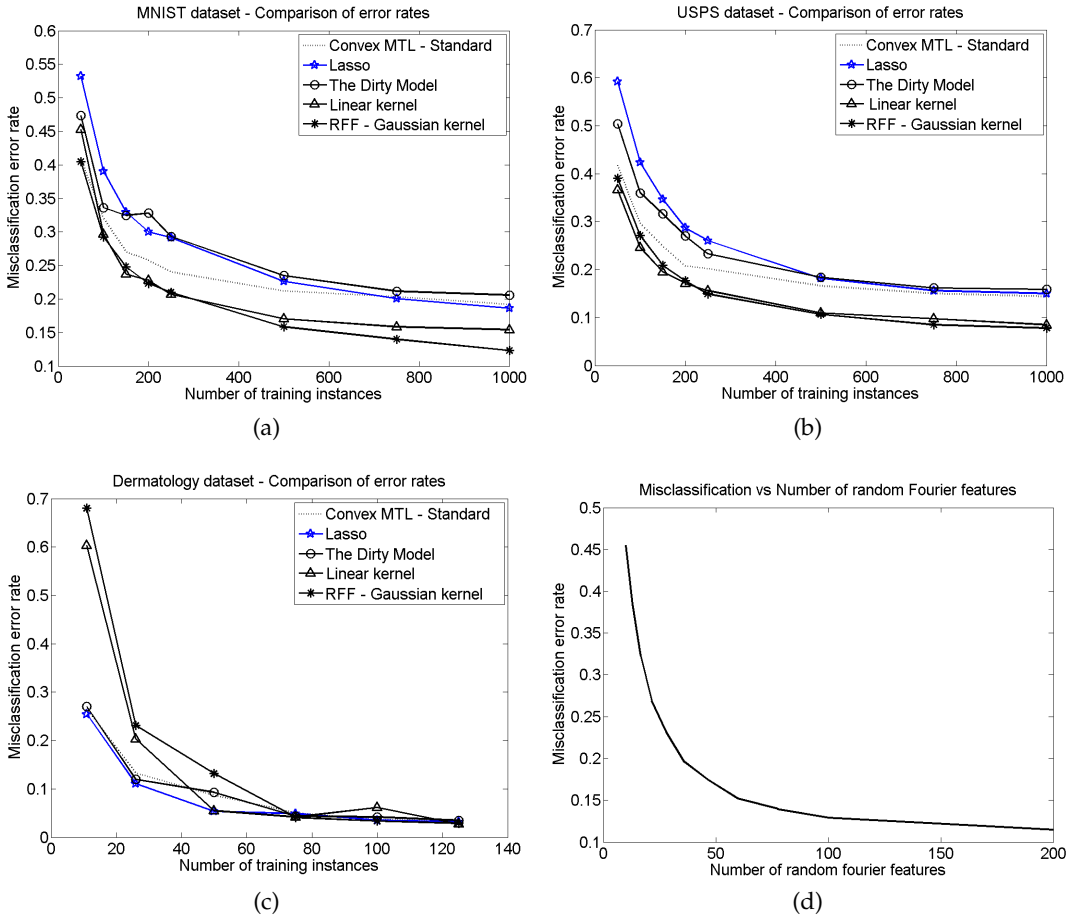

Figure 1: Misclassification error rates. (a) MNIST (b) USPS (c) Dermatological dataset (d) Effect of number of Random Fourier Features on misclassification error rate.

6 Conclusions

In this project, we implemented convex Multi-task Learning (MTL) proposed in [5] and got an idea of the various applications of MTL in general. The original authors used the squared loss for their MTL framework and we extended this to a hinge loss based algorithm. Experimental results on three real-world datasets show that the hinge loss improves classification accuracy. Also, we have used random Fourier features in order to obtain a non-linear feature mapping which further improves accuracy. MTL is very effective in scenarios where multiple tasks indeed have a shared feature representation. However, this might have an adverse effect in cases where this hypothesis of shared features is not true. Also, many applications of MTL involve multi-class classification and there may be existing frameworks which are more suited to such problems than MTL.

7 Work distribution

- Common - Literature survey, Documentation, Cross validation, Ortho-MTL [11]
- Aparna - MALSAR algorithms and experiments
- Aniket - Hinge loss framework, Linear Kernel & RFF implementation experiments
- Naveen - Convex MTL implementation and experiments
- NOTE: We implemented Ortho-MTL but obtained results different from [11]

References

[1] R. K. Ando and T. Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *The Journal of Machine Learning Research*, 6:1817–1853, 2005.

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

[3] B. Bakker and T. Heskes. Task clustering and gating for bayesian multitask learning. *The Journal of Machine Learning Research*, 4:83–99, 2003.

[4] R. Caruana. *Multitask learning*. Springer, 1998.

[5] A. Evgeniou and M. Pontil. Multi-task feature learning. *Advances in neural information processing systems*, 19:41, 2007.

[6] T. Evgeniou, C. A. Micchelli, and M. Pontil. Learning multiple tasks with kernel methods. In *Journal of Machine Learning Research*, pages 615–637, 2005.

[7] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. Liblinear: A library for large linear classification. *The Journal of Machine Learning Research*, 9:1871–1874, 2008.

[8] A. Jalali, S. Sanghavi, C. Ruan, and P. K. Ravikumar. A dirty model for multi-task learning. In *Advances in Neural Information Processing Systems*, pages 964–972, 2010.

[9] S. J. Pan and Q. Yang. A survey on transfer learning. *Knowledge and Data Engineering, IEEE Transactions on*, 22(10):1345–1359, 2010.

[10] A. Rahimi and B. Recht. Random features for large-scale kernel machines. In *Advances in neural information processing systems*, pages 1177–1184, 2007.

[11] B. Romera-Paredes, A. Argyriou, N. Berthouze, and M. Pontil. Exploiting unrelated tasks in multi-task learning. In *International Conference on Artificial Intelligence and Statistics*, pages 951–959, 2012.

[12] L. Rosasco, E. Vito, A. Caponnetto, M. Piana, and A. Verri. Are loss functions all the same? *Neural Computation*, 16(5):1063–1076, 2004.

[13] W. Rudin. *Fourier analysis on groups*. John Wiley & Sons, 2011.

[14] B. Schölkopf, A. Smola, and K.-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural computation*, 10(5):1299–1319, 1998.

[15] B. Scholkopft and K.-R. Mullert. Fisher discriminant analysis with kernels. *Neural networks for signal processing IX*, 1999.

[16] S. Thrun and J. O'Sullivan. Discovering structure in multiple learning tasks: The tc algorithm. In *ICML*, volume 96, pages 489–497, 1996.

[17] M. K. Warmuth, W. Kotłowski, and S. Zhou. Kernelization of matrix updates, when and how? In *Algorithmic Learning Theory*, pages 350–364. Springer, 2012.

[18] K. Yu, V. Tresp, and A. Schwaighofer. Learning gaussian processes from multiple tasks. In *Proceedings of the 22nd international conference on Machine learning*, pages 1012–1019. ACM, 2005.

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