Adaptive Group Sparse Multi-task Learning via Trace Lasso

Sulin Liu and Sinno Jialin Pan
Nanyang Technological University, Singapore
{liusl, sinnopan}@ntu.edu.sg

Abstract

In multi-task learning (MTL), tasks are learned jointly so that information among related tasks is shared and utilized to help improve generalization for each individual task. A major challenge in MTL is how to selectively choose what to share among tasks. Ideally, only related tasks should share information with each other. In this paper, we propose a new MTL method that can adaptively group correlated tasks into clusters and share information among the correlated tasks only. Our method is based on the assumption that each task parameter is a linear combination of other tasks' and the coefficients of the linear combination are active only if there is relatedness between the two tasks. Through introducing trace Lasso penalty on these coefficients, our method is able to adaptively select the subset of coefficients with respect to the tasks that are correlated to the task. Our model frees the process of determining task clustering structure as used in the literature. Efficient optimization method based on alternating direction method of multipliers (ADMM) is developed to solve the problem. Experimental results on both synthetic and real-world datasets demonstrate the effectiveness of our method in terms of clustering related tasks and generalization performance.

1 Introduction

Multi-task learning (MTL) aims at learning multiple different tasks together by transferring knowledge among them to improve generalization across all the tasks [Caruana, 1997; Pan and Yang, 2010]. The general idea is to extract and exploit shared information in related tasks, so that knowledge in terms of features or model parameters can be transferred among tasks. In most existing MTL methods, this is achieved by introducing an inductive bias in the hypothesis space of all tasks. The inductive bias includes assumptions on the task relatedness structure. Different assumptions lead to different particular approaches. Some assume that task parameters are close to each other or share a common prior [Evgeniou and Pontil, 2004], while some assume that they lie in a low rank subspace [Ando and Zhang, 2005; Argyriou *et al.*, 2008b; Liu *et al.*, 2009]. Chen *et al.* (2011) further assumes that the task parameters could be modeled by a combination of low rank and group sparse structure, where the second part is used for accounting outlier tasks. A key question in incorporating such assumptions is how to selectively transfer knowledge among related tasks only, and ensure that unrelated tasks do not affect each other. If information among unrelated tasks are shared with each other, generalization performance may become worse, which is known as *negative transfer* [Rosenstein *et al.*, 2005; Pan and Yang, 2010].

In the literature, there are various kinds of attempts to answer this question. A usual starting point is by assuming that tasks come from different clusters, and intra-cluster tasks share some form of relatedness. For instance, in [Jacob *et al.*, 2009; Zhou *et al.*, 2011], it is assumed that tasks within a cluster are close to each other in the sense of ℓ_2 distance. Regularization comes in the form of ℓ_2 distance of tasks within the same cluster. However, a major limitation in this kind of approach is that tasks that are negatively correlated will be put into different clusters. This prevents the sharing of information between negatively correlated tasks, which is undesired for MTL. A recent work [Zhou and Zhao, 2016] proposed a flexible clustered MTL method that aims to minimize the weighted ℓ_2 distance between each task and its representative tasks. However, it still fails to exploit common information among negatively correlated tasks.

Some methods were proposed to remedy this issue [Argyriou *et al.*, 2008b; Kang *et al.*, 2011] by assuming that task parameters in a cluster share a low dimensional subspace. Trace norm penalty is used as regularization in these methods to help find the low dimensional subspace within each cluster. Another work [Kumar and Daumé III, 2012] that exploits the low dimensional subspace structure of tasks assumes that each task is represented by a sparse representation of a set of latent basis tasks. It allows tasks from the same cluster to have similar sparse representation pattern. Meanwhile, it allows different clusters have one or more common bases. However, most of the above methods [Argyriou *et al.*, 2008b; Kang *et al.*, 2011; Kumar and Daumé III, 2012] need to either assume the number of clusters or assume the number of latent tasks beforehand, which makes it inflexible for MTL when the number of clusters is usually unknown. One may need to try different numbers of clusters (or latent tasks) and cross-validate before determining the number, which incurs a lot of computation. This is particularly costly when the number of clusters is large and hard to determine.

One solution to eliminate the selection process is by considering all available tasks as basis tasks, and each task can be reconstructed by a sparse linear combination of other tasks through a ℓ_1-norm regularizer [Lee *et al.*, 2016]. However, the ℓ_1-norm sparsity-induced solution may not provide a complete information of the clustering structure among the tasks. Specifically, when a task is highly correlated to several tasks, the ℓ_1-norm sparsity-induced solution will only assign a non-zero weight to one of the correlated tasks, and leave the weights of other correlated tasks being zeros, which prohibits the method from finding a clustering structure in MTL.

In our proposed method, we assume that each task parameter is represented by a linear combination of other tasks' parameters. To enable automatically grouping of correlated tasks, trace Lasso [Grave *et al.*, 2011] penalty instead of ℓ_1-norm is used as regularization in our model. This penalty is adaptive to correlatedness of variables, therefore, making it possible for our model to adaptively learn the grouping information as well as the sparse relationship between tasks. Our model can be reduced to MTL with ℓ_1-norm based regularization when all the tasks are uncorrelated with each other. But when some tasks have correlations with each other (which is the case in MTL), our model can adaptively learn the weights for linear combinations of tasks so that correlated tasks are grouped together. Experiments results on both synthetic and real world datasets validate the effectiveness of our method in both generalization performance and group clustering effect.

