A Recursive Regularization Based Feature Selection Framework for Hierarchical Classification

Hong Zhao, Qinghua Hu, *Senior Member, IEEE*, Pengfei Zhu, Yu Wang, and Ping Wang

Abstract—The sizes of datasets in terms of the number of samples, features, and classes have dramatically increased in recent years. In particular, there usually exists a hierarchical structure among class labels as hundreds of classes exist in a classification task. We call these tasks hierarchical classification, and hierarchical structures are helpful for dividing a very large task into a collection of relatively small subtasks. Various algorithms have been developed to select informative features for flat classification. However, these algorithms ignore the semantic hyponymy in the directory of hierarchical classes, and select a uniform subset of the features for all classes. In this paper, we propose a new feature selection framework with recursive regularization for hierarchical classification. This framework takes the hierarchical information of the class structure into account. In contrast to flat feature selection, we select different feature subsets for each node in a hierarchical tree structure with recursive regularization. The proposed framework uses parent-child, sibling, and family relationships for hierarchical regularization. By imposing $\ell_{2,1}$-norm regularization to different parts of the hierarchical classes, we can learn a sparse matrix for the feature ranking at each node. Extensive experiments on public datasets demonstrate the effectiveness and efficiency of the proposed algorithms.

Index Terms—Feature selection, hierarchical classification, recursive regularization, semantic hyponymy

1 Introduction

In the era of big data, the scale of classification tasks increases with respect to the size of samples and features, as well as the number of candidate class labels [1]. For example, tens of millions of image samples from tens of thousands of class labels are collected in ImageNet [2], where each sample is described with thousands of attributes. It is remarkable that the number of classes in various tasks exceeds 100 with the rapid surge of data. Some large-scale classification tasks have hundreds, thousands, or even tens of thousands of class labels [3]. As the number of labels increases, there typically exists a semantic structure among the labels, which leads to a structural learning task. This structure can typically be represented by a hierarchical tree, and we call such tasks hierarchical classification [4].

The hierarchical class structure is obviously important auxiliary information for classification learning. This information helps to divide a large and complex task into a set of relatively small and easy subtasks. Growing attention has been paid to this topic in recent years [5], [6]. A collection of algorithms have been developed to exploit hierarchies in training classification models, including text categorization [7], visual recognition [8], lung disease classification [9], gene function prediction [10], and plant species identification [11].

Feature selection has gotten much attention in classification learning [12], [13]. Redundant and irrelevant features are gathered during data collection because users typically do not know which features are useful for current tasks. It is well accepted that superfluous features lead to classification performance deterioration because of the curse of dimensionality. Selecting a subset of features from the data can provide a compact representation of a classification task [14], [15]. A great number of algorithms have been proposed in recent years for conventional classification. These algorithms select a common subset of features for discriminating all objects. These algorithms need to select many features if there are many classes to be discriminated. In fact, some of the selected features are only useful for recognizing one or several classes. Thus, these algorithms are not applicable to large-scale classification tasks.

To combat the challenge of feature selection for large-scale classification, hierarchical structures can also be considered. It is not feasible to assume that all of the classes share the same set of relevant features for hierarchical feature selection [16]. The useful features for distinguishing some classes may be useless for others [17]. Thus we should select different features for different subtasks to construct an effective feature subset which leads to a compact and powerful classification

- *H. Zhao is with the College of Intelligence and Computing, Tianjin University, with Tianjin Key Lab of Machine Learning, Tianjin 300354, China, and with the Fujian Key Laboratory of Granular Computing and Application, Minnan Normal University, Zhangzhou 363000, China. E-mail: hongzhaocn@tju.edu.cn.*
- *Q. H. Hu, P. F. Zhu, and Y. Wang are with the College of Intelligence and Computing, Tianjin University, and with Tianjin Key Lab of Machine Learning, Tianjin 300354, China. E-mail: {huqinghua, zhupengfei, armstrong_wangyu}@tju.edu.cn.*
- *P. Wang is with the School of Mathematics, School of Computer Software, Tianjin University, Tianjin 300354, China. E-mail: wang_ping@tju.edu.cn.*

Manuscript received 21 Apr. 2018; revised 14 Nov. 2019; accepted 12 Dec. 2019. Date of publication 23 Dec. 2019; date of current version 3 June 2021. (Corresponding author: Hong Zhao.)
Recommended for acceptance by S. Matwin.
Digital Object Identifier no. 10.1109/TKDE.2019.2960251

1041-4347 © 2019 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

model. We believe a machine learning system can and should leverage such information as well for better performance [16].

However, little work has been conducted to deal with hierarchical feature selection. For such tasks, we divide a large-scale classification task into a set of smaller classification problems, where each subtask uses an independent feature subset [17]. In 2011, a method for joint feature selection and hierarchical classifier was developed using genetic algorithms [18]. In 2015, a feature selection algorithm was proposed for hierarchical text classification [19]. They did not consider the dependence among different classes in the hierarchical tree, and independently selected features for each node. However, classes in a hierarchical structure have both parent-child and sibling relationships. Classes with a parent-child relationship are similar to each other and may share common features for classification, while distinguishing among classes with a sibling relationship may require different features. However, these algorithms evaluate the importance of features individually.

In this paper, we design a hierarchical feature selection framework with recursive regularization for hierarchical classification. This framework considers the hierarchical information of the class structure. First, we model the loss term for each node with the hierarchical class structure. We then model the hierarchical information as a hierarchical regularization with parent-child relationship, sibling relationship, and family relationship, respectively. We use square loss function to measure the dependencies of hierarchical structure among parent and children. We then use the Hilbert-Schmidt Independence Criterion (HSIC) [20] to measure the independence of the sibling classes, and penalize the dependence among the features selected at sibling nodes. Thus the final subsets are similar if the nodes have a parent-child relationship, while they are different if there is a sibling relationship. The contributions of this paper are summarized as follows.

- We design a hierarchical feature selection framework using the hierarchical class structure and select different feature subsets for different class nodes.
- We attempt to conduct hierarchical feature selection by considering the hierarchical class structure of parent-child relationship, sibling relationship, and family relationship, respectively. These relationships are modeled by hierarchical recursive regularization, which is more reasonable for representing the relationships between nodes than flat approaches.
- In contrast to existing algorithms, we model hierarchical feature selection as a convex objective function and explore an alternation minimization strategy to solve the optimization problem with guaranteed convergence.
- We use six metrics to evaluate the performance of the proposed feature selection algorithms. Extensive experiments on six hierarchical datasets demonstrate the effectiveness of our algorithms in terms of efficiency and accuracy.

