Learning (from) Deep Hierarchical Structure among Features

Yu Zhang\textsuperscript{1}, Lei Han\textsuperscript{2}
\textsuperscript{1}HKUST, \textsuperscript{2}Tencent AI Lab
yu.zhang.ust@gmail.com, leihan.cs@gmail.com

Abstract

Data features usually can be organized in a hierarchical structure to reflect the relationships among them. Most of previous studies that utilize the hierarchical structure to help improve the performance of supervised learning tasks can only handle the structure of a limited height such as 2. In this paper, we propose a Deep Hierarchical Structure (DHS) method to handle the hierarchical structure of an arbitrary height with a convex objective function. The DHS method relies on the exponents of the edge weights in the hierarchical structure but the exponents need to be given by users or set to be identical by default, which may be suboptimal. Based on the DHS method, we propose a variant to learn the exponents from data. Moreover, we consider a case where even the hierarchical structure is not available. Based on the DHS method, we propose a Learning Deep Hierarchical Structure (LDHS) method which can learn the hierarchical structure via a generalized fused-Lasso regularizer and a proposed sequential constraint. All the optimization problems are solved by proximal methods where each subproblem has an efficient solution. Experiments on synthetic and real-world datasets show the effectiveness of the proposed methods.

Introduction

Most of previous studies to utilize the hierarchical structure among features, including the group Lasso (Yuan and Lin 2006) and the Hierarchical Penalization (HP) method (Szafranski, Grandvalet, and Morizet-Mahoudeaux 2007), can only handle the hierarchical structure of a limited height up to 2. In this paper, we aim to break this assumption by utilizing the available hierarchical structure with an arbitrary height to help learn an accurate model. Moreover, in some case where the hierarchical structure is unavailable, we aim to learn such hierarchical structure among features to improve the interpretability of the resultant learner.

Specifically, given a hierarchical structure to describe relations among features, we propose a Deep Hierarchical Structure (DHS) method to utilize it. In the DHS method, each model parameter corresponding to a data feature is penalized by the product of edge weights along the path from the root to the leaf node for that feature. Interestingly, when the exponents of the edge weights along the path from the root to each leaf node are summed to 1, the proposed objective function can be proved to be convex no matter what the height of the hierarchical structure is. Moreover, when all the exponents take the same value, we can show that the proposed objective function is equivalent to a problem with a hierarchical group lasso regularization term. In order to optimize the objective function of the DHS method, we adopt the FISTA algorithm (Beck and Teboulle 2009) each of whose subproblems has an efficiently analytical solution. Moreover, in the proposed DHS method, the exponents of the edge weights need to be set based on a priori information. When this information is not available, by default we just set them to be identical. Usually this strategy works but it may be suboptimal. In order to alleviate this problem, we propose a variant of the DHS method to learn the exponents from data.

Moreover, we consider a more general case where the hierarchical structure is not available. A hierarchical structure can give us more insight about the relations among features but learning it from data is a difficult problem. To the best of our knowledge, there is no work to directly learn the hierarchical structure among features. Here we give the first try based on the DHS method by proposing a Learning Deep Hierarchical Structure (LDHS) method. Given the height of the hierarchical structure, the LDHS method assumes that each path from the root to a leaf node corresponding to a data feature does not share any node between each other, then uses a generalized fused-Lasso regularizer to enforce nodes to fuse at each height, and finally designs a sequential constraint to make the learned structure form a hierarchical structure. For optimization, we use the GIST algorithm (Gong et al. 2013) to solve the objective function of the LDHS method. By comparing with several state-of-the-art baseline methods, experiments on several synthetic and real-world datasets show the effectiveness of the proposed models.

Related Work

The Composite Absolute Penalties (CAP) method (Zhao, Rocha, and Yu 2009) learns from the hierarchical structure but via the group sparsity and its objective function is different from those of the proposed methods. Moreover, the proposed LDHS method can learn the hierarchical structure but the CAP method cannot. The tree-guided group Lasso proposed in (Kim and Xing 2010) can learn from the hierarchical structure for multi-output regression, whose settings are different from ours. There are some
Suppose the training dataset is denoted by \( \mathcal{D} = \{(x_i, y_i)\}_{i=1}^{n} \) where \( x_i \in \mathbb{R}^d \) denotes the \( i \)th data instance and \( y_i \) is its label, and the linear learning function is denoted by \( f(x) = w^T x \). Suppose that the features are organized in a balanced hierarchical structure of height \( m \) where \( m \geq 2 \). Based on a loss function \( l(\cdot, \cdot) \), the objective function of the DHS method is formulated as

\[
\min_{w, \sigma} \frac{1}{n} \sum_{i=1}^{n} l(x_i, y_i, w) + \frac{\lambda_1}{2} \sum_{i=1}^{d} \frac{w_i^2}{\prod_{j=1}^{m} \left( \sigma_{p_{i-1}^{(j)}, p_i^{(j)}} \right)^{\theta_{i,j}^{(j)}}} + \frac{\lambda_2}{2} \| w \|_2^2
\]

s.t. \( \sum_{j=1}^{s_i} d_{i,j} \sigma_{p_{i,j}^{(j)}, p_{i}^{(j)}} = 1 \forall i \in [m] \), \( \sigma_{p_{i,j}^{(j)}, p_{i}^{(j)}} \geq 0 \forall i, j \),

(1)

where \( w_i \) is the \( i \)th entry in \( w \), an edge weight \( \sigma_{p_{i,j}^{(j)}, p_{i}^{(j)}} \) is defined in the previous section, \( \| \cdot \|_2 \) denotes the \( \ell_2 \) norm of a vector, \( a/b \) for two scalars \( a \) and \( b \) is defined by continuation at zero as \( a/0 = \infty \) if \( a \neq 0 \) and \( 0/0 = 0 \), \( [m] \) denotes an integer set from 1 to \( m \), and \( \theta_{i,j}^{(j)} \) exponent, can be viewed as the importance for the edge weight \( \sigma_{p_{i,j}^{(j)}, p_{i}^{(j)}} \).

The summand in the second term of the objective function in problem (1) penalizes each \( w_i \) based on all the weights of edges on the path from the root node to the \( i \)th leaf node as well as the exponents. Hence two coefficients \( w_i \) and \( w_j \) will tend to have similar penalizations if they share many edges in the hierarchical structure. As we will see in Theorem 3, when exponents are identical to each other, this term is related to the group Lasso regularizer which enforces the group sparsity. The equality constraint in problem (1) is to restrict the scale of \( \sigma \). To preserve the convexity of problem (1) as we will see in the next section, it is required that \( \theta_{i,j}^{(j)} \geq 0 \forall i, j \) and \( \sum_{j=1}^{s_i} \theta_{i,j}^{(j)} = 1 \forall i \), which means that the sum of the nonnegative exponents of all the edges along a path from the root to each leaf node equals 1.
Properties

We first introduce a new family of convex functions with the proof in the supplementary material.

**Theorem 1** \( f(w, z) = \frac{w^2}{\prod_{i=1}^{m} z_i} \) is jointly convex with respect to \( w \in \mathbb{R} \) and \( z = (z_1, \ldots, z_m)^T \in \mathbb{R}^m \), where \( z_i \)'s are required to be positive, given that \( \theta_i \geq 0 \) for \( i = 1, \ldots, m \) and \( \sum_{i=1}^{m} \theta_i = 1 \).

When \( m = 1 \), Theorem 1 asserts that \( f(w, z) = w^2/z \) is jointly convex with respect to \( w \) and \( z \) when \( z > 0 \), which is a well-known result (pp. 72, (Boyd and Vandenberghe 2004)). When \( m = 2 \) and \( \theta_1 = \theta_2 = 1/2 \), Theorem 1 recovers Proposition 1 in (Szafranski, Grandvalet, and Morizet-Mahoudeaux 2007). Theorem 1 is more general since \( m \) can be any positive integer and different \( \theta_i \)'s can have different values.

Based on Theorem 1, we can prove the convexity of problem (1) in the following theorem.

**Theorem 2** Given that the loss function \( l(x, y, w) \) is convex with \( w \), problem (1) is jointly convex with respect to \( w \) and \( \sigma \).

To see the effect of the regularizer in the second term of problem (1), we investigate two special cases, where \( m \) equals 2 or 3. When \( m = 2 \), problem (1) degenerates to the HP method (Szafranski, Grandvalet, and Morizet-Mahoudeaux 2007), which shows that when all the \( \theta_{p,i} = \frac{1}{2} \), problem (1) is equivalent to the \( \ell_2,1 \)-regularized group Lasso. When \( m = 3 \), we can derive an equivalent formulation of problem (1) as follows.

**Theorem 3** When \( m = 3 \) and all the \( \theta_{p,i} = \frac{1}{2} \), problem (1) is equivalent to this problem:

\[
\min_{w, \lambda_1} \frac{\lambda_1}{2} \left( \sum_{i=1}^{n} (d_i^{(1)})^2 \left( \sum_{j \in c_i^1} (d_j^{(2)})^2 \left( \sum_{k \in c_i^2} |w_k|^\frac{3}{2} \right) \right)^{\frac{2}{3}} \right)^2 + \frac{1}{n} \sum_{i=1}^{n} l(x_i, y_i, w_i) + \frac{\lambda_2}{2} ||w||_2^2.
\]

(2)

According to Theorem 3, we can see the second term in the objective function of problem (1) can be converted to the first one of problem (2), which places the \( \ell_2 \) norm on model parameters corresponding to the leaf nodes which share the same parent node, then places the weighted \( \ell_2 \) norm on the internal nodes at height 2 which have the same parent node, and finally computes the squared weighted sum on the internal nodes at height 1. This regularizer can be viewed as a hierarchical group Lasso where at each height, the weights corresponding to nodes sharing the same parent node will be combined together via some norm. In general, for any positive integer \( m \), when all the exponents of different \( \sigma_{j,k} \)'s have the same value (i.e., \( 1/m \)), we can always find the explicit form of the second term in the objective function of problem (1) in a similar way to the proof of Theorem 3.

Optimization

Since problem (1) is convex, we use the FISTA method (Beck and Teboulle 2009) to solve it. We use a variable \( \phi \) to denote the concatenation of \( w \) and \( \sigma \). We define

\[
f(\phi) = \sum_{i=1}^{n} l(x_i, y_i, w_i) + \frac{\lambda_1}{2} \sum_{i=1}^{d} \lambda_i w_i^2 + \frac{\lambda_2}{2} \sum_{i=1}^{d} \sigma_i^{(j)} w_{j,i}^2 \]

and \( g(\phi) = \frac{\lambda_2}{2} ||w||_2^2 \). We define the set of constraints on \( \phi \) as

\[S_\phi = \{ \phi | \sum_{j=1}^{m} d_j^{(i)} \sigma_j^{(i)} = 1 \forall i \in [m], \sigma_j^{(i)} \geq 0 \forall i, j \} \].

In the FISTA algorithm, it does not minimize the original composite objective function \( F(\phi) = f(\phi) + g(\phi) \), but instead solves a surrogate function:

\[q_r(\hat{\phi}) = \min_{\phi \in S_\phi} Q_r(\phi, \hat{\phi}), \]

where \( Q_r(\phi, \hat{\phi}) = g(\phi) + f(\hat{\phi} + (\phi - \hat{\phi})^T \nabla f(\hat{\phi}) + \frac{r}{2} ||\phi - \hat{\phi}||_2^2 \) and \( \nabla f(\hat{\phi}) \) denotes the derivative of \( f(\phi) \) with respect to \( \phi \) at \( \phi = \hat{\phi} \). Hence, in the FISTA algorithm, we just need to minimize \( Q_r(\phi, \hat{\phi}) \) with respect to \( \phi \in S_\phi \).

Specifically, we need to solve the following problem:

\[\min_{w, \sigma} \frac{\lambda_1}{2} ||w||_2^2 + \frac{r}{2} ||w - \tilde{w}||_2^2 + \frac{r}{2} ||\sigma - \tilde{\sigma}||_2^2 \]

s.t. \[\sum_{j=1}^{m} d_j^{(i)} \sigma_j^{(i)} = 1 \forall i \in [m], \sigma_j^{(i)} \geq 0 \forall i, j \]

(3)

where \( r \) is a step size which can be determined by the FISTA algorithm, \( \tilde{w} = \hat{w} - \frac{1}{r} \nabla w f(\hat{w}) \), and \( \tilde{\sigma} = \hat{\sigma} - \frac{1}{r} \nabla \sigma f(\hat{\sigma}) \). It is easy to see that the solution for \( w \) in problem (3) is \( w = \frac{\tilde{w}}{\lambda_2 + r||\tilde{w}||_2^2} \). For \( \sigma \) in problem (3), we can find that the corresponding problem can be decomposed into \( m - 1 \) subproblems with each one solving a problem with respect to \( \{ \sigma_j^{(i)} \}_{j=1}^{m} \) and these subproblems have the same formulation as

\[\min_{\rho} ||\rho - \hat{\rho}||_2^2 \text{ s.t. } \rho \geq 0, a^T \rho = 1, \]

(4)

where \( \hat{\rho} \) denotes a zero vector with appropriate size, \( \geq \) denotes the elementwise inequalities between two vectors, and \( a \) is a constant vector with all entries positive. Problem (4) is a quadratic programming (QP) problem and we can use some off-the-shelf QP solver to solve it. To accelerate the optimization of problem (4), we propose a more efficient solution for problem (4) by solving its dual form and the detailed procedure is put in the supplementary material.

A Variant to Learn Exponents

In the DHS method, we need to manually set the exponents \( \{\theta_{p,i} \} \). By default, we usually assume that all \( \theta_{p,i} \)'s are equal to \( \frac{1}{m} \), which satisfies the requirement to guarantee the convexity of problem (1). However,
this setting may be suboptimal. In this section, we propose a DHS, method, a variant of the DHS method, to learn the exponents from data directly.

The objective function of the DHS method is formulated as

\[
\min_{w, a, \theta} \frac{1}{n} \sum_{i=1}^{n} l(x_i, y_i, w) + \frac{\lambda_1}{2} \sum_{i=1}^{d} \sum_{\ell=1}^{\ell_i} w_i^2 \prod_{j=1}^{m} \left( \sigma_{p_{\ell-1}^{(j)}, p_{\ell}^{(j)}} \right)^{\theta_{p_{\ell-1}^{(j)}, p_{\ell}^{(j)}}} \\
+ \frac{\lambda_2}{2} \|w\|_2^2
\]

s.t. \sum_{j=1}^{m} d_j^{(i)} \sigma_{p_{\ell-1}^{(j)}, p_{\ell}^{(j)}} = 1 \forall i \in [m], \sigma_{p_{\ell-1}^{(j)}, p_{\ell}^{(j)}} \geq 0 \forall i, j

\sum_{j=1}^{m} \theta_{p_{\ell-1}^{(j)}, p_{\ell}^{(j)}} = 1 \forall i \in [d], \theta_{p_{\ell-1}^{(j)}, p_{\ell}^{(j)}} \geq 0 \forall i, j.

Different from problem (1) where all the \(\theta_{p_{\ell-1}^{(j)}, p_{\ell}^{(j)}}\)'s are constants, all the \(\theta_{p_{\ell-1}^{(j)}, p_{\ell}^{(j)}}\)'s in problem (5) are variables to be optimized. The equality and inequality constraints with respect to \(\theta\) in problem (5) satisfy the requirements for the constant exponents in problem (1) and they form an \((m-1)\)-dimensional simplex for each feature. Different from problem (1) which is convex, problem (5) can be proved to be non-convex with respect to all variables and hence we use the GIST algorithm (Gong et al. 2013) to solve it. Due to page limit, we put the detailed optimization procedure in the supplementary material.

**Learning Deep Hierarchical Structure**

In some applications, the hierarchical structure is not available. In this section, we propose the LDHS method to learn both the hierarchical structure and the model parameters from data directly.

We assume that the height of the hierarchical structure to be learned is given as \(m\). Here we use slightly different notations to define the hierarchical structure. The weights of edges on the path from the root node to the \(i\)th leaf node corresponding to the \(i\)th feature are denoted by \(\omega_{i}^{(1)}, \ldots, \omega_{i}^{(m)}\), where \(\omega_{i}^{(j)}\) denotes the weight of an edge connecting the height \(j-1\) and \(j\) on the path from the root node to the \(i\)th leaf node. At the beginning, we assume that the paths from the root node to any two different leaf nodes do not share any edge. When \(\omega_{i}^{(j)}\) equals \(\omega_{k}^{(j)}\) for some \(i\) and \(j\) and \(k\), we can view it as a sign for that the two paths from the root node to the \(j\)th and \(k\)th leaf nodes become fused at height \(i\) and then in order to keep the whole structure as a hierarchical structure, it should be required that the subpaths on the two paths above height \(i\) are always fused, implying that \(\omega_{j}^{(i')}\) will always equal \(\omega_{k}^{(i')}\) when \(i' \leq i\). So an algorithm that can learn a valid hierarchical structure should satisfy the following two requirements:

1. It should have the ability to enforce \(\omega_{j}^{(i)}\) to be equal to \(\omega_{k}^{(i)}\) for some \(i\) and \(j\) and \(k\);
2. It should guarantee that when \(\omega_{j}^{(i)}\) equals \(\omega_{k}^{(i)}\) for all \(i' < i\), \(\omega_{j}^{(i')}\) will equal \(\omega_{k}^{(i')}\).

Here we present the objective function of the LDHS method which can satisfy those two requirements:

\[
\min_{w, \omega} \frac{1}{n} \sum_{i=1}^{n} l(x_i, y_i, w) + \frac{\lambda_1}{2} \prod_{j=1}^{m} \left( \omega_{j}^{(i)} \right)^{\frac{1}{2}}
+ \frac{\lambda_2}{2} \|w\|_2^2 + \sum_{i=1}^{m} \eta_i \sum_{j<k} |\omega_i^{(j)} - \omega_j^{(j)}|
\]

s.t. \(\sum_{j=1}^{d} \omega_j^{(i)} = 1 \forall i \in [m], \omega_j^{(i)} \geq 0 \forall i, j\)

\(|\omega_i^{(1)} - \omega_j^{(1)}| \leq \ldots \leq |\omega_i^{(m)} - \omega_j^{(m)}| \forall i, k\).

where \(\omega\) is a vector containing all \(\omega_i^{(j)}\)'s. The last term in the objective function of problem (6), a layer-wise generalized fused-Lasso regularizer (Tibshirani et al. 2005; Hocking et al. 2011), can make \(\omega_i^{(j)}\) equal to \(\omega_j^{(j)}\) for some \(i\), \(j\) and \(k\), and so this regularizer can satisfy the first requirement. The sequential inequality constraint in problem (6) can satisfy the second requirement since when \(\omega_i^{(j)}\) equals \(\omega_k^{(j)}\), we can get \(|\omega_i^{(j)} - \omega_j^{(j)}| \leq 0\) for any \(1 \leq i' < i\), implying that \(\omega_i^{(j)} = \omega_k^{(j)}\). Therefore, problem (6), which satisfies the two requirements, can learn a hierarchical structure. The exponents of all the \(\omega_i^{(j)}\)'s are set to be \(\frac{1}{m}\). We can also set them to be other values or even learn them as the DHS method did and this will be left as the future work. The regularization parameter \(\eta\) controls the level of fusion between \(\{\omega_i^{(j)}\}_{j=1}^{d}\) at height \(i\). A larger \(\eta\) will lead to more identical values in \(\{\omega_i^{(j)}\}_{j=1}^{d}\). Therefore, it is intuitive to define an increasing order for \(\eta_i\)'s from height \(m\) to height \(0\) to help construct the hierarchical structure. In practice, we set \(\eta_i = \eta \nu^i\) for \(i \geq 2\) with some constant \(\nu > 1\). When the hierarchical structure is available or equivalently \(\omega\) is given, problem (6) becomes problem (1) and hence the LDHS method is a generalization of the DHS method to learn the hierarchical structure.

Even though the objective function of problem (6) is convex based on Theorem 1, the whole problem is non-convex due to the sequential inequality constraint and we also use the GIST method to solve it. We still use the variable \(\phi\) to denote the concatenation of \(w\) and \(\omega\). We define a set of constraints on \(\phi\) as \(S_{\phi} = \{\phi| \sum_{j=1}^{m} \omega_j^{(i)} = 1 \forall i \in [m], \omega_j^{(i)} \geq 0 \forall i, j, |\omega_i^{(1)} - \omega_j^{(1)}| \leq \ldots \leq |\omega_i^{(m)} - \omega_j^{(m)}| \forall j, k\}\). We define

\[
g(\phi) = \left\{ \begin{array}{ll}
\frac{\lambda_2}{2} \|w\|_2^2 + \sum_{i=1}^{m} \eta_i \sum_{j<k} |\omega_i^{(j)} - \omega_j^{(j)}| & \text{if } \phi \in S_{\phi} \\
\infty & \text{otherwise}
\end{array} \right.
\]

\[
f(\phi) = \frac{1}{n} \sum_{i=1}^{n} l(x_i, y_i, w) + \frac{\lambda_1}{2} \prod_{j=1}^{m} \left( \omega_{j}^{(i)} \right)^{\frac{1}{2}}.
\]

By omitting some constant terms, the proximal problem to
be solved in the GIST algorithm can be formulated as
\[
\min_{\omega, \mu} \frac{r}{2} \| \omega - \hat{\omega} \|^2_2 + \frac{\rho}{2} \| \omega - \hat{\omega} \|^2_2 + \frac{\lambda}{2} \| \omega \|^2_2 + \sum_{i=1}^m \eta_i \sum_{j<k} |\omega_{k}^{(i)} - \omega_{j}^{(i)}| \\
\text{s.t.} \quad \sum_{j=1}^d \omega_{j}^{(i)} = 1 \forall i \in [m], \quad \omega_{j}^{(i)} \geq 0 \forall i, j
\]
where \( \hat{\omega} \) and \( \omega \) are previous estimations for \( \omega \) and \( \hat{\omega} \) respectively. This problem can be reformulated as
\[
\min_{\omega, \mu} \frac{r}{2} \| \omega - \hat{\omega} \|^2_2 + \sum_{i=1}^m \eta_i \| \mu \|_1 \\
\text{s.t.} \quad \sum_{j=1}^d \omega_{j}^{(i)} = 1 \forall i \in [m], \quad \omega_{j}^{(i)} \geq 0 \forall i, j
\]
where \( \| \cdot \|_1 \) denotes the \( \ell_1 \) norm of a vector, \( \omega^{(i)} = (\omega_{1}^{(i)}, \ldots, \omega_{d}^{(i)})^T \), \( \mu \) is a vector of all ones with an appropriate size, \( \omega^{(i)} \) for a vector \( \omega \), and \( B \) is a \( d \times d \) matrix with each of its rows containing only two non-zero entries 1 and \(-1\) at corresponding locations. Problem (8) seems complicated and we reformulate it as
\[
\min_{\omega, \mu, \tau} \frac{r}{2} \| \omega - \hat{\omega} \|^2_2 + \sum_{i=1}^m \eta_i \| \tau^{(i)} \|_1 \\
\text{s.t.} \quad \sum_{j=1}^d \omega_{j}^{(i)} = 1 \forall i \in [m], \quad \omega_{j}^{(i)} \geq 0 \forall i, j
\]
where \( \tau^{(i)} \) are nonnegative. We first use the algorithm with fixed \( \mu \), then we need to solve the following subproblem with respect to \( \omega \) as:
\[
\min_{\omega} \frac{m}{2} \| \omega^{(i)} - \hat{\omega} \|^2_2 \text{ s.t. } 1^T \omega^{(i)} = 1, \quad \omega^{(i)} \geq 0 \forall i
\]
where \( \hat{\omega} = \frac{1}{r \rho} (r \hat{\omega} + p_i + \rho \mu^{(i)}) \). This problem can be decomposed into \( m \) subproblems where the \( i \)th subproblem with respect to \( \omega^{(i)} \) has the same formulation as problem (4), which has an efficient solution.

In the ADMM algorithm, problem (10) can be solved alternatively with respect to \( \omega, \mu \) and \( \tau \).

With fixed \( \mu \) and \( \tau \), we need to solve the following subproblem with respect to \( \omega \) as:
\[
\min_{\omega} \frac{m}{2} \| \omega^{(i)} - \hat{\omega} \|^2_2 \text{ s.t. } 1^T \omega^{(i)} = 1, \quad \omega^{(i)} \geq 0 \forall i
\]
where \( \hat{\omega} = \frac{1}{r \rho} (r \hat{\omega} + p_i + \rho \mu^{(i)}) \). This problem can be decomposed into \( m \) subproblems where the \( i \)th subproblem with respect to \( \omega^{(i)} \) has the same formulation as problem (4), which has an efficient solution.

With fixed \( \omega \) and \( \tau \), the subproblem with respect to \( \mu \) is a QP problem. By setting the derivative of \( \mathcal{L}(\omega, \mu, \tau) \) with respect to \( \mu^{(i)} \) to zero, we can obtain the analytical solution for \( \mu^{(i)} \)
\[
\mu^{(i)} = (I + B^T B)^{-1} \left( B^T \tau^{(i)} + \frac{1}{\rho} B^T q_i + \omega^{(i)} - \frac{1}{\rho} p_i \right).
\]
Note that \((I + B^T B)^{-1}\) is a constant matrix. So it can be computed and stored before solving the whole problem, leading to a more efficient implementation.

With fixed \( \omega \) and \( \mu \), the subproblem with respect to \( \tau \) can be decomposed into \( \frac{d(d-1)}{2} \) subproblems with the \( j \)th one formulated as
\[
\min_{\tau^{(j)}} \sum_{i=1}^m \left( \frac{\rho}{2} \left( \nu^{(j)} - \nu^{(j)}_{i} \right)^2 + \eta_i |\tau^{(j)}_{i}| \right) \text{ s.t. } |\tau^{(j)}_{i}| \leq \cdots \leq |\tau^{(j)}_{m}|,
\]
where \( \nu^{(j)} = \omega^{(j)} \nu^{(j)} - \frac{1}{\rho} q_i \), and \( \nu^{(j)}_{i} \) are the \( j \)th entries of \( \nu^{(j)} \) and \( \nu^{(j)} \), respectively. The optimal \( \tau^{(j)}_{i} \) must have the same sign as \( \nu^{(j)}_{i} \) since otherwise we can flip the sign of \( \nu^{(j)}_{i} \) to achieve a lower value for the first term in the summand of the objective function in problem (11) while keeping other terms unchanged and also satisfying the constraints, which leads to a lower objective value. Then by defining new variables \{\tilde{\tau}^{(j)}_{i}\} as \( \tilde{\tau}^{(j)}_{i} = \text{sgn}(\nu^{(j)}_{i})\tau^{(j)}_{i} \) where \( \text{sgn}(\cdot) \) gives the sign of a scalar, \( \tilde{\tau}^{(j)}_{i} \) is always nonnegative and based on problem (11), the problem with respect to \( \{\tilde{\tau}^{(j)}_{i}\} \) can be formulated as
\[
\min_{\{\tilde{\tau}^{(j)}_{i}\}} \sum_{i=1}^m \left( \tilde{\tau}^{(j)}_{i} - \nu^{(j)}_{i} \right)^2 \text{ s.t. } 0 \leq \tilde{\tau}^{(j)}_{i} \leq \cdots \leq \tilde{\tau}^{(j)}_{m},
\]
where \( \tilde{\nu}^{(j)}_{i} = |\nu^{(j)}_{i}| - \frac{1}{\rho} \eta_i \). This problem is similar to problem (17) in (Han and Zhang 2015b) except for the requirement that all \( \tilde{\tau}^{(j)}_{i} \)’s are nonnegative. We first use the algorithm with linear complexity in Section 3.2 of (Han and Zhang 2015b) to solve this problem and then make negative ones in \( \{\tilde{\tau}^{(j)}_{i}\} \) become zero to obtain the final solution.

**Experiments**

In this section, we conduct empirical evaluations on both synthetic and real-world problems.
We compare the proposed models (i.e., DHS, DHS, and LDHS) with state-of-the-art structured feature learning methods, including the Lasso, Group Lasso (GLasso) (Yuan and Lin 2006), the CAP family with $\ell_4$ and $\ell_\infty$ penalties denoted by CAP$_{\ell_4}$ and CAP$_{\ell_\infty}$ (Zhao, Rocha, and Yu 2009), and the HP method (Szafranski, Grandvalet, and Morizet-Mahoudeaux 2007). Among these baseline methods, the Lasso method does not take any group or hierarchical structure into consideration and the GLasso and HP models require a hierarchical structure of height 2, while the CAP, DHS and DHS methods need a hierarchical structure with an arbitrary height. In order to provide a fair comparison, in all the experiments we first generate a hierarchical structure on the features and then apply it to the GLasso, CAP, HP, DHS and DHS methods, where the group structure required by the GLasso and HP methods is obtained from the two bottom-most layers of the hierarchical structure. For the LDHS method, we set its height to be that of the given hierarchical structure.

**Experiments on Synthetic Data**

We first experiment on synthetic data. We generate three synthetic data by varying the height of the hierarchical structure as $m = 3$, 4 and 5, respectively. For simplicity, we use full binary trees and the numbers of features $d$ under the three settings are equal to $2^3$, $2^4$ and $2^5$, respectively. The ground truth of the feature weights $w^*$ when $m = 3$ is shown in Fig. 2(a) and those for other cases are put in the supplementary material. Then, we generate data instances from a normal distribution $\mathcal{N}(0, I_d)$, where $I_d$ is a $d \times d$ identity matrix. We assume a linear model between data instances and labels, i.e., $y = Xw^* + \epsilon$ with each entry $\epsilon_i$ in $\epsilon$ ($i = 1, \cdots, n$) following $\mathcal{N}(0, \xi_i)$. In all the settings, we generate $n = 100$ training samples and set $\xi = 2$.

Table 1: The MSE’s in terms of ‘mean±std’ on synthetic data.

<table>
<thead>
<tr>
<th>m</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lasso</td>
<td>0.2913±0.0399</td>
<td>0.8669±0.0499</td>
<td>1.8660±0.0549</td>
</tr>
<tr>
<td>GLasso</td>
<td>0.2945±0.0372</td>
<td>0.8957±0.0402</td>
<td>1.8877±0.0593</td>
</tr>
<tr>
<td>CAP$_{\ell_4}$</td>
<td>0.3069±0.0349</td>
<td>0.8957±0.0402</td>
<td>1.8885±0.0495</td>
</tr>
<tr>
<td>CAP$<em>{\ell</em>\infty}$</td>
<td>0.3069±0.0349</td>
<td>0.8957±0.0402</td>
<td>1.8885±0.0497</td>
</tr>
<tr>
<td>HP</td>
<td>0.2795±0.0439</td>
<td>0.8403±0.0314</td>
<td>1.5519±0.0550</td>
</tr>
<tr>
<td>DWS</td>
<td>0.2743±0.0337</td>
<td>0.8275±0.0336</td>
<td>1.4682±0.0433</td>
</tr>
<tr>
<td>DHS,</td>
<td>0.2748±0.0305</td>
<td>0.8337±0.0324</td>
<td>1.5596±0.0474</td>
</tr>
<tr>
<td>LDHS</td>
<td><strong>0.2370±0.0303</strong></td>
<td><strong>0.8283±0.0354</strong></td>
<td><strong>1.3894±0.0460</strong></td>
</tr>
</tbody>
</table>

We randomly generate 100 samples for testing and use another 100 random samples for validation to choose regularization parameters of all the methods. All of the regularization parameters in different models are chosen from a set $\{10^{-5}, \cdot \cdot \cdot, 1\}$, except $\eta_i$’s in the proposed LDHS method. As discussed before, we set $\eta_{i+1} = \eta_i/v$ for $i < m$, where we choose $\eta_1$ and $v$ from $\{10^{-5}, \cdot \cdot \cdot, 1\}$ and $\{1, 1.2, 10\}$, respectively. We use the Mean Square Error (MSE) to evaluate different methods, where the MSE is defined as $\frac{1}{n}(\hat{w} - w^*)^\top X^\top X(\hat{w} - w^*)$ for an estimation $\hat{w}$.

All the settings are repeated for 10 times to obtain the average results, which are reported in Table 1. From the results, we observe that the proposed models with deeper hierarchical structure, i.e., the DHS, DHS, and LDHS models, generally show better performance than Lasso, GLasso, CAP$_{\ell_4}$, CAP$_{\ell_\infty}$ and HP. This demonstrates that whenever features can be organized into a deeper hierarchical structure, using the correct (deeper) hierarchical structure will allow a lower prediction error compared to models with shallow structure, e.g., groups or a hierarchical structure of height 2. The GLasso, CAP$_{\ell_4}$ and CAP$_{\ell_\infty}$ methods show comparable results since all of them belong to the CAP family and may acquire similar information from the feature groups. The DHS and DHS methods show comparable performance, while the LDHS method, which learns the hierarchical structure automatically, achieves the lowest MSE in two of all the three settings.

Moreover, in Fig. 2, we show the estimated parameter $\sigma$ in the DHS method and $\omega$ in the LDHS method when $m = 3$. The estimated $\sigma$ in DHS is similar to that in DHS, hence we omit its result here. In Fig. 2(b), the estimated $\sigma$
well matches the hierarchical structure. For example, the estimated \( \sigma \) along the path from the root to the last four features are generally small. According to the formulation of the DHS model, these \( \sigma \)'s will give heavier penalizations on the corresponding feature weights and as a consequence, we can obtain small feature weights, which match the true values of \( w^* \). Similarly, for the LDHS model, the parameter \( \omega \) is an \( m \times d \) matrix. We show the estimated \( \omega \) in Fig. 2(e), from which we can generally observe a hierarchical structure by comparing their values, and this hierarchical structure well matches the ground truth. This result demonstrates that the LDHS method is able to recover the hierarchical structure.

**Experiments on Real-World Datasets**

In this section, we experiment on three real-world datasets including the traffic volume data (Han and Zhang 2015b), the breast cancer data (Jacob, Obozinski, and Vert 2009) and the Covtype data. In these datasets, the hierarchical structure over the features is not available. By following (Kim and Xing 2010), we use a simple hierarchical \( k \)-means clustering method to generate the hierarchical structure on the features. Specifically, we perform \( k \)-means clustering to split the features into two groups, and for each group we recursively perform \( k \)-means clustering to obtain four sub-groups. Therefore, the resultant hierarchical structure has a height of \( m = 3 \) and hence in the LDHS method, the height of the learned hierarchical structure is also set to 3. The following experiments show that such a simple hierarchical structure is sufficient to obtain good performance for the proposed methods.

First we experiment on the traffic volume data. This dataset collects the traffic volumes from 136 entries (treated as features) and the traffic volumes through some exits (treated as response) in a highway traffic network. We choose 5 exits with the highest volumes in the network to form 5 learning tasks, each of which is to use the volumes from entries to predict the volume through a specific exit. There are totally 384 data samples. These tasks are regression problems and we use the square loss for all the methods. We randomly select 80% and 20% samples for training and testing, respectively. The regularization parameters are chosen from the same candidate set as used in the synthetic setting via 5-fold cross-validation. We use the MSE, i.e., \( \frac{1}{n} \| y - \hat{y} \|_2^2 \), to measure the performance of all the methods. The results on the 5 tasks are given in Table 2. From the results, we observe that in 4 out of the 5 tasks, the proposed DHS, DHS\(_c\), and LDHS methods outperform the baseline methods and the LDHS method achieves the lowest MSE. This demonstrates again that a deeper tree structure can provide a better feature structure in this data and our proposed methods can take advantage of this structure.

<table>
<thead>
<tr>
<th>Task</th>
<th>Task 2</th>
<th>Task 3</th>
<th>Task 4</th>
<th>Task 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lasso</td>
<td>0.1066±0.0179</td>
<td>0.1366±0.0279</td>
<td>0.1155±0.0231</td>
<td>0.1215±0.0132</td>
</tr>
<tr>
<td>GLasso</td>
<td>0.1042±0.0190</td>
<td>0.1185±0.0201</td>
<td>0.1030±0.0228</td>
<td>0.0934±0.0122</td>
</tr>
<tr>
<td>CAP(_e)_1</td>
<td>0.0939±0.0173</td>
<td>0.1122±0.0164</td>
<td>0.0994±0.0180</td>
<td><strong>0.0864±0.0109</strong></td>
</tr>
<tr>
<td>CAP(<em>e)</em>∞</td>
<td>0.0977±0.0172</td>
<td>0.1179±0.0222</td>
<td>0.1029±0.0193</td>
<td>0.0925±0.0136</td>
</tr>
<tr>
<td>HP</td>
<td>0.0869±0.0189</td>
<td>0.1087±0.0270</td>
<td>0.0963±0.0196</td>
<td>0.0915±0.0133</td>
</tr>
<tr>
<td>DHS</td>
<td>0.0839±0.0187</td>
<td>0.1049±0.0262</td>
<td>0.0929±0.0187</td>
<td>0.0884±0.0132</td>
</tr>
<tr>
<td>DHS(_c)</td>
<td>0.0843±0.0189</td>
<td>0.1051±0.0262</td>
<td>0.0935±0.0189</td>
<td>0.0890±0.0128</td>
</tr>
<tr>
<td>LDHS</td>
<td><strong>0.0838±0.0186</strong></td>
<td><strong>0.1048±0.0262</strong></td>
<td><strong>0.0928±0.0186</strong></td>
<td>0.0882±0.0126</td>
</tr>
</tbody>
</table>

Next, we conduct experiments on the breast cancer and Covtype datasets which have been studied in (Jacob, Obozinski, and Vert 2009; Han and Zhang 2015a). The breast cancer dataset contains genes in 295 tumors, among which 78 of them are metastatic while 217 are non-metastatic. Hence, the tasks here are binary classification problems. We use the square loss for all methods. It has been shown that in this dataset, some latent hierarchical structure exists. Similar to (Jacob, Obozinski, and Vert 2009; Han and Zhang 2015a), we select 300 most correlated genes as features and the traffic volumes through some exits to the DHS model, these \( \sigma \)'s will give heavier penalizations on the corresponding feature weights and as a consequence, we can obtain small feature weights, which match the true values of \( w^* \). Similarly, for the LDHS model, the parameter \( \omega \) is an \( m \times d \) matrix. We show the estimated \( \omega \) in Fig. 2(c), from which we can generally observe a hierarchical structure by comparing their values, and this hierarchical structure well matches the ground truth. This result demonstrates that the LDHS method is able to recover the hierarchical structure.

**Experiments on Real-World Datasets**

In this section, we experiment on three real-world datasets including the traffic volume data (Han and Zhang 2015b), the breast cancer data (Jacob, Obozinski, and Vert 2009) and the Covtype data. In these datasets, the hierarchical structure over the features is not available. By following (Kim and Xing 2010), we use a simple hierarchical \( k \)-means clustering method to generate the hierarchical structure on the features. Specifically, we perform \( k \)-means clustering to split the features into two groups, and for each group we recursively perform \( k \)-means clustering to obtain four sub-groups. Therefore, the resultant hierarchical structure has a height of \( m = 3 \) and hence in the LDHS method, the height of the learned hierarchical structure is also set to 3. The following experiments show that such a simple hierarchical structure is sufficient to obtain good performance for the proposed methods.

First we experiment on the traffic volume data. This dataset collects the traffic volumes from 136 entries (treated as features) and the traffic volumes through some exits (treated as response) in a highway traffic network. We choose 5 exits with the highest volumes in the network to form 5 learning tasks, each of which is to use the volumes from entries to predict the volume through a specific exit. There are totally 384 data samples. These tasks are regression problems and we use the square loss for all the methods. We randomly select 80% and 20% samples for training and testing, respectively. The regularization parameters are chosen from the same candidate set as used in the synthetic setting via 5-fold cross-validation. We use the MSE, i.e., \( \frac{1}{n} \| y - \hat{y} \|_2^2 \), to measure the performance of all the methods. The results on the 5 tasks are given in Table 2. From the results, we observe that in 4 out of the 5 tasks, the proposed DHS, DHS\(_c\), and LDHS methods outperform the baseline methods and the LDHS method achieves the lowest MSE. This demonstrates again that a deeper tree structure can provide a better feature structure in this data and our proposed methods can take advantage of this structure.

**Table 3: Results on the Breast Cancer (top) and Covtype (bottom) datasets.**

<table>
<thead>
<tr>
<th>Method</th>
<th>ACC (%)</th>
<th>SEN (%)</th>
<th>SPE (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lasso</td>
<td>76.81±3.54</td>
<td>91.48±4.25</td>
<td>71.99±6.22</td>
</tr>
<tr>
<td>GLasso</td>
<td>80.69±3.08</td>
<td>94.42±3.67</td>
<td>76.24±4.77</td>
</tr>
<tr>
<td>CAP(_e)_1</td>
<td>81.53±3.08</td>
<td><strong>95.47±3.25</strong></td>
<td>76.96±4.97</td>
</tr>
<tr>
<td>CAP(<em>e)</em>∞</td>
<td>80.97±3.33</td>
<td><strong>95.47±3.25</strong></td>
<td>76.23±5.04</td>
</tr>
<tr>
<td>HP</td>
<td>79.68±3.94</td>
<td>92.22±3.44</td>
<td>75.48±6.46</td>
</tr>
<tr>
<td>DHS</td>
<td>82.54±3.78</td>
<td>91.59±5.30</td>
<td>79.56±4.96</td>
</tr>
<tr>
<td>DHS(_c)</td>
<td><strong>83.49±3.48</strong></td>
<td>92.84±5.53</td>
<td><strong>80.41±4.32</strong></td>
</tr>
<tr>
<td>LDHS</td>
<td>82.38±3.71</td>
<td>91.59±7.16</td>
<td>79.36±3.75</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Method</th>
<th>ACC (%)</th>
<th>SEN (%)</th>
<th>SPE (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lasso</td>
<td>74.61±0.28</td>
<td>67.03±1.24</td>
<td>72.58±0.73</td>
</tr>
<tr>
<td>GLasso</td>
<td>74.97±0.28</td>
<td>70.91±1.06</td>
<td>79.24±0.56</td>
</tr>
<tr>
<td>CAP(_e)_1</td>
<td>74.81±0.11</td>
<td>70.33±0.19</td>
<td>79.51±0.13</td>
</tr>
<tr>
<td>CAP(<em>e)</em>∞</td>
<td>74.80±0.10</td>
<td>70.30±0.18</td>
<td>79.52±0.14</td>
</tr>
<tr>
<td>HP</td>
<td>75.39±0.08</td>
<td><strong>74.31±0.18</strong></td>
<td>76.54±0.10</td>
</tr>
<tr>
<td>DHS</td>
<td>75.58±0.07</td>
<td>73.86±0.14</td>
<td>77.39±0.09</td>
</tr>
<tr>
<td>DHS(_c)</td>
<td>75.37±0.07</td>
<td>71.12±0.32</td>
<td><strong>79.84±0.30</strong></td>
</tr>
<tr>
<td>LDHS</td>
<td><strong>75.60±0.08</strong></td>
<td>74.04±0.15</td>
<td>77.24±0.10</td>
</tr>
</tbody>
</table>
581, 012 examples and the feature dimensionality is \( d = 54 \). We evaluate all the methods on these two datasets based on three measures, i.e., the accuracy (ACC), sensitivity (SEN), and specificity (SPE), similar to (Yang et al. 2012; Han and Zhang 2015a). By following (Yang et al. 2012; Han and Zhang 2015a), 50%, 30% and 20% of the data are randomly chosen for training, validation, and testing, respectively. The regularization parameters are selected from the same candidate sets as described in the previous experiments. The top table in Table 3 shows the average results over 10 repetitions on the breast cancer data. According to the results, hierarchical methods with deep tree structure generally show more accurate predictions. The DHS method achieves the best performance in this case. The CAP\(_{e}\) and CAP\(_{\ell_\infty}\) have better sensitivities than the proposed methods, implying that they can recover more true positive examples, while their SPE is lower. According to results reported in the bottom table of Table 3, we can see that the HP, DHS, DHS\(_e\), and LDHS generally outperform the Lasso and GLasso methods on the Covtype data. Again, the LDHS method, which learns the tree structure, obtains the best accuracy.

Conclusions

In this paper, we study the problem of learning (from) deep hierarchical structure step by step. In the first step, we propose the convex DHS method to learn from hierarchical structure with an arbitrary height. Secondly, we propose the DHS method, a variant of the DHS method, to learn the exponents from data. Finally, we propose the LDHS method to learn the hierarchical structure since it may be unavailable in some applications.

In our future work, we will learn the exponents of the LDHS method in a way similar to the DHS\(_e\) method.

Acknowledgment

This research has been supported by NSFC (61473087 and 61673202).

References