2 Related Work

There are two main categories of approaches proposed in the literature for MTL based on different assumptions. One category of approaches assumes all tasks be related to each other [Evgeniou and Pontil, 2004; Ando and Zhang, 2005]. Relatedness among tasks is either by enforcing the distance between task parameters to be close to each other or by enforcing the task parameters lie in the same low-rank subspace. However, in real-world applications this assumption may not hold. To address this issue, several methods in the other category have been proposed by assuming that tasks can be grouped into clusters. Within each cluster, task parameters share a common prior or/and are close to each other in some distance metric [Jacob *et al.*, 2009]. However, a problem for this kind of approaches is that tasks which are negatively correlated are put in different clusters, which is not desirable.

There are several works that aim at solving this problem in clustered MTL. One solution is to incorporate grouping information into subspace regularization based method [Argyriou *et al.*, 2008b; Kang *et al.*, 2011; Kumar and Daumé III, 2012]. In [Argyriou *et al.*, 2008b], tasks in each group are assumed to share a common linear transformation of some feature vectors. It has been shown to be equivalent to minimizing the trace norm of each group's task weight matrix. In [Kang *et al.*, 2011], tasks are assumed to come in groups, and each group of task parameters are assumed to lie in a low-dimensional subspace. This method proposes to minimize the square of trace-norm of each group's weight matrix. However, number of groups needs to be pre-determined in advance. In practice, cross-validation is needed to choose an appropriate number of groups. This will become more and more expensive when number of groups becomes larger and larger. In [Kumar and Daumé III, 2012], it assumes there exist a small number of latent tasks and every task can be represented by a linear combination of the latent tasks. The modeling of latent tasks allows tasks within a group lie in a low-dimensional subspace, and it also allows tasks from different groups to overlap with each other through having some common bases. A practical issue for this method is that number of latent tasks still needs to be determined beforehand.

Different from the above three clustered MTL methods, another work [Zhou and Zhao, 2016] allows for flexible clustering of tasks by assuming that one task can be clustered into multiple clusters with different weights. Each cluster has its own representative task, and each task is represented by some of these. This is done by regularizing the ℓ_2 distance between tasks with their representative tasks. Though this method can waive the determination on the number of task clusters, it fails to group negatively correlated tasks into a same cluster as ℓ_2-norm distance is used for measuring the similarity between task parameters. In some other recent works, methods were proposed to learn sparse multi-task relationships by imposing ℓ_1-norm penalty on task covariance matrix [Zhang and Yang, 2017] or on precision matrix [Zhang and Schneider, 2010; Rai *et al.*, 2012]. Those methods are based on probabilistic models by placing a matrix variate prior on task weight parameters. By imposing the sparsity-inducing regularization, those methods attempt to learn a sparse task covariance or precision matrix and use it in learning tasks jointly. They are able to learn negative task correlations. However, their focus is more on learning sparse relationship instead of learning the grouping structure of tasks.

Motivated by the limitations of existing clustered MTL methods, which need to either pre-assume certain information about the grouping, such as number of groups, or number of latent tasks, or fail to exploit negative correlations among tasks, we aim to design our method such that it could automatically find the grouping (or clustering) information among multiple tasks and exploit this grouping information in learning related tasks jointly. We also hope that tasks of both positive and negative correlations can be assigned to the same cluster. A recent work that is relevant to ours is [Lee *et al.*, 2016], where each task is assumed to be represented by other tasks, but through a linear combination of them. In their model, besides asymmetric relationship between tasks, it also aims at learning a sparse relationship between tasks by imposing a sparsity penalty. When tasks come in groups, their method could also discover the underlying structure of clusters, however the sparse solution of relationship could sometimes be too sparse to provide sufficient information in grouping related tasks together. We will further discuss the difference between our proposed method and theirs in Section 4.2.

3 Preliminary: Trace Lasso

Trace Lasso [Grave *et al.*, 2011] is proposed as a regularizer that interplays between the ℓ_1-norm and the ℓ_2-norm. Consider an empirical risk minimization problem as follows,

$$\min_{\mathbf{w}} \frac{1}{n} \sum_{i=1}^{n} \ell(\mathbf{w}^\top \mathbf{x}_i, y_i) + \tau \Omega(\mathbf{w}),$$

where $\mathbf{x}_i \in \mathbb{R}^{d\times 1}$ is an input, y_i is its corresponding output, and $\Omega(\mathbf{w})$ is a regularization term needed to be specified. We denote by $\mathbf{X} = (\mathbf{x}_1, \cdots, \mathbf{x}_n)^\top \in \mathbb{R}^{n\times d}$ the input matrix, and by $\mathbf{y} = (y_1, \cdots, y_n)^\top \in \mathbb{R}^{n\times 1}$ the output vector. The column vectors of $\mathbf{X}$, $\{\mathbf{x}^{(i)}\}_{i=1}^{d}$, are called predictors, which correspond to different dimensions of $\mathbf{w}$. Formally, the trace Lasso penalty is defined as

$$\Omega(\mathbf{w}) = \|\mathbf{X}\ \mathrm{Diag}(\mathbf{w})\|_*,$$

where $\mathrm{Diag}(\mathbf{w})$ converts the vector $\mathbf{w}$ into a diagonal matrix in which the i-th diagonal entry is $\mathbf{w}_i$. There are some interesting properties of trace Lasso:

- If all the predictors are orthogonal, then it is equivalent to ℓ_1-norm with normalized predictors, $\left\{\frac{\mathbf{x}^{(i)}}{\|\mathbf{x}^{(i)}\|_2}\right\}_{i=1}^{d}$.
- If all the predictors are equal, it becomes equivalent to ℓ_2-norm with normalized predictors.

These properties make the penalty adaptive to the correlation between predictors, and thus make the selection of predictors more stable than Lasso.

4 Methodology

Suppose we are given m tasks $\{T_i\}_{i=1}^{m}$, and each task T_i consists of a set of training data $\mathcal{D}_i = \{(\mathbf{x}_{ij}, y_{ij}), j = 1, 2, ..., N_i\}$, where $\mathbf{x}_{ij} \in \mathbb{R}^{d\times 1}$ denotes the j-th data input of task i with y_{ij} being its corresponding output. Denote by $\{\mathbf{w}_i \in \mathbb{R}^{d\times 1}\}_{i=1}^{m}$ the weight parameters of the m tasks, which are stacked to form a weight matrix $\mathbf{W}$ of size $d \times m$. To exploit the relationships between tasks, we assume that each task parameter is represented by a linear combination of other task parameters:

$$\mathbf{w}_i \approx \sum_{k=1}^{m} \mathbf{C}_{ki}\mathbf{w}_k = \mathbf{W}\mathbf{c}_i,\ \forall i \in \{1, 2, \cdots, m\},$$

where $\mathbf{C} \in \mathbb{R}^{m\times m}$ is a matrix that describes the correlation between tasks, $\mathbf{c}_i$ denotes the i-th column of $\mathbf{C}$, and the entry $\mathbf{C}_{ki}$ denotes the weight of basis task $\mathbf{w}_k$ in representing $\mathbf{w}_i$. It characterizes how much of the model of task k is transferred to the model of task i. Note that $\{\mathbf{C}_{ii}\}_{i=1}^{m}$ are defined to be zeros, and the values of the other entries are to be learned, which can be positive, zero, or negative such that both positive and negative correlations between tasks can be captured.

4.1 Motivation

By considering each column of $\mathbf{W}$ as a basis, then $\mathbf{c}_i \in \mathbb{R}^{m\times 1}$ can be considered as a new representation (or a vector of coefficients) for $\mathbf{w}_i \in \mathbb{R}^{d\times 1}$. Tasks that belong to the same group should have similar sparsity patterns in their new representations. Therefore, the grouping structure of the tasks is embedded in the task correlation matrix $\mathbf{C}$. Ideally, only coefficients between related tasks should be active. If we reorder the index of tasks according to task groups, we expect $\mathbf{C}$ to be a block-diagonal matrix, i.e. only tasks within a group should have weight coefficients with each other.

In the literature, a common approach to learn $\mathbf{C}$ is by imposing the ℓ_1-norm penalty on each column of $\mathbf{C}$, i.e., $\mathbf{c}_i$, [Kumar and Daumé III, 2012; Lee *et al.*, 2016]. However, when task parameters from the same group are highly correlated, the ℓ_1-norm minimization would generally select one representative task at random, leaving the coefficients of other related tasks to be zeros (or close to zeros). Therefore, the ℓ_1-norm induced sparse solution may not capture the sufficient information about task grouping. What we hope to derive is a method that can automatically learn the grouping structure of tasks. At the same time, by making use of the grouping structure, the generalization performance can be improved through learning the related tasks together.

Trace Lasso [Grave *et al.*, 2011] mentioned in section 3 is a regularizer that interplays between the ℓ_1-norm and the ℓ_2-norm. The regularizer is adaptive to the correlation of data, i.e., the task weight parameters $\mathbf{W}$ in our case, while learning the sparsity of the coefficient vectors, i.e., each $\mathbf{c}_i$ in our case. In other words, it incorporates the information of correlation between data predictors, $\{\mathbf{w}_i\}$s, into the regularizer. When all data predictors are uncorrelated, the trace Lasso behaves as the ℓ_1-norm. In the case when data predictors are highly correlated, trace Lasso behaves like the ℓ_2-norm. Therefore, we propose to use the trace Lasso penalty as a regularizer on the task linear combination coefficients $\{\mathbf{c}_i\}$'s. The adaptivity of trace Lasso to the correlation between different task weight parameters can help us to represent each task with a group of its correlated tasks.

4.2 Proposed Objective

Based on the motivation described in the previous section, our proposed method is formulated as follows:

$$\begin{aligned} \min_{\mathbf{W},\mathbf{C}} \sum_{i=1}^{m} \mathcal{L}(\mathbf{w}_i, \mathcal{D}_i) + \frac{\lambda}{2}\sum_{i=1}^{m} \left\| \mathbf{w}_i - \mathbf{W}_{-i}\mathbf{c}_i^{[-i]} \right\|_2^2 \\ + \gamma \sum_{i=1}^{m} \left\| \mathbf{W}_{-i}\ \mathrm{Diag}(\mathbf{c}_i^{[-i]}) \right\|_*, \end{aligned} \qquad (1)$$

where the $\mathcal{L}(\mathbf{w}_i, \mathcal{D}_i)$ denotes the training loss of the i-th task with a convex loss function $\mathcal{L}(\cdot)$, and $\lambda, \gamma > 0$ are the regularization parameters. $\mathbf{W}_{-i}$ denotes the matrix $\mathbf{W}$ without the i-th column, and $\mathbf{c}_i^{[-i]}$ denotes the column vector $\mathbf{c}_i$ without its i-th entry. The first term in the objective is the sum of training losses over all the tasks. The loss functions can be different for different learning problems. The second term acts as a regularization to enforce the assumption that each task is a linear combination of other tasks. And the third term is the trace Lasso penalty imposed on the task relationship vectors $\{\mathbf{c}_i\}$'s in $\mathbf{C}$.

Note that in [Lee *et al.*, 2016], each task is also represented by a linear combination of other tasks' parameters. However, instead of using the trace Lasso regularization as proposed in our method, their method used the ℓ_1-norm regularization on each coefficient vector $\mathbf{c}_i$. As discussed in Section 4.1, the ℓ_1-norm regularization fails to return rich clustering structure among the tasks. In addition, although it is possible for the matrix $\mathbf{C}$ to take negative values, they restrict $\mathbf{C}$ to be of non-negative values in their proposed model. In this way, their method fails to exploit negatively correlated tasks for task representation reconstruction.

5 Optimization

Before introducing the optimization method, we have the following theorem about the optimization problem (1).

Theorem 1. *Problem* (1) *is not jointly convex but is bi-convex with respect to* $\mathbf{W}$ *and* $\mathbf{C}$.

Proof. We prove that problem (1) is not jointly convex by showing that the last term $f(\mathbf{W}, \mathbf{c}) = \|\mathbf{W}\text{Diag}(\mathbf{c})\|_*$ is not convex through an counter example: given $\theta = 0.5$,

$$\mathbf{W}_1 = \begin{bmatrix} 10 & 10 \\ 10 & 10 \end{bmatrix}, \mathbf{c}_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \mathbf{W}_2 = \begin{bmatrix} 1 & 1 \\ 0 & 0 \end{bmatrix}, \mathbf{c}_2 = \begin{bmatrix} 1 \\ 1 \end{bmatrix},$$

we have

$$\begin{aligned} &\|(\theta\mathbf{W}_1 + (1-\theta)\mathbf{W}_2)\text{Diag}(\theta\mathbf{c}_1 + (1-\theta)\mathbf{c}_2)\|_* \\ &\quad > \theta\|\mathbf{W}_1\text{Diag}(\mathbf{c}_1)\|_* + (1-\theta)\|\mathbf{W}_2\text{Diag}(\mathbf{c}_2)\|_*, \end{aligned}$$

which violates the definition of convex function. Thus problem (1) is not convex.

With fixed $\mathbf{W}$, the optimization problem (1) is left with last two terms. It is easy to verify that the second last term is convex with respect to $\{\mathbf{c}_i^{[-i]}\}_{i=1}^m$. The last term is separable in $\mathbf{c}_i^{[-i]}$, and each sub-term is convex with respect to $\mathbf{c}_i^{[-i]}$ as shown in [Grave *et al.*, 2011].

If $\mathbf{C}$ is fixed, then the optimization problem (1) is left with the first term and the last term. It is easy to verify that the first term is convex with respect to $\mathbf{W}$ since $\mathcal{L}$ is convex with respect to $\mathbf{W}$. It remains to show that the last term is convex. Define $f(\mathbf{W}) = \|\mathbf{W}\text{Diag}(\mathbf{c})\|_*$, we have:

$$\begin{aligned} &f(\theta\mathbf{W}_1 + (1-\theta)\mathbf{W}_2) \\ &= \|(\theta\mathbf{W}_1 + (1-\theta)\mathbf{W}_2)\text{Diag}(\mathbf{c})\|_* \\ &= \|\theta\mathbf{W}_1\text{Diag}(\mathbf{c}) + (1-\theta)\mathbf{W}_2\text{Diag}(\mathbf{c})\|_* \\ &\le \theta\|\mathbf{W}_1\text{Diag}(\mathbf{c})\|_* + (1-\theta)\|\mathbf{W}_2\text{Diag}(\mathbf{c})\|_* \\ &= \theta f(\mathbf{W}_1) + (1-\theta) f(\mathbf{W}_2) \end{aligned}$$

This completes the proof. □

Based on Theorem 1, we propose to an alternating optimization algorithm over $\mathbf{W}$ and $\mathbf{C}$ which is presented in Algorithm 1. To be specific, $\mathbf{W}$ is initialized by performing single task learning over each task. In each alternating step (e.g. $\mathbf{W}$-step or $\mathbf{C}$-step), the alternating direction method of multipliers (ADMM) method [Boyd *et al.*, 2011] is used to solve the convex optimization problem over the respective variable.

5.1 Solving C

With fixed $\mathbf{W}$, since the trace Lasso penalty is non-smooth, we propose to use ADMM for solving the problem. In order to apply ADMM, we rewrite the optimization problem (1) as:

$$\begin{aligned} \min_{\mathbf{C}} \quad & \frac{\lambda}{2}\sum_{i=1}^m \left\|\mathbf{w}_i - \mathbf{W}_{-i}\mathbf{c}_i^{[-i]}\right\|_2^2 + \gamma\sum_{i=1}^m \|\mathbf{J}_i\|_*, \qquad (2) \\ \text{s.t.} \quad & \mathbf{J}_i = \mathbf{W}_{-i}\text{Diag}(\mathbf{c}_i^{[-i]}), \quad \forall i = 1, ..., m. \end{aligned}$$

Algorithm 1 Optimization procedure for solving (1)

```
Input: Data $\mathcal{D}_i$
Initialize: $\mathbf{W}^0, \mathbf{C}^0, q = 0$
while not converge do
    update $\mathbf{C}^{q+1}$ through solving (2) by ADMM. ($\mathbf{C}$-step)
    update $\mathbf{W}^{q+1}$ through solving (4) by ADMM. ($\mathbf{W}$-step)
    $q = q + 1$.
end while
Output: $\mathbf{W}^q, \mathbf{C}^q$
```

This step can be interpreted as learning the task relationships through fixed task parameters. The augmented Lagrangian for (2) is defined as follows ($\{\mathbf{U}_i\}_{i=1}^m$ are introduced as dual variables in scaled form):

$$\begin{aligned} &L_1(\mathbf{C}, \{\mathbf{J}_i\}_{i=1}^m, \{\mathbf{U}_i\}_{i=1}^m) \\ &\quad = \frac{\lambda}{2}\sum_{i=1}^m \left\|\mathbf{w}_i - \mathbf{W}_{-i}\mathbf{c}_i^{[-i]}\right\|_2^2 + \gamma\sum_{i=1}^m \|\mathbf{J}_i\|_* \\ &\quad + \frac{\rho}{2}\sum_{i=1}^m \left\|\mathbf{J}_i - \mathbf{W}_{-i}\text{Diag}(\mathbf{c}_i^{[-i]}) + \mathbf{U}_i\right\|_F^2, \qquad (3) \end{aligned}$$

The ADMM algorithm consists of the following iterations:

(a) $\mathbf{C}^+ \leftarrow \underset{\mathbf{C}}{\text{argmin}}\, L_1(\mathbf{C}, \{\mathbf{J}_i\}_{i=1}^m, \{\mathbf{U}_i\}_{i=1}^m)$,

(b) $\mathbf{J}_i^+ \leftarrow \underset{\mathbf{J}_i}{\text{argmin}}\, L_1(\mathbf{C}^+, \{\mathbf{J}_i\}_{i=1}^m, \{\mathbf{U}_i\}_{i=1}^m),\ \forall i \in \{1, ..., m\}$,

(c) $\mathbf{U}_i^+ \leftarrow \mathbf{U}_i + \mathbf{J}_i^+ - \mathbf{W}_{-i}\text{Diag}(\mathbf{c}^{+[-i]}_i),\ \forall i \in \{1, ..., m\}$,

where $^+$ indicate that the updated version of the corresponding variable. Step (a) has a closed form solution whose form is similar to that of the least square problem, while the closed form solution of step (b) is given by applying matrix soft-thresholding operation.

5.2 Solving W

With fixed $\mathbf{C}$, similarly, ADMM is used to solve the following reformulated optimization problem.

$$\begin{aligned} \min_{\mathbf{W}} \quad & \sum_{i=1}^m \mathcal{L}(\mathbf{w}_i, \mathcal{D}_i) + \frac{\lambda}{2}\sum_{i=1}^m \left\|\mathbf{w}_i - \mathbf{W}_{-i}\mathbf{c}_i^{[-i]}\right\|_2^2 \\ & + \gamma\sum_{i=1}^m \|\mathbf{Z}_i\|_*, \qquad (4) \\ \text{s.t.} \quad & \mathbf{Z}_i = \mathbf{W}_{-i}\text{Diag}(\mathbf{c}_i^{[-i]}) \quad \forall i = \{1, ..., m\} \end{aligned}$$

In this step, task relationship information in $\mathbf{C}$ is used to help learn $\mathbf{W}$. The second term in the objective enforces the linear combination assumption. The trace Lasso term helps with encouraging related tasks to have correlated task parameters. The augmented Lagrangian for (4) becomes:

$$\begin{aligned} &L_2(\mathbf{W}, \{\mathbf{Z}_i\}_{i=1}^m, \{\mathbf{U}_i\}_{i=1}^m) \\ &= \sum_{i=1}^m \mathcal{L}(\mathbf{w}_i, \mathcal{D}_i) + \frac{\lambda}{2}\sum_{i=1}^m \left\|\mathbf{w}_i - \mathbf{W}_{-i}\mathbf{c}_i^{[-i]})\right\|_2^2 \\ &\quad + \gamma\sum_{i=1}^m \|\mathbf{Z}_i\|_* + \frac{\rho}{2}\sum_{i=1}^m \left\|\mathbf{Z}_i - \mathbf{W}_{-i}\text{Diag}(\mathbf{c}_i^{[-i]}) + \mathbf{U}_i\right\|_F^2, \end{aligned}$$

Again $\{\mathbf{U}_i\}_{i=1}^m$ are introduced as dual variables in scaled form. The ADMM algorithm consists of procedure that is similar to that of solving $\mathbf{C}$.

6 Experiments

We perform experiments on both synthetic and real-world datasets. Our proposed method Group Adaptive Multi-Task Learning (**GAMTL**) is compared with the following baselines:

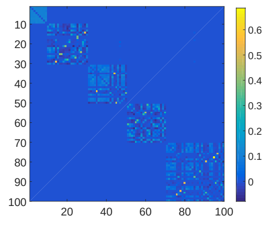
- **STL**: Single task learning method, in which tasks are learned independently.
- **MTRL**: Multi-task relationship learning method proposed in [Zhang and Yeung, 2010] is a MTL method that attempts to learn task covariance matrix and uses it to help MTL learning.
- **AMTL**: Asymmetric multi-task learning method proposed in [Lee *et al.*, 2016]. It assumes that each task is represented by a sparse non-negative linear combination of other tasks, which allows for asymmetric information transfer between tasks. The method also introduces a model with a different training loss model scaled by task outgoing weights in the task correlation coefficient vector. As our proposed model focuses on learning task relatedness, we compare our method with the AMTL-noLoss model in [Lee *et al.*, 2016], which uses the normal training loss term as ours.
- **GO-MTL**: Method proposed in [Kumar and Daumé III, 2012], which allows for task grouping and overlap by learning latent basis tasks. Each task is represented by a sparse linear combination of the latent basis tasks.

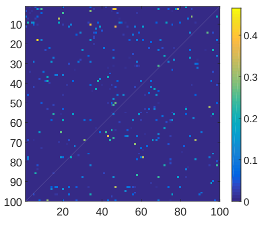
6.1 Synthetic Dataset

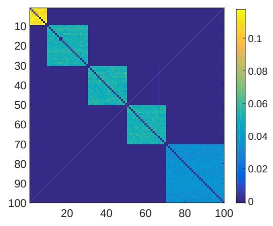
To test the effectiveness of our proposed method in terms of adaptive grouping and generalization performance, we evaluate our method on two synthetic datasets. The first synthetic dataset, denoted by **Synthetic**, consists of 100 classification tasks, with 5 clusters of tasks. Size of each cluster is set to 10, 20, 20, 20, 30, respectively. In each cluster, we first randomly generate 3$\sim$5 correlated tasks, and then generate the rest tasks by linearly combining those tasks. The second synthetic dataset, denoted by **Synthetic-pos**, is generated in the same manner but with only positive linear combination in terms of task relationships. This dataset is for better comparison with AMTL which only learns positive task relations. Figure 1 shows the learned task correlation matrix of our method with comparison to ATML. Our method learns the underlying group structure of task correlations precisely. In comparison, ATML could only learn a sparse correlation, but fails to recover the task grouping structure. Our method also has a better performance in terms of prediction error rate on the testing set: {7.55%, 1.56%} compared with {17.43%, 17.53%} for AMTL.

6.2 Real-world Datasets

Next, we evaluate our proposed method on the following real-world datasets. **School**: this is a regression dataset which is consisted of exam scores of students from 139 schools. Each


(a) Learned with GAMTL


(b) Learned with AMTL


(c) Learned with GAMTL

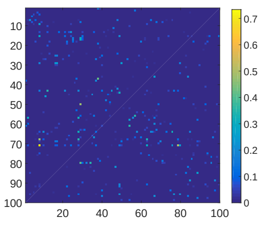

(d) Learned with AMTL

Figure 1: Comparison of learned task correlation matrix

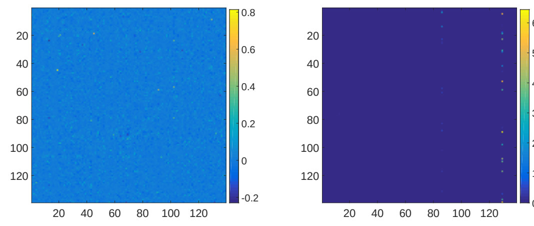
school corresponds to a task. By adding 1 to the end of all data to account for the bias term, each data point has 26 features. We use the splits given by [Argyriou *et al.*, 2008a] for training and testing. **MDS** [Blitzer *et al.*, 2007]: this is a dataset of product reviews on 25 domains (apparel, books, DVD, etc.) crawled from Amazon.com. We delete three domains with less than 100 instances and make it a MTL problem with 22 tasks. Each task is a sentiment classification task that classifies a review as negative or positive. The number of instances per task of the original dataset varies from 314 to 20,751. In our dataset, for tasks with more than 500 instances, 500 instances are randomly drawn to form the reduced-size tasks. Training and testing samples are obtained using a 30%-70% split. **CIFAR** [Krizhevsky, 2009]: CIFAR10 is an image classification dataset with 10 classes. We generate $\binom{10}{2} = 45$ tasks , each of which is a one v.s. one classification task. For example, task '1 v.s. 2' means we use class 1 and class 2 to construct a binary classification task. The rest of the tasks are constructed in a similar fashion. In each task, 120 training data points are randomly drawn equally from the two classes, the rest are used as testing data.

Table 1 shows the prediction performance of our proposed method (GAMTL) and other baselines on real-world datasets. It could be observed that except School dataset, our proposed GAMTL method could outperform other state of art methods. On school dataset, GO-MTL gets the best prediction performance. Our method and AMTL has similar performance. This is also observed in [Lee *et al.*, 2016]. On School dataset, all tasks are highly correlated with each other as they are solving the same problem (score prediction). There is less task group structure information that could be utilized by our method. This is also verified in our learned task correlation matrix on School dataset as shown in Figure 2. As can be seen from the figure, there is no obvious group structure.

On MDS dataset, our method performs the best among all

Table 1: Comparison performance on real-world datasets

Method	School	MDS	CIFAR
STL	10.39 ± 0.12	$28.77 \pm 0.56\%$	$19.01 \pm 0.43\%$
MTRL	10.30 ± 0.08	$12.92 \pm 0.23\%$	$16.70 \pm 0.19\%$
AMTL	10.23 ± 0.10	$14.31 \pm 0.29\%$	$18.66 \pm 0.22\%$
GO-MTL	$\mathbf{10.02 \pm 0.09}$	$17.01 \pm 0.38\%$	$17.03 \pm 0.34\%$
GAMTL	10.25 ± 0.09	$\mathbf{12.82 \pm 0.25\%}$	$\mathbf{16.63 \pm 0.21\%}$



(a) GAMTL(School) (b) AMTL(School)

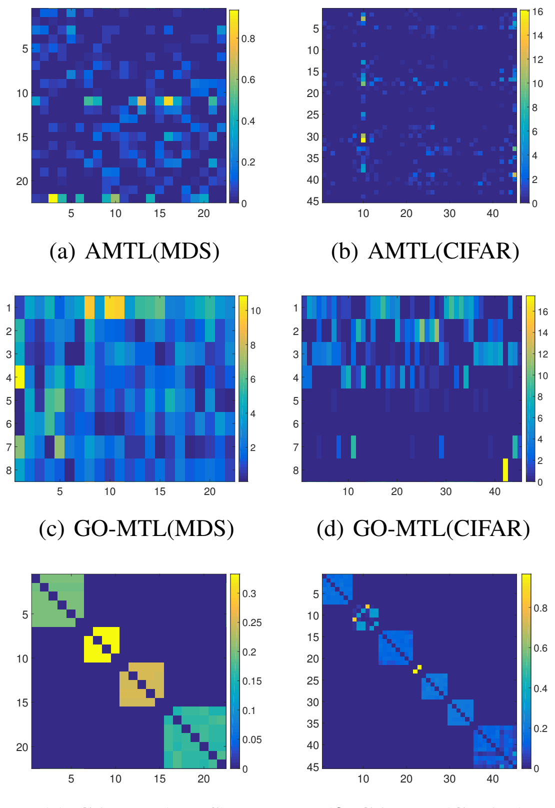
Figure 2: Learnd task correlation matrix on School

the methods. Figures 3(a), 3(c), 3(e) visualize of the learned task correlation with reordered task index. For AMTL and our method, we plot the task correlation matrix $\mathbf{C}$ as described in section 4. For GO-MTL, we plot the weights of linear combination of latent tasks (denoted by matrix $\mathbf{S}$ in their paper). The x-axis denotes the weights of linear combination, and the y-axis denotes the latent tasks index. We see from the figures that our method has learned useful information about the task grouping in MDS dataset, while the other two baseline methods fail to learn a clear grouping structure. According to the learned task correlation matrix of our method, the 22 tasks are grouped into 4 clusters shown as follows:

Group 1	books, computer & video games, dvd, magazines, music, videos
Group 2	camera & photo, cell phones & service, electronics, software
Group 3	apparel, beauty, gourmet food, grocery, jewelry & watches
Group 4	automotive, baby, healthy & personal care, kitchen & housewares, outdoor living, sports & outdoor, toys & games

The learned task grouping turns out to be meaningful, especially for the first two clusters. The first cluster consists of tasks that are products from cultural industries, while the second cluster contains tasks about electronics and software. Tasks in the third cluster are mostly products that people will pay more attention to their visual aesthetics. Tasks in the fourth cluster are less relevant as a group but some of them share common relevance with each other. The degree of relevance shown in the learned task correlation matrix is also consistent with the actual relevance of those tasks within the cluster. Tasks within first three clusters have a larger relevance compared to those in the fourth cluster according to the learned task correlation matrix.

On CIFAR dataset, our method obtains the best prediction performance compared to other baseline methods. Figures 3(b), 3(d), 3(f) gives a visualization of the learned task correlation. The original task order are defined in the task description section. In the visualization, the tasks index are reordered. Similarly, as shown in the figures, we find that our method is able to learn meaningful task group structure. Due to limitation of space, we could not present the full grouping information. To give an example, the first cluster learned by GAMTL includes tasks '2 vs 9', '3 vs 9', '4 vs 9', '5 vs 9', '6 vs 9', '7 vs 9', and '8 vs 9'. In comparison, the task correlation learned by the other two methods (AMTL and GO-MTL) do not have very clear task grouping structure.



(a) AMTL(MDS) (b) AMTL(CIFAR)

(c) GO-MTL(MDS) (d) GO-MTL(CIFAR)

(e) GAMTL(MDS) (f) GAMTL(CIFAR)

Figure 3: Learned task correlation on MDS and CIFAR

In summary, experiments have shown that our method is effective in adaptively learning the underlying task grouping structure (if there is) in MTL. When there exists task grouping structure in the MTL problem, our method can exploit the task grouping formation to help improve the generalization.

7 Conclusion

We propose a MTL method to adaptively learn grouping structure among tasks for improving generalization performance of related tasks. Efficient algorithm is proposed to solve the proposed optimization problem. Experimental results have demonstrated the effectiveness of our method compared with other state-of-the-art methods.

Acknowledgements

This work is supported by the NTU Singapore Nanyang Assistant Professorship (NAP) grant M4081532.020. The authors would like to thank Yaodong Yu for helpful discussions.

References

[Ando and Zhang, 2005] Rie Kubota Ando and Tong Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *JMLR*, 6(Nov):1817–1853, 2005.

[Argyriou *et al.*, 2008a] Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. Convex multi-task feature learning. *Machine Learning*, 73(3):243–272, 2008.

[Argyriou *et al.*, 2008b] Andreas Argyriou, Andreas Maurer, and Massimiliano Pontil. An algorithm for transfer learning in a heterogeneous environment. In *ECML-PKDD*, pages 71–85. Springer, 2008.

[Blitzer *et al.*, 2007] John Blitzer, Mark Dredze, Fernando Pereira, et al. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *ACL*, pages 440–447, 2007.

[Boyd *et al.*, 2011] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122, 2011.

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

[Chen *et al.*, 2011] Jianhui Chen, Jiayu Zhou, and Jieping Ye. Integrating low-rank and group-sparse structures for robust multi-task learning. In *KDD*, pages 42–50. ACM, 2011.

[Evgeniou and Pontil, 2004] Theodoros Evgeniou and Massimiliano Pontil. Regularized multi–task learning. In *KDD*, pages 109–117. ACM, 2004.

[Grave *et al.*, 2011] Edouard Grave, Guillaume R Obozinski, and Francis R Bach. Trace lasso: a trace norm regularization for correlated designs. In *NIPS*, pages 2187–2195, 2011.

[Jacob *et al.*, 2009] Laurent Jacob, Jean-philippe Vert, and Francis R Bach. Clustered multi-task learning: A convex formulation. In *NIPS*, pages 745–752, 2009.

[Kang *et al.*, 2011] Zhuoliang Kang, Kristen Grauman, and Fei Sha. Learning with whom to share in multi-task feature learning. In *ICML*, pages 521–528, 2011.

[Krizhevsky, 2009] Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009.

[Kumar and Daumé III, 2012] Abhishek Kumar and Hal Daumé III. Learning task grouping and overlap in multitask learning. In *ICML*, 2012.

[Lee *et al.*, 2016] Giwoong Lee, Eunho Yang, et al. Asymmetric multi-task learning based on task relatedness and confidence. In *ICML*, pages 230–238, 2016.

[Liu *et al.*, 2009] Jun Liu, Shuiwang Ji, and Jieping Ye. Multi-task feature learning via efficient $\ell_{2,1}$-norm minimization. In *UAI*, pages 339–348, 2009.

[Pan and Yang, 2010] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE TKDE*, 22(10):1345–1359, October 2010.

[Rai *et al.*, 2012] Piyush Rai, Abhishek Kumar, and Hal Daume. Simultaneously leveraging output and task structures for multiple-output regression. In *NIPS*, pages 3185–3193, 2012.

[Rosenstein *et al.*, 2005] Michael T. Rosenstein, Zvika Marx, and Leslie Pack Kaelbling. To transfer or not to transfer. In *NIPS-05 Workshop on Inductive Transfer: 10 Years Later*, December 2005.

[Zhang and Schneider, 2010] Yi Zhang and Jeff G Schneider. Learning multiple tasks with a sparse matrix-normal penalty. In *NIPS*, pages 2550–2558, 2010.

[Zhang and Yang, 2017] Yu Zhang and Qiang Yang. Learning sparse task relations in multi-task learning. In *AAAI*, 2017.

[Zhang and Yeung, 2010] Yu Zhang and Dit-Yan Yeung. A convex formulation for learning task relationships in multi-task learning. In *UAI*, pages 733–442, 2010.

[Zhou and Zhao, 2016] Qiang Zhou and Qi Zhao. Flexible clustered multi-task learning by learning representative tasks. *IEEE TPAMI*, 38(2):266–278, 2016.

[Zhou *et al.*, 2011] Jiayu Zhou, Jianhui Chen, and Jieping Ye. Clustered multi-task learning via alternating structure optimization. In *NIPS*, pages 702–710, 2011.