The original idea in this work appeared in [21]. We extend it here in the following aspects.

- Based on different hierarchical relationships, we propose Hier-FS, HiRRpar-FS and HiRRsib-FS models respectively. We discuss the relationships among different models.
- We add several hierarchical feature selection methods to compare with our methods.
- Instead of considering local classification accuracy on each node, we include experiments using hierarchical classification to compare the global effectiveness of the proposed models.
- Finally, we include two hierarchical evaluation methods to evaluate the selected feature subsets.

The remainder of this paper is organized as follows. In Section 2, we present some preliminaries about feature selection and hierarchical tree structure. We then present four models of hierarchical feature selection framework with recursive regularization in Section 3. Four hierarchical feature selection algorithms are designed in this section. In Section 4, we discuss experimental results, and analyze the effectiveness of the hierarchical feature selection algorithm. Section 5 concludes this paper and outlines the research direction in the future.

2 Preliminaries

We introduce some basic knowledge on feature selection and hierarchical tree structure in this section.

2.1 Feature Selection

Let $\mathbf{X} \in \mathbf{R}^{m \times n}$ be a data matrix, where m and n are the numbers of samples and features, respectively. We use $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m$ to represent the m samples, where $\mathbf{x}_i \in \mathbf{R}^n$ and $\mathbf{X} = [\mathbf{x}_1; \mathbf{x}_2; \ldots; \mathbf{x}_m]$. Let $\mathbf{Y} \in \mathbf{R}^{m \times d}$ be a class matrix, where d is the number of classes. We use $\mathbf{y}_i \in \{0, 1\}^d$ to represent the class of sample $\mathbf{x}_i$, and $\mathbf{Y} = [\mathbf{y}_1; \mathbf{y}_2; \ldots; \mathbf{y}_m]$.

Feature selection, i.e., the task of choosing a small subset of features which is sufficient to predict the target labels well, is crucial for efficient learning [22], [23]. Many feature selection algorithms can be formulated as a penalized optimization problem

$$\min_{\mathbf{W}} L(\mathbf{XW}, \mathbf{Y}) + \lambda R(\mathbf{W}), \tag{1}$$

where $L(\cdot)$ is the empirical loss function. The least squares loss, logistic loss and hinge loss are three popularly used loss functions in learning. $\mathbf{W} = [w_{ij}] \in \mathbf{R}^{n \times d}$ is the feature weight matrix, and $R(\mathbf{W})$ is the regularizer imposed on $\mathbf{W}$ and λ is a positive constant. The selected features have large weights across all classes.

Recently, sparse learning via ℓ_1 regularization [24] and its various extensions has received increasing attention in many areas including machine learning, signal processing, and statistics [25], [26]. The $\ell_{2,1}$-norm based regression loss function is proposed in [27], which is convex and can be easily optimized. The $\ell_{2,1}$-norm based feature selection method imposes the structural sparsity in feature selection.

2.2 Hierarchical Tree Structure

A hierarchical tree is defined as a pair $(\mathcal{D}, \prec)$, where $\mathcal{D} = \{1, 2, \ldots\}$ is the set of all classes and "$\prec$" represents the "IS-A" relationship, which is the *subclass-of* relationship with the following properties [28]:

TABLE 1
Description of Symbols Used Throughout the Article

Symbol	Meaning
p_i	The parent category of class i
C_i	The set of child categories of class i
S_i	The set of sibling categories of class i
$\|C_i\|$	The number of child categories of class i
$\|S_i\|$	The number of sibling categories of class i

1) Asymmetry: if $i \prec j$ then $j \not\prec i$ for every $i, j \in \mathcal{D}$.
2) Anti-reflexivity: $i \not\prec i$ for every $i \in \mathcal{D}$.
3) Transitivity: if $i \prec j$ and $j \prec k$, then $i \prec k$ for every $i, j, k \in \mathcal{D}$.

In a hierarchical tree structure,

1) p_i is the parent of node $i \in \mathcal{D}$;
2) S_i is the set of all siblings of node $i \in \mathcal{D}$, and $|S_i|$ is the number of the siblings of i;
3) C_i is the set of all children of node $i \in \mathcal{D}$, and $|C_i|$ is the number of the children of i. Table 1 describes the most frequent symbols used throughout this paper.

3 Hierarchical Recursive Regularization for Feature Selection

In this section, we first describe the hierarchical feature selection model with recursive regularization for tasks with a hierarchical tree structure. We also define three models by considering different relationships, namely parent-child, sibling, and family relationships. Four algorithms are designed in this section.

3.1 Basic Framework

In this paper, we focus on hierarchical feature selection based on different hierarchical relationships. Flat feature selection methods assume that the relevant features are shared by all classes. This is too restrictive in real-world applications. In a hierarchical class structure, the classifier at each internal node need only distinguish among a small number of classes [29]. Thus, each subtask is significantly simpler than the original classification task. Hierarchical feature selection aims to learn the features specific to each internal node of the class structure.

Let $\mathbf{X}_i \in \mathbf{R}^{m_i \times n}$ be a data matrix, where m_i is the number of the samples in subtree of node i, and n is the number of features. We use $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{m_i}$ to represent the samples of the ith internal node, where $\mathbf{x}_i \in \mathbf{R}^n$ and $\mathbf{X}_i = [\mathbf{x}_1; \mathbf{x}_2; \ldots; \mathbf{x}_{m_i}]$. Let $\mathbf{Y}_i \in \mathbf{R}^{m_i \times d}$ be a class matrix of the ith internal node, where d is the largest number of sub-classes. We use $\mathbf{y}_i \in \{0,1\}^d$ to represent the class of sample $\mathbf{y}_i$, and $\mathbf{Y}_i = [\mathbf{y}_1; \mathbf{y}_2; \ldots; \mathbf{y}_{m_i}]$.

An example of a tree-based hierarchical class structure is shown in Fig. 1. We compute the feature weight matrix $\mathbf{W}_i$ for each internal node.

From this figure, we observe the following.

1) d_0 is the root of the class tree. $\mathbf{X}_0$ is a data matrix containing all samples in the dataset. The samples are divided into classes d_1 and d_2. We select a feature subset for this node that can distinguish these two classes.

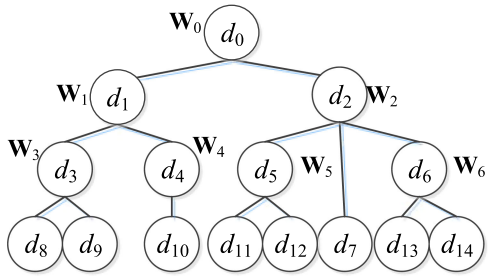

Fig. 1. Tree structure.

2) d_N is one of the internal nodes of the class tree. $\mathbf{X}_N$ is a data matrix containing samples in classes d_{13} and d_{14}.
3) In this tree, node d_2 has the largest number of subclasses, and specifically sub-classes are d_5, d_7, and d_N. This means that $d = 3$.

Let $\|\cdot\|_F$ denote the Frobenius norm of a matrix, and let $\|\cdot\|_{2,1}$ denote the $\ell_{2,1}$-norm of a matrix. In the context of hierarchies, the optimization problem is to minimize $J(\mathbf{W}_0, \mathbf{W}_i, \ldots, \mathbf{W}_N)$

$$J(\mathbf{W}_0, \mathbf{W}_i, \ldots, \mathbf{W}_N) = \sum_{i=0}^{N} (\|\mathbf{X}_i\mathbf{W}_i - \mathbf{Y}_i\|_F^2 + \lambda\|\mathbf{W}_i\|_{2,1}), \quad (2)$$

where $\mathbf{W}_i \in \mathbf{R}^{n \times d}$ is the feature weight matrix, the first term is the loss item, the second term is the regularization imposed on $\mathbf{W}_i$, λ is a positive constant, and N is the number of internal nodes. Linear regression is one of the most broadly and frequently used statistical tools. For convenience, we use least squares loss in this study. A more appropriate convex loss such as hinge loss or logistic loss could be used, but this would require different optimization methods that could not benefit to the closed form solutions used with the mean square error.

$\ell_{2,1}$-norm is already successfully applied in Group Lasso, multi-task feature learning, joint covariance selection and joint subspace selection. We use $\ell_{2,1}$-norm on the parameter $\mathbf{W}_i$, which leads to row-sparsity of the projection matrix. Hence, it is able to discard the irrelevant features and transform the relevant ones simultaneously [27]. The model for hierarchical feature selection is shown in Fig. 2.

It is difficult to derive a closed solution to the optimization problem in Equation (2) directly because of the non-smoothness of the $\ell_{2,1}$-norm. There are a number of convex optimization techniques for dealing with this non-smooth optimization such as Bregman operator splitting algorithm and alternative algorithm. According to [27], this problem can be solved in an alternative way. When $\mathbf{w}_i \neq 0$ for $i = 1, \ldots, d$, the derivative of $||\mathbf{W}||_{2,1}$ with respect to $\mathbf{W}$ is

$$\frac{\partial(||\mathbf{W}||_{2,1})}{\partial \mathbf{W}} = 2\mathbf{D}\mathbf{W}, \quad (3)$$

where $\mathbf{D} \in \mathbf{R}^{d \times d}$ is a diagonal matrix with the ith diagonal element as $\mathbf{D}_{jj} = \frac{1}{2||\mathbf{w}_j||_2}$. We set $\mathbf{D}_{jj} = \epsilon$ when $\mathbf{w}_j = 0$.

It can be easily verified that the derivative in Equation (3) can also be regarded as the derivative of $Tr(\mathbf{W}^T\mathbf{D}\mathbf{W})$.

For the root and internal nodes of the tree, by setting the derivative of Equation (2) with respect to $\mathbf{W}_i$ to 0, we have

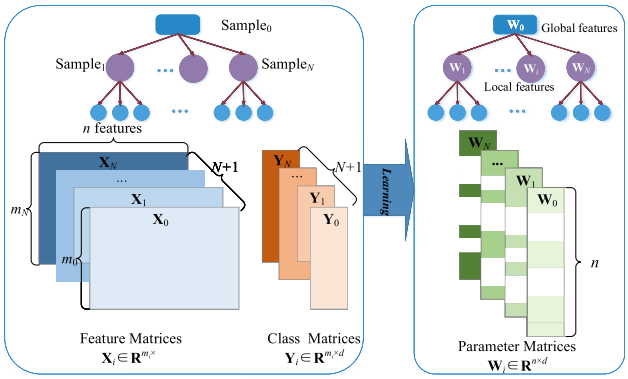

Fig. 2. Model of hierarchical feature selection.

$$\frac{\partial J}{\partial \mathbf{W}_i} = 2\mathbf{X}_i^T(\mathbf{X}_i\mathbf{W}_i - \mathbf{Y}_i) + 2\lambda \mathbf{D}_i\mathbf{W}_i = (2\mathbf{X}_i^T\mathbf{X}_i + 2\lambda\mathbf{D}_i)\mathbf{W}_i - 2\mathbf{X}_i^T\mathbf{Y}_i, \tag{4}$$

where i is the ith internal node of the tree. Therefore, we have

$$\mathbf{W}_i = (\mathbf{X}_i^T\mathbf{X}_i + \lambda\mathbf{D}_i)^{-1}(\mathbf{X}_i^T\mathbf{Y}_i). \tag{5}$$

A hierarchical feature selection algorithm (Hier-FS) is formulated in Algorithm 1. We use a top-down strategy to exploit a priori knowledge of the class structure to divide a large classification task into some small sub-classification tasks. The tree structure consists of a root node, internal nodes, and leaf nodes. We update the root node and internal nodes using Lines 7 to 9. Given a hierarchical structure, the node d_i is the ith internal node, where $i \in \{0, \ldots, N\}$. We consider the child classes of the node d_i to compute the weight vector $\mathbf{W}_i$ when we deal with the current sub-classification task. We focus on selecting the feature subset that can distinguish the child classes of the current node. For example, all the data are divided into two classes d_1 and d_2 for the root classification task according to the hierarchical class structure in Fig. 1. We design different hierarchical feature selection algorithms using different hierarchical class structures in the following sections. This algorithm determines the weight vector $\mathbf{W} = [\mathbf{W}_0, \mathbf{W}_1, \ldots, \mathbf{W}_N]$. We sort the n features for the ith internal node (sub-classification task) in descending order according to $||\mathbf{w}_j^i||_F$ $(j = 1, \ldots, n)$, and select the top-ranked subset for this sub-classification task, where $i = 0, \ldots, N$ and N is the number of internal nodes.

Hier-FS degrades to a conventional feature selection algorithm if there are no internal nodes. The optimization problem degenerates to $\min_{\mathbf{W}}\|\mathbf{XW} - \mathbf{Y}||_F^2 + \lambda\|\mathbf{W}\|_{2,1}$ when $N = 0$. Therefore, the hierarchical feature selection algorithm is a generalization of feature selection.

