Tree Guided Learning for Structured Sparsity

Avinash N Bukkittu
Department of Computer Science
Columbia University
ab4377@columbia.edu

Varsha G. Maragi
Department of Computer Science
Columbia University
vgm2115@columbia.edu

Abstract

Exploiting structured sparsity in high dimensional dataset is highly important to perform efficient supervised learning tasks. Traditional methods include applying l_1 norm to induce sparsity. However, this fails to capture the inherent structure among the features. The classical group lasso solves this problem by capturing the group information of the features. However, this approach does not allow overlapping of features which restricts the application of this method. In this paper, we compare the results of tree-guided group lasso with lasso in high-dimensional data settings. In tree-guided regularization, we assume that the input variables are organized into groups in the form of a tree where the groups represented by the parent nodes form a superset of the groups represented by the child nodes. Thus, it is a form of overlapping group lasso with a hierarchical structure. This paper also explores multi-task learning by considering a tree structure over the output variables. Our experiments demonstrate that assuming a structure over either the input features or the output variables can improve the accuracy of supervised learning tasks.

1 Introduction

Many of the machine learning problems can be formulated as the following minimization problem

$$\min_{\boldsymbol{\beta}} l(\boldsymbol{\beta}) + \lambda\phi(\boldsymbol{\beta}) \tag{1}$$

In the above equation, $l(\boldsymbol{\beta})$ function represents any loss function. For example,

1. For least square loss, we have $l(\boldsymbol{\beta}) = \frac{1}{2}||X\boldsymbol{\beta} - \boldsymbol{y}||_2^2$
2. For logistic loss, we have $l(\boldsymbol{\beta}) = \sum_{i=1}^{m} \log(1 + \exp(-y_i(< \boldsymbol{\beta}, \boldsymbol{x_i} >)))$ and various others.

$\phi(\boldsymbol{\beta})$ is the regularizer term. λ controls the amount of regularization.

In high-dimensional data applications, we have $p >> n$, where p is the dimension of the data and n is the number of samples. In such applications not all features are important and we strive for a sparse solution. The traditional method for sparse learning is to introduce l_1 penalty (Lasso penalty) in the objective function. However, the lasso penalty fails to capture the inherent structure among the features.

l_1 penalty finds a solution with few non-zero entries. Suppose we can group our features into different groups like the grouping of genes into functional groups. In such applications, we would like to gain sparsity among these groups. We can use the group lasso to solve such problems [1]. The group lasso employs the $l_{1,2}$ penalty. For group lasso, we take l_2 norm of the features within each group. We then take l_1 norm of the groups. This ensures that all the features of the irrelevant groups have their coefficient equated to zero. The key point to be noted here is that the groups are assumed to have a

non-overlapping structure. It is assumed that the features belonging to one group are not related to the features belonging to another group. This could be an unreasonable assumption in many applications. Hence, we need an extension to group lasso which can handle overlapping features. This motivated us to explore the tree guided group lasso solution [6].

In many applications, features can be naturally represented in a hierarchical structure. The tree guided group lasso assumes a tree structure among the features. A major concern with this kind of optimization is that it has a complex regularization term which is briefly explained in Section 2.

In image classification application like facial expression classification,as given in [6], the pixels of an image can be represented as a tree. Each node of the tree represents a block of pixels. The leaf nodes represent a single pixel, which indicate one feature of the image. The parent node represents a block of pixels which belong to one particular spatial locality. We expand this structure detection to multi-task regression where we assume a tree structure over the output variables. We performed experiments on the JAFFE dataset [8]. The dataset contains facial expressions of 10 Japanese subject and represent 6 emotions(excluding Neutral expression). The task is to classify the images into different expressions of the Japanese subject. The image features can be categorized into different groups based on the locality information. The results show that using tree-guided group lasso gives a better performance when compared to lasso penalty. We extended the idea to capture structured sparsity among output variables by considering a tree structure on the 6 different emotions. We found that the idea works well in this setting as well.

2 Technical Details

We begin with the definition of an index tree.

Definition 2.1. Consider a tree T of depth d. Let $T_i = \{G_1^i, G_2^i, ..., G_{n_i}^i\}$ be the nodes of the tree at the level i. Each node G_j^i holds a set of indices of the leaf nodes that are part of the subtree contained in the subtree rooted at G_j^i. We denote G_j^i to indicate the set of leaf nodes contained in that subtree. We call this set as index set. The tree satisfies the following conditions (1) The nodes at the same level have non-overlapping set of leaf nodes i.e $G_j^i \cap G_k^i = \phi, \forall i \in \{1, 2, ..., d\}, j \neq k, 1 \leq j, k \leq n_i$ (2). The set of leaf nodes contained in the parent node is the superset of any of its child nodes.

Figure 1 shows an example of an index tree. Here, $\{G_1^2, ..., G_7^2\}$ are the leaf nodes indicating the features. $G_1^1 = \{G_1^2, G_2^2\}$ is the grouping of the first two variables. $G_2^1 = \{G_3^2, G_4^2\}$ are the grouping of the next two variables and so on. It is important to note that the nodes at the same level have non-overlapping set of indices and that the index set of the parent node is the superset of the child nodes.

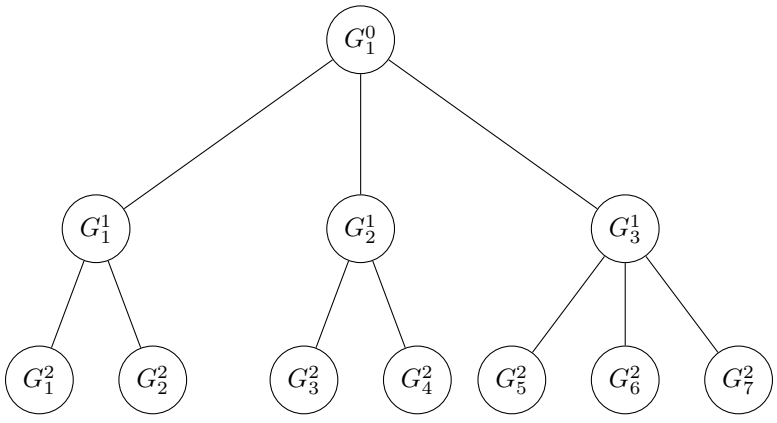

Figure 1: An example for tree-guided regularization. Here, $G_1^2 = \{\beta_1\}$, $G_2^2 = \{\beta_2\}$, $G_2^2 = \{\beta_3\}$,...,$G_7^2 = \{\beta_7\}$, $G_1^1 = \{\beta_1, \beta_2\}$ and so on.

The grouped-tree regularization is defined as

$$\phi(\boldsymbol{\beta}) = \sum_{i=0}^{d} \sum_{j=1}^{n_i} w_j^i ||\boldsymbol{\beta}_{G_j^i}||_2 \tag{2}$$

where $\boldsymbol{\beta} \in \mathbf{R}^p, w_j^i \geq 0, i = \{0, 1, 2..., d\}, j = \{1, 2, ..., n_i\}$ is the pre-defined weight.

In the next section, we explain a method to solve (1) using the Moreau-Yosida regularization associated with the tree-guided regularization (2).

2.1 Moreau-Yosida Regularization of $\phi()$

The Moreau-Yosida regularization associated with (2) for a given vector $\boldsymbol{v} \in \mathbb{R}^p$ is given by

$$\phi_\lambda(\boldsymbol{v}) = \min_{\beta} \frac{1}{2}||\boldsymbol{\beta} - \boldsymbol{v}||_2^2 + \lambda \sum_{i=0}^{d} \sum_{j=1}^{n_i} w_j^i ||\boldsymbol{\beta}_{G_j^i}|| \tag{3}$$

(3) has many interesting properties one of which is that $\phi_\lambda((v))$ is continuously differentiable. [3], [4] explain the properties in detail. The works by J.Liu et. al in [7] show that the efficient optimization of Moreau-Yosida optimization is the key to solving to many convex optimization problems. We now explain an algorithm developed in [6] that gives an analytical solution to (3). In the algorithm below, we have assumed $\lambda_j^i = \lambda w_j^i$

1: **function** MOREAU-YOSIDA-SOLVER($\boldsymbol{v}$,G,λ)
2: $\quad \boldsymbol{u}^{d+1} \leftarrow \boldsymbol{v}$
3: $\quad$ **for** $i = d$ to 0 **do**
4: $\quad\quad$ **for** $j = 1$ to n_i **do**
5: $\quad\quad\quad$ **if** $||\boldsymbol{u}_{\boldsymbol{G_j^i}}^{\boldsymbol{i+1}}|| \leq \lambda_j^i$ **then**
6: $\quad\quad\quad\quad \boldsymbol{u}_{G^i}^i \leftarrow \mathbf{0}$
7: $\quad\quad\quad$ **else**
8: $\quad\quad\quad\quad \boldsymbol{u}_{G_j^i}^i \leftarrow \frac{||\boldsymbol{u}_{G_j^i}^{i+1} - \lambda_j^i||}{||\boldsymbol{u}_{G_j^i}^{i+1}||} \boldsymbol{u}_{G_j^i}^{i+1}$
9: $\quad$ **return** $\boldsymbol{u}^0$

As can be seen in the above pseudo-code, the algorithm only needs to maintain the most recent value of $\boldsymbol{u}$ which is initialized to the input variable $\boldsymbol{v}$. The algorithm then traverses the tree in reverse-breadth first order, each time updating the variable $\boldsymbol{u}$ according to lines 5-8.

The time complexity of the above algorithm is $T = O(\sum_{i=0}^{d} \sum_{j=1}^{n_i} |G_j^i|)$ where d is the depth of the tree and n_i is the number of nodes at level i. If p is the total number of features in the tree, then $|G_j^i| \leq p$ and the time complexity becomes $O(pd)$. If the tree is balanced then $d = O(\log(p))$ then $T = O(p \log(p))$.

Let us explain the above algorithm for Figure 1. For simplicity assume $\lambda = \sqrt{2}, w_j^i = 1 \Rightarrow \lambda_j^i = \sqrt{2}$. Let the initial vector be $[1, 1, 2, 2, 4, 4, 2]^T$. Line-2 sets $\boldsymbol{u}^3 = [1, 1, 2, 2, 4, 4, 2]^T$. After updating $\boldsymbol{u}$ for $d = 2$, we have $\boldsymbol{u}^2 = [0, 0, 2 - \sqrt{2}, 2 - \sqrt{2}, 4 - \sqrt{2}, 4 - \sqrt{2}, 2 - \sqrt{2}]^T$. After processing for $d = 1$, we have $\boldsymbol{u}^1 = [0, 0, 0, 0, 1.5984, 1.5894, 0.3621]^T$. Finally for $d = 0$, we have $\boldsymbol{u}^0 = [0, 0, 0, 0, 0.611, 0.611, 0.1384]$. It is important to note the final solution is sparse and variables within a group, namely G_3^1 are all non-zero. λ is the regularization parameter and it can be noted that as λ increases, the entries in the solution are monotonically decreasing.

After finding the minimizer to (3), we can apply existing methods like gradient descent or its derivatives like accelerated gradient descent to solve (1). [9] explains discusses methods for solving optimization objectives involving grouped and hierarchical variables.

The work by J. Lui et. al. [6] provides a mathematical treatment to the above algorithm and proves that the algorithm finds an exact minimizer to (3).

In the rest of the paper, we refer to the tree-guided regularized objective function as tgLasso [6].

3 Experimental Results

We have conducted experiments on the JAFFE dataset [8]. It contains images of 10 Japanese subjects totaling to 213 images. These images portray 6 different emotions(excluding Neutral expression). The expressions include Happy, Sad, Surprise, Disgust, Anger and Fear. The task is to classify these images into 6 expressions. Experts have classified each of the images into one of the six categories and have also rated each image on a scale of 5 for each expression. We have resized each image to 64 $\times$ 64 pixels and hence a feature size of 4096. Each image can be divided into blocks as shown in figures 2, 3 & 4.

Figure 2: Original Image

Figure 3: Divided into 4x4 blocks

Figure 4: Divided into 16x16 blocks

Each block represents a node in the tree. Every block is again divided into sub-blocks which serve as the child node of the parent block. The leaf nodes represent each pixel of the image. We have used the SLEP [5] package to perform a tree-guided regularization on this setup. The first task was to compare the tgLasso results, using least square loss function with that of Lasso. We used a 8:2 ratio for training and testing as stated in [6]. The 2 test subjects were independent of the training subjects. This leads to a challenging supervised learning task. We have shown the results for all six expressions. We compared the RMSE values of the predicted values with the true values. The RMSE values were averaged for 10 runs with random selection of test and train subjects. We plotted the RMSE values for different λ for both Lasso and tgLasso. The results are shown in Figure 5.

Next, we considered the binary classification task and used the logistic loss function. We used a Balanced Error Rate(BER) to compare the results between lasso and the tgLasso. The Balanced Error Rate [2] was used to cope with unbalanced positive and negative samples. The results are shown in Figure 6.

So far we imposed a tree structured on the features for a regression/classification task. We extended this idea to multi-task regression models where the k-regression tasks are related via a tree like structure. The output variables,i.e, the 6 different expressions can be assumed to be related as shown in Figure 7.

We performed tree guided regularization assuming this tree structure and compared it with lasso. We have used the least square loss and plotted the RMSE values similar to the first experiment. The results are shown in Figure 8.

These results prove that tree guided group lasso outperforms lasso in all three cases. We can also conclude that structure can be assumed over input as well as output features. One of the reasons why we think tgLasso performs better than Lasso is that it is efficient in capturing *block sparsity* as compared to Lasso which assumes no structure over the inputs. The source code for the experiments is available online at https://github.com/VarshaMaragi/advanced-machine-learning-project.

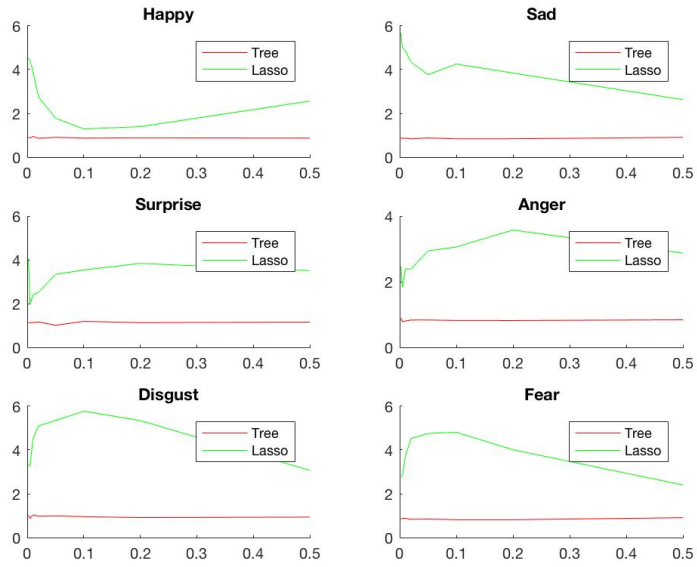

Figure 5: Tree-Guided Lasso vs Lasso

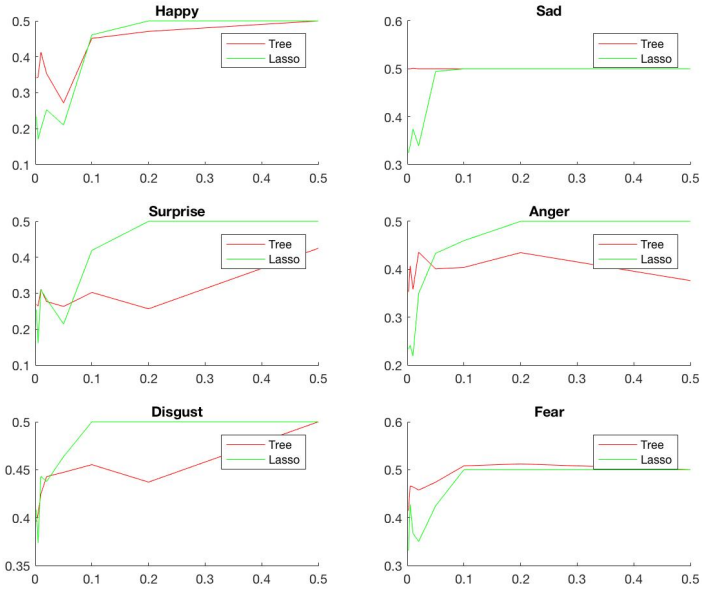

Figure 6: Tree- Guided Lasso vs Lasso

4 Conclusion and Future Work

In this paper, we explain a method to solve tree-guided regularized optimization functions. The key idea was to minimize the associated Moreau-Yosida regularization of the tree-guided penalty. We showed that exploiting structure in high dimensional data for structured sparsity can lead to better results than simply striving for a sparse solution. Our experiments confirmed that imposing a tree structure over the features is better at detecting facial expressions than just a sparse solution. In particular, we compared the tree-guided group regularization with lasso for square loss and logistic

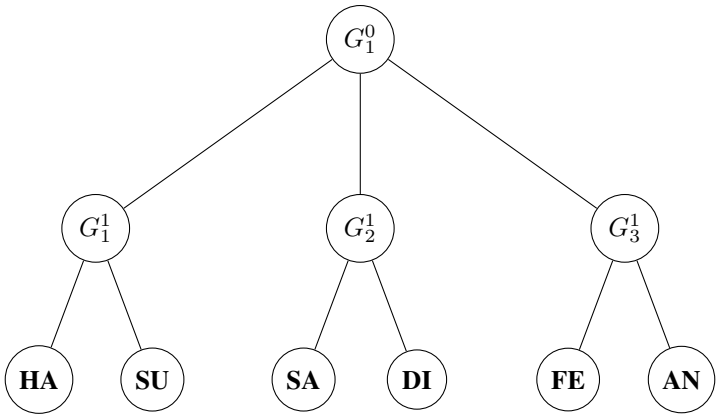

Figure 7: Multi-task learning applied to JAFFE dataset. Similar expressions are grouped together.

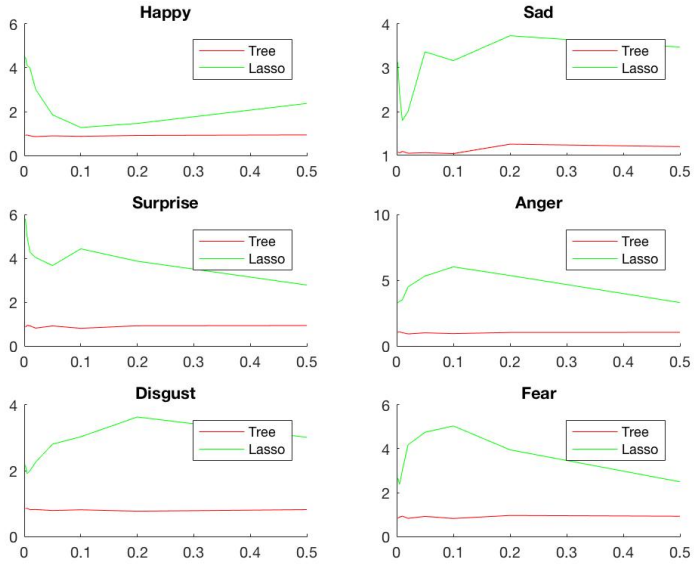

Figure 8: Tree- Guided Lasso vs Lasso for Multitask learning

loss. For square loss, we plotted the RMSE values for different values of λ and for logistic loss we plotted the balanced error rate with different values of λ . The results proved that on an average the tgLasso showed an improvement over the traditional Lasso. We extended our work to multi-task regression setting where we assumed a tree structure over the output variables. A tree structure over the facial expressions performed better than the lasso penalty (assuming no structure among the output variables). An interesting area to look into would be to incorporate the structure of both input variables and output variables for predictions tasks. We would like to extend this work for face recognition applications along with face expression detection. We would also like to apply tree-guided regularization in the frequency space to see if structure sparsity can provide valuable insights among the features.

References

[1] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. A note on the group lasso and a sparse group lasso. *arXiv preprint arXiv:1001.0736*, 2010.

[2] Isabelle Guyon, Steve R Gunn, Asa Ben-Hur, and Gideon Dror. Result analysis of the nips 2003 feature selection challenge. In *NIPS*, volume 4, pages 545–552, 2004.

[3] Jean-Baptiste Hiriart-Urruty and Claude Lemaréchal. *Convex analysis and minimization algorithms I: Fundamentals*, volume 305. Springer science & business media, 2013.

[4] Claude Lemaréchal and Claudia Sagastizábal. Practical aspects of the moreau–yosida regularization: Theoretical preliminaries. *SIAM Journal on Optimization*, 7(2):367–385, 1997.

[5] J. Liu, S. Ji, and J. Ye. *SLEP: Sparse Learning with Efficient Projections*. Arizona State University, 2009.

[6] Jun Liu and Jieping Ye. Moreau-yosida regularization for grouped tree structure learning. In *Advances in Neural Information Processing Systems*, pages 1459–1467, 2010.

[7] Jun Liu, Lei Yuan, and Jieping Ye. An efficient algorithm for a class of fused lasso problems. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 323–332. ACM, 2010.

[8] Miyuki Kamachi Jiro Gyoba. Michael J. Lyons, Shigeru Akemastu. Coding facial expressions with gabor wavelets. *3rd IEEE International Conference on Automatic Face and Gesture Recognition*, pages pp. 200–205, 1998.

[9] Peng Zhao, Guilherme Rocha, and Bin Yu. The composite absolute penalties family for grouped and hierarchical variable selection. *The Annals of Statistics*, pages 3468–3497, 2009.