3.2 Hierarchical Recursive Regularization With Parent-Child Relationship

We incorporate the dependencies of the hierarchical structure into the regularization structure of the parameters. In general, categories from the same subtree share more domain knowledge than those from different subtrees [30]. Parent-child relationship is the closest relationship among all nodes in a tree. That is, they relate to neighboring nodes in the hierarchical structure. We expect that these classes are similar to each other and should share common features for classification. We introduce these relationships into the learning process by incorporating a recursive structure into the regularization term.

Algorithm 1. Hierarchical Feature Selection (Hier-FS)

Input: Input data $\mathbf{X}_i \in \mathbf{R}^{m_i \times n}$ and labels $\mathbf{Y} \in \{0, 1\}^{m_i \times d}$, where $i = 0, 1, \ldots, N$, and N is the number of internal nodes. To facilitate the calculation, we let d be the maximum number of classes of internal nodes. Regularization parameter is λ, and the maximal iteration number is T.
Output: Matrix $\mathbf{W} \in \mathbf{R}^{n \times d(N+1)}$.

1: Set $t = 0$ and initialize $\mathbf{W}_i \in \mathbf{R}^{n \times d}$ randomly;
2: $\mathbf{W} = [\mathbf{W}_0, \mathbf{W}_1, \ldots, \mathbf{W}_N]$;
3: **while** $t < T$ **do**
4: **for** $i = 0 : N$ **do**
5: Compute the diagonal matrix $\mathbf{D}_i^{(t)}$ according to $d_{jj}^i = \frac{1}{2\|\mathbf{w}_j^i\|_2}$;
6: **end for**
// Update the root node and internal nodes.
7: **for** $i = 0 : N$ **do**
8: Update $\mathbf{W}_i$ by $\mathbf{W}_i^{(t+1)} = (\mathbf{X}_i^T\mathbf{X}_i + \lambda\mathbf{D}_i^{(t)})^{-1}(\mathbf{X}_i^T\mathbf{Y}_i)$;
9: **end for**
10: Update $\mathbf{W}^{(t+1)} = [\mathbf{W}_0, \mathbf{W}_1, \ldots, \mathbf{W}_N]$;
11: $t = t + 1$;
12: **end while**
13: return $\mathbf{W}$;

The regularization term for parent-child relationship is

$$\sum_{i=1}^{N}\|\mathbf{W}_i - \mathbf{W}_{p_i}\|_F^2. \tag{6}$$

In the context of hierarchies, the primary optimization problem for parent-child relationship is to minimize $J_{par}(\mathbf{W}_0, \mathbf{W}_i, \ldots, \mathbf{W}_N)$

$$J_{par}(\mathbf{W}_0, \mathbf{W}_i, \ldots, \mathbf{W}_N) = \sum_{i=0}^{N}(\|\mathbf{X}_i\mathbf{W}_i - \mathbf{Y}_i||_F^2 + \lambda\|\mathbf{W}_i\|_{2,1}) + \alpha\sum_{i=1}^{N}\|\mathbf{W}_i - \mathbf{W}_{p_i}\|_F^2. \tag{7}$$

We call this task parent-child relationship based hierarchical recursive regularization for feature selection (HiRRpar-FS).

$$J_{par}(\mathbf{W}_0, \mathbf{W}_i, \ldots, \mathbf{W}_N) = \|\mathbf{X}_0\mathbf{W}_0 - \mathbf{Y}_0\|_F^2 + \lambda\|\mathbf{W}_0\|_{2,1} + \sum_{i=1}^{N}(\|\mathbf{X}_i\mathbf{W}_i - \mathbf{Y}_i\|_F^2 + \lambda\|\mathbf{W}_i\|_{2,1} + \alpha\|\mathbf{W}_i - \mathbf{W}_{p_i}\|_F^2). \tag{8}$$

For the root of the tree, by setting the derivative of Equation (8) with respect to $\mathbf{W}_0$ to 0, we have

$$\frac{\partial J_{par}}{\partial \mathbf{W}_0} = 2\mathbf{X}_0^T(\mathbf{X}_0\mathbf{W}_0 - \mathbf{Y}_0) + 2\lambda\mathbf{D}_0\mathbf{W}_0 - 2\alpha\sum_{i\in C_0}(\mathbf{W}_i - \mathbf{W}_0) = (2\mathbf{X}_0^T\mathbf{X}_0 + 2\lambda\mathbf{D}_0 + 2\alpha|C_0|\mathbf{I})\mathbf{W}_0 - 2\mathbf{X}_0^T\mathbf{Y}_0 - 2\alpha\sum_{i\in C_0}\mathbf{W}_i = 0, \tag{9}$$

where i is the ith child of root node C_0, and $|C_0|$ is the number of all children of root node. Therefore, we have

$$\mathbf{W}_0 = (\mathbf{X}_0^T\mathbf{X}_0 + \lambda\mathbf{D}_0 + \alpha|C_0|\mathbf{I})^{-1}\left(\mathbf{X}_0^T\mathbf{Y}_0 + \alpha\sum_{i\in C_0}\mathbf{W}_i\right). \tag{10}$$

By setting the derivative of Equation (8) with respect to internal node $\mathbf{W}_i$ to 0, we have

$$\begin{aligned}\frac{\partial J_{par}}{\partial \mathbf{W}_i} &= 2\mathbf{X}_i^T(\mathbf{X}_i\mathbf{W}_i - \mathbf{Y}_i) + 2\lambda\mathbf{D}_i\mathbf{W}_i + 2\alpha(\mathbf{W}_i - \mathbf{W}_{p_i}) \\ &= (2\mathbf{X}_i^T\mathbf{X}_i + 2\lambda\mathbf{D}_i + 2\alpha\mathbf{I})\mathbf{W}_i - 2\mathbf{X}_i^T\mathbf{Y}_i - 2\alpha\mathbf{W}_{p_i} = 0.\end{aligned} \tag{11}$$

Therefore, we have

$$\mathbf{W}_i = (\mathbf{X}_i^T\mathbf{X}_i + \lambda\mathbf{D}_i + \alpha\mathbf{I})^{-1}(\mathbf{X}_i^T\mathbf{Y}_i + \alpha\mathbf{W}_{p_i}). \tag{12}$$

We present an algorithm for parent-child relationship based hierarchical feature selection with recursive regularization (HiRRpar-FS) in Algorithm 2. This is a sub-function of Algorithm 1. The input and output are the same as in Algorithm 1 except for the parameter α. The root node is updated separately because the root node does not have a parent node.

Algorithm 2. Parent-Child Relationship Based Hierarchical Feature Selection with Recursive Regularization (HiRRpar-FS)

Input: $\mathbf{W}^{(t)}$. Regularization parameters are λ and α.
Output: An iteration result $\mathbf{W}^{(t+1)}$ of HiRRpar-FS.
1: Update $\mathbf{W}_0$ by $\mathbf{W}_0^{(t+1)} = (\mathbf{X}_0^T\mathbf{X}_0 + \lambda\mathbf{D}_0^{(t)} + \alpha|C_0|\mathbf{I})^{-1}(\mathbf{X}_0^T\mathbf{Y}_0 + \alpha\sum_{i\in C_0}\mathbf{W}_i^{(t)})$; // Update the root node.
// Update the internal nodes.
2: **for** $i = 1 : N$ **do**
3: Update $\mathbf{W}_i$ by $\mathbf{W}_i^{(t+1)} = (\mathbf{X}_i^T\mathbf{X}_i + \lambda\mathbf{D}_i^{(t)} + \alpha\mathbf{I})^{-1}(\mathbf{X}_i^T\mathbf{Y}_i + \alpha\mathbf{W}_{p_i}^{(t)})$;
4: **end for**
5: Update $\mathbf{W}^{(t+1)} = [\mathbf{W}_0, \mathbf{W}_1, \ldots, \mathbf{W}_N]$;

3.3 Hierarchical Recursive Regularization With Sibling Relationship

Sibling relationship is also an important relationship among nodes in a hierarchical class tree structure. Although two internal sibling nodes share a parent node, they each have their own subtree. The classifier for internal nodes should distinguish subcategories. Internal nodes that have a sibling relationship come from different subtrees. Therefore, we should select discriminative features for each subcategory.

Fig. 3 shows an example of a sibling relationship. This example is a subtree of the PASCAL Visual Object Classes (VOC) dataset, which is a benchmark for visual object category recognition and detection and provides the vision and machine learning communities with a standard dataset of images with annotation [31]. For example, textural features can identify animals, but edge features are more representative for furniture. Therefore, we expect the features at sibling nodes to be different from each other.

We measure the independence using a kernel dependence measure (the HSIC) by mapping variables into a reproducing kernel Hilbert space (RKHS). This criterion measures the high-order joint moments between the original distributions [32]. We use the HSIC to penalize the

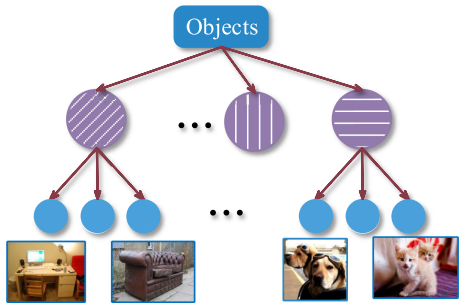

Fig. 3. Example of sibling relationship.

dependence between the selected features at sibling nodes in an RKHS.

Let $\mathbf{K}_i$ and $\mathbf{K}_l$ be kernel matrices on $\mathbf{W}_i \in \mathbf{R}^{n\times d}$ and $\mathbf{W}_l \in \mathbf{R}^{n\times d}$, where $l \in S_i$ is the lth sibling node of node i. Matrixes $\mathbf{W}_i$ and $\mathbf{W}_l$ are the projection matrices for the ith node and the lth sibling node, respectively. Then

$$\mathrm{HSIC}(\mathbf{W}_i, \mathbf{W}_l) = tr(\mathbf{K}_i\mathbf{H}\mathbf{K}_l\mathbf{H}), \tag{13}$$

where $\mathbf{K}_i = \mathbf{W}_i\mathbf{W}_i^T$, $\mathbf{K}_l = \mathbf{W}_l\mathbf{W}_l^T$, and $1 \le l \le |S_i|$. $\mathbf{H} = \mathbf{I} - \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^T \in \mathbf{R}^{n\times n}$ centers both the feature space and the sample space to have zero mean, where $\mathbf{1}_n \in \mathbf{R}^n$ is a column vector with all elements being 1.

The primary optimization problem for sibling relationship is to minimize $J_{sib}(\mathbf{W}_0, \mathbf{W}_i, \ldots, \mathbf{W}_N)$

$$\begin{aligned}J_{sib}(\mathbf{W}_0, \mathbf{W}_i, \ldots, \mathbf{W}_N) = &\sum_{i=0}^{N}(\|\mathbf{X}_i\mathbf{W}_i - \mathbf{Y}_i\|_F^2 + \lambda\|\mathbf{W}_i\|_{2,1}) \\ &+ \beta\sum_{i=1}^{N}\sum_{l\in S_i}\mathrm{HSIC}(\mathbf{W}_i, \mathbf{W}_l),\end{aligned} \tag{14}$$

where l is a sibling of node i and $|S_i|$ is the number of siblings of node i. We call this task sibling relationship based hierarchical recursive regularization for feature selection (HiRRsib-FS).

The root node needs to be computed separately. Therefore, the objective function can be rewritten as

$$\begin{aligned}J_{sib}(\mathbf{W}_0, \mathbf{W}_i, \ldots, \mathbf{W}_N) = &\|\mathbf{X}_0\mathbf{W}_0 - \mathbf{Y}_0\|_F^2 + \lambda Tr(\mathbf{W}_0^T\mathbf{D}_0\mathbf{W}_0) \\ &+ \sum_{i=1}^{N}(\|\mathbf{X}_i\mathbf{W}_i - \mathbf{Y}_i\|_F^2 + \lambda Tr(\mathbf{W}_i^T\mathbf{D}_i\mathbf{W}_i)) \\ &+ \beta\sum_{i=1}^{N}\sum_{l\in S_i}Tr(\mathbf{W}_i\mathbf{W}_i^T\mathbf{H}\mathbf{W}_l\mathbf{W}_l^T\mathbf{H}).\end{aligned} \tag{15}$$

For the root of the tree, by setting the derivative of Equation (15) with respect to $\mathbf{W}_0$ to 0, we have

$$\begin{aligned}\frac{\partial J_{sib}}{\partial \mathbf{W}_0} &= 2\mathbf{X}_0^T(\mathbf{X}_0\mathbf{W}_0 - \mathbf{Y}_0) + 2\lambda\mathbf{D}_0\mathbf{W}_0 \\ &= (2\mathbf{X}_0^T\mathbf{X}_0 + 2\lambda\mathbf{D}_0)\mathbf{W}_0 - 2\mathbf{X}_0^T\mathbf{Y}_0 = 0.\end{aligned} \tag{16}$$

Therefore, we have

$$\mathbf{W}_0 = (\mathbf{X}_0^T\mathbf{X}_0 + \lambda\mathbf{D}_0)^{-1}(\mathbf{X}_0^T\mathbf{Y}_0). \tag{17}$$

Let $\mathbf{U}_l = \mathbf{H}\mathbf{W}_l\mathbf{W}_l^T\mathbf{H}$. By setting the derivative of Equation (15) with respect to internal node $\mathbf{W}_i$ to 0, we have

$$\frac{\partial J_{sib}}{\partial \mathbf{W}_i} = 2\mathbf{X}_i^T(\mathbf{X}_i\mathbf{W}_i - \mathbf{Y}_i) + 2\lambda\mathbf{D}_i\mathbf{W}_i + \beta\sum_{l\in S_i}(\mathbf{U}_l + \mathbf{U}_l^T)\mathbf{W}_i$$
$$= (2\mathbf{X}_i^T\mathbf{X}_i + 2\lambda\mathbf{D}_i + \beta\sum_{l\in S_i}(\mathbf{U}_l + \mathbf{U}_l^T))\mathbf{W}_i - 2\mathbf{X}_i^T\mathbf{Y}_i = 0. \tag{18}$$

Finally, we have

$$\mathbf{W}_i = (\mathbf{X}_i^T\mathbf{X}_i + \lambda\mathbf{D}_i + \beta\sum_{l\in S_i}(\mathbf{U}_l + \mathbf{U}_l^T))^{-1}(\mathbf{X}_i^T\mathbf{Y}_i). \tag{19}$$

We present an algorithm for sibling relationship based hierarchical feature selection with recursive regularization (HiRRsib-FS) in Algorithm 3. This is a sub-function of Algorithm 1. The input and output are the same as in Algorithm 1 except for parameter β. The update process for the root node is same as in Algorithm 1 because the root node does not have sibling nodes.

Algorithm 3. Sibling Relationship Based Hierarchical Feature Selection with Recursive Regularization (HiRRsib-FS)

Input: $\mathbf{W}^{(t)}$. Regularization parameters are λ and β.
Output: An iteration result $\mathbf{W}^{(t+1)}$ of HiRRsib-FS.
1: Update $\mathbf{W}_0$ by $\mathbf{W}_0^{(t+1)} = (\mathbf{X}_0^T\mathbf{X}_0 + \lambda\mathbf{D}_0^{(t)})^{-1}(\mathbf{X}_0^T\mathbf{Y}_0)$; //Update the root node.
// Update the internal nodes.
2: **for** $i = 1 : N$ **do**
3: Update $\mathbf{W}_i$ by $\mathbf{W}_i^{(t+1)} = (\mathbf{X}_i^T\mathbf{X}_i + \lambda\mathbf{D}_i^{(t)} + \beta\sum_{l\in S_i}(\mathbf{U}_l + \mathbf{U}_l^T))^{-1}(\mathbf{X}_i^T\mathbf{Y}_i)$;
4: **end for**
5: Update $\mathbf{W}^{(t+1)} = [\mathbf{W}_0, \mathbf{W}_1, \ldots, \mathbf{W}_N]$;

3.4 Hierarchical Recursive Regularization With Family Relationship

In this section, we consider parent-child and sibling relationships between categories in a hierarchy simultaneously. We expect that parent and children share common features. In addition, we also expect that internal nodes that have a sibling relationship have their own unique features.

The primary optimization problem when considering parent-child and sibling relationships simultaneously is to minimize $J_{fam}(\mathbf{W}_0, \mathbf{W}_i, \ldots, \mathbf{W}_N)$ [21]

$$J_{fam}(\mathbf{W}_0,\mathbf{W}_i,\ldots,\mathbf{W}_N) = \sum_{i=0}^{N}(\|\mathbf{X}_i\mathbf{W}_i - \mathbf{Y}_i\|_F^2 + \lambda\|\mathbf{W}_i\|_{2,1})$$
$$+ \alpha\sum_{i=1}^{N}\|\mathbf{W}_i - \mathbf{W}_{p_i}\|_F^2 + \beta\sum_{i=1}^{N}\sum_{l\in S_i}\mathrm{HSIC}(\mathbf{W}_i, \mathbf{W}_l), \tag{20}$$

where $i = 0$ indicates a root node with no parent node or sibling nodes. Therefore, the value of i in the two regularization terms starts at 1. We call this task family relationship based hierarchical recursive regularization for feature selection (HiRRfam-FS).

The root node needs to be computed separately. Therefore, the objective function is rewritten as

$$J_{fam}(\mathbf{W}_0, \mathbf{W}_i, \ldots, \mathbf{W}_N) = \|\mathbf{X}_0\mathbf{W}_0 - \mathbf{Y}_0\|_F^2 + \lambda Tr(\mathbf{W}_0^T\mathbf{D}_0\mathbf{W}_0)$$
$$+ \sum_{i=1}^{N}(\|\mathbf{X}_i\mathbf{W}_i - \mathbf{Y}_i\|_F^2 + \lambda Tr(\mathbf{W}_i^T\mathbf{D}_i\mathbf{W}_i)) + \alpha\|\mathbf{W}_i - \mathbf{W}_{p_i}\|_F^2$$
$$+ \beta\sum_{i=1}^{N}\sum_{l\in S_i}Tr(\mathbf{W}_i\mathbf{W}_i^T\mathbf{H}\mathbf{W}_l\mathbf{W}_l^T\mathbf{H}). \tag{21}$$

For the root of the tree, by setting the derivative of Equation (21) with respect to $\mathbf{W}_0$ to 0, we have

$$\frac{\partial J_{fam}}{\partial \mathbf{W}_0} = 2\mathbf{X}_0^T(\mathbf{X}_0\mathbf{W}_0 - \mathbf{Y}_0) + 2\lambda\mathbf{D}_0\mathbf{W}_0$$
$$= (2\mathbf{X}_0^T\mathbf{X}_0 + 2\lambda\mathbf{D}_0)\mathbf{W}_0 - 2\mathbf{X}_0^T\mathbf{Y}_0 = 0, \tag{22}$$

where i is the ith child of root node C_0, and $|C_0|$ is the number of all children of root node. Therefore, we have

$$\mathbf{W}_0 = (\mathbf{X}_0^T\mathbf{X}_0 + \lambda\mathbf{D}_0 + \alpha|C_0|\mathbf{I})^{-1}\left(\mathbf{X}_0^T\mathbf{Y}_0 + \alpha\sum_{i\in C_0}\mathbf{W}_i\right). \tag{23}$$

Let $\mathbf{U}_l = \mathbf{H}\mathbf{W}_l\mathbf{W}_l^T\mathbf{H}$. By setting the derivative of Equation (21) with respect to internal node $\mathbf{W}_i$ to 0, we have

$$\frac{\partial J}{\partial \mathbf{W}_i} = 2\mathbf{X}_i^T(\mathbf{X}_i\mathbf{W}_i - \mathbf{Y}_i) + 2\lambda\mathbf{D}_i\mathbf{W}_i + 2\alpha(\mathbf{W}_i - \mathbf{W}_{p_i})$$
$$+ \beta\sum_{l\in S_i}(\mathbf{U}_l + \mathbf{U}_l^T)\mathbf{W}_i$$
$$= \left(2\mathbf{X}_i^T\mathbf{X}_i + 2\lambda\mathbf{D}_i + 2\alpha\mathbf{I} + \beta\sum_{l\in S_i}(\mathbf{U}_l + \mathbf{U}_l^T)\right)\mathbf{W}_i$$
$$- 2\mathbf{X}_i^T\mathbf{Y}_i - 2\alpha\mathbf{W}_{p_i} = 0. \tag{24}$$

Finally, we have

$$\mathbf{W}_i = \left(\mathbf{X}_i^T\mathbf{X}_i + \lambda\mathbf{D}_i + \alpha\mathbf{I} + \beta\sum_{l\in S_i}(\mathbf{U}_l + \mathbf{U}_l^T)\right)^{-1}(\mathbf{X}_i^T\mathbf{Y}_i + \alpha\mathbf{W}_{p_i}). \tag{25}$$

We present an algorithm for family relationship based hierarchical feature selection with recursive regularization (HiRRfam-FS) in Algorithm 4. This is a sub-function of Algorithm 1. The input and output are the same as in Algorithm 1 except for the parameters α and β.

Algorithm 4. Family Relationship Based Hierarchical Feature Selection with Recursive Regularization (HiRRfam-FS)

Input: $\mathbf{W}^{(t)}$. Regularization parameters λ, α, and β.
Output: An iteration result $\mathbf{W}^{(t+1)}$ of HiRRfam-FS.
1: Update $\mathbf{W}_0$ by $\mathbf{W}_0^{(t+1)} = (\mathbf{X}_0^T\mathbf{X}_0 + \lambda\mathbf{D}_0^{(t)} + \alpha|C_0|\mathbf{I})^{-1}(\mathbf{X}_0^T\mathbf{Y}_0 + \alpha\sum_{i\in C_0}\mathbf{W}_i^{(t)})$; //Update the root node.
// Update the internal nodes.
2: **for** $i = 1 : N$ **do**
3: Update $\mathbf{W}_i$ by $\mathbf{W}_i^{(t+1)} = (\mathbf{X}_i^T\mathbf{X}_i + \lambda\mathbf{D}_i^{(t)} + \alpha\mathbf{I} + \beta\sum_{l\in S_i}(\mathbf{U}_l + \mathbf{U}_l^T))^{-1}(\mathbf{X}_i^T\mathbf{Y}_i + \alpha\mathbf{W}_{p_i}^{(t)})$;
4: **end for**
5: Update $\mathbf{W}^{(t+1)} = [\mathbf{W}_0, \mathbf{W}_1, \ldots, \mathbf{W}_N]$;

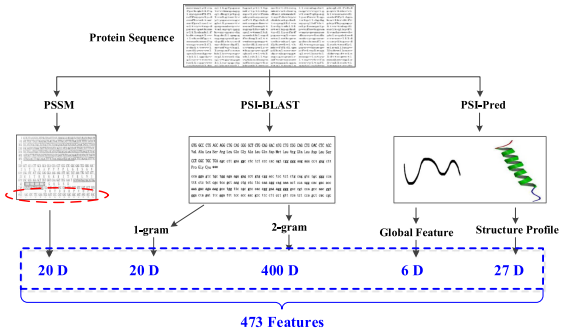

Fig. 4. Feature distribution of *DD* dataset.

From Algorithm 4, the time complexity for each iteration largely depends on the calculation for updating $\mathbf{W}$. This requires $O(n^3 + n^2d + n^2m_i + nm_id)$ operations for the ith internal node in each iteration, where n is the number of features, d is the maximal number of classes, and m_i is the number of samples for the ith internal node. In our experiments, each of $\mathbf{X}_i^T\mathbf{X}_i$ and $\mathbf{X}_i^T\mathbf{Y}_i$ for the ith internal node is prepared once. These require n^2m_i and nm_id operations, respectively. For all internal nodes, this requires n^2m and nmd, where m is the number of samples for all internal nodes. Hence, the time complexity of HiRRfam-FS is $O(R(n^3 + n^2d) + n^2m + nmd)$, where R is the total number of iterations.

4 Experiments and Discussion

In this section, we introduce the datasets, evaluation measures, and experiment setup used in our experiments.[1] We then present four metrics to verify the effectiveness of the proposed hierarchical feature selection framework. Finally, we analyze the convergence of the algorithm.

4.1 Datasets

There are six datasets used in the experiments, including two protein datasets and four image datasets. All these datasets are single-labeled, and the examples are assigned to the leaf nodes in the hierarchy. All of the tasks incorporate information about the class hierarchy. For two protein datasets, Fig. 4 is the feature distribution, which is extracted from [33]. There are 473 features (five group: amino acid, 1-gram, 2-gram, global, local) in two protein datasets, where from 441 to 446 are global features. For *VOC* dataset, we use the features from [31]. For other image datasets, we extract features with the VGG19 model [34] pre-trained with the ImageNet dataset.[2]

Detailed information about the datasets is summarized in Table 2. The number of features varies from 473 to 4,096.

The first dataset is a protein dataset called *F194* [35]. It has 8,525 samples and 473 features. There are 194 classes in this dataset, which are all leaf nodes. The second dataset is *DD* dataset [36], which is also a protein dataset. It has 27 real classes and four major structural classes.

1. The code and data underlying this study have been uploaded to Github and are accessible using the following link: https://github.com/fhqxa/TKDE2019.

2. The VGG19 model and its parameters can be downloaded from: https://github.com/SnailTyan/caffe-model-zoo/tree/master/VGG19.

TABLE 2
Data Description

No.	Dataset	Train	Test	Feature	Node	Leaf	Height
1	*F194*	7,105	1,420	473	202	194	3
2	*DD*	3,020	605	473	32	27	3
3	*VOC*	7,178	5,105	1,000	30	20	5
4	*Cifar100*	50,000	10,000	4,096	121	100	3
5	*SUN*	45,109	22,556	4,096	343	324	4
6	*ILSVRC65*	12,346	11,845	4,096	65	57	4

The third dataset is the PASCAL *VOC* dataset. Fig. 5 shows the hierarchy for *VOC*. In Table 2, there are 7,178 samples in the training dataset and 5,105 samples in the test dataset for PASCAL *VOC* [31]. The fourth dataset is *Cifar100* [37], which contains labeled subsets of 80 million tiny image datasets collected by Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. This dataset has 100 classes containing 600 images each. There are 500 training images and 100 test images per class. The 100 classes in *Cifar100* are grouped into 20 superclasses. Each image comes with a "fine" label (leaf node) and a "coarse" label (the superclass to which it belongs). The hierarchical class structure is shown in Fig. 6.

The fifth dataset is Scene UNderstanding (*SUN*) [38], [39] which is an extensive database that contains 397 well-sampled classes. There are multi-label objects in the *SUN* dataset. As we do not discuss this kind of task in this study, we remove these multi-label samples. The sixth dataset is *ILSVRC65* dataset [40]. There are 65 class nodes in this hierarchical structure and the number of internal nodes is $N = 7$.

4.2 Evaluation Measures

The proposed method implements hierarchical classification, which is different from flat classification. Accordingly, the evaluation measures for hierarchical feature selection should be different. Some measures for evaluating hierarchical classification were introduced in [28]. We select the feature subsets from the training sets and test them on the test sets. We also perform 10-fold cross-validation with the same parameter setup. We use the hierarchical F_1-measure to evaluate the experimental results and use the tree induced error to consider different errors caused by the hierarchy [28].

In hierarchical classification, different classification errors result from different levels of penalty. In our model, this penalty is defined by the tree distance, which is called the *tree induced error* (TIE) in [41]. The TIE is computed by predicting the label d_v when the correct label is d_u

$$TIE(d_u, d_v) = |E_H(d_u, d_v)|, \tag{26}$$

where $E_H(d_u, d_v)$ is the set of edges along the path from d_u to d_v in the hierarchy, and $|\cdot|$ denotes the count of elements. That is, $TIE(d_u, d_v)$ is defined to be the number of edges along the path from d_u to d_v in the tree of D.

The hierarchical F_1-measure takes into account entire sets of predicted and true classes including their ancestors or descendants. Hierarchical F_1-measure has two distinct phases: (1) The augmentation of D and $\hat{D}$ with information about the hierarchy. (2) The calculation of a cost measure based on the augmented sets.

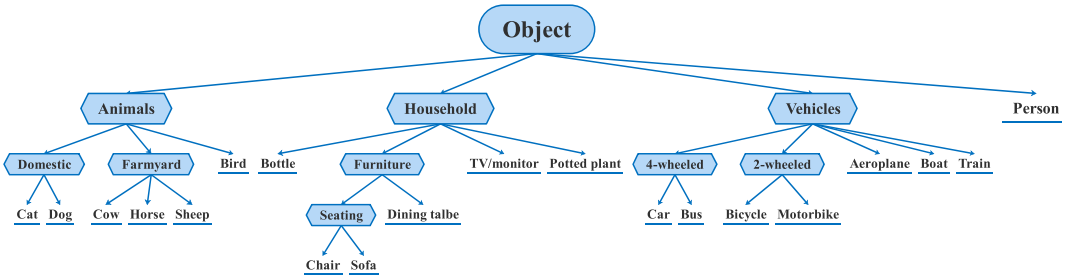

Fig. 5. Hierarchy of PASCAL *VOC* dataset.

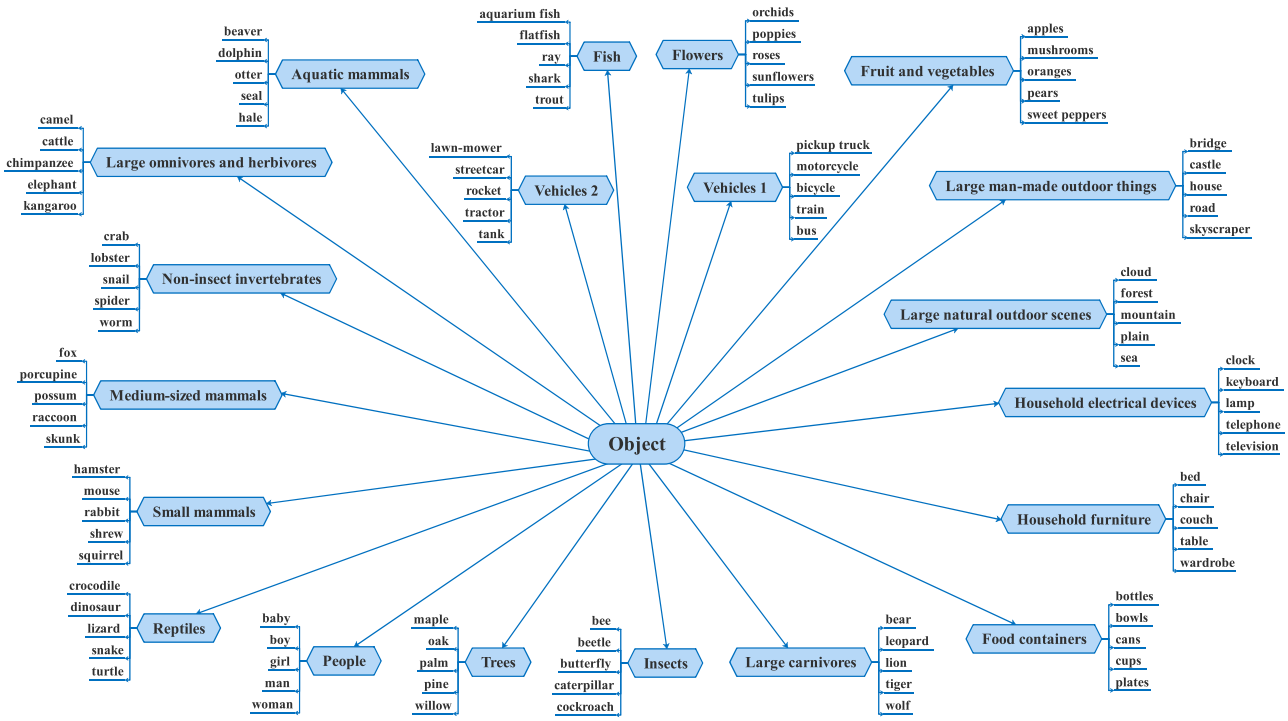

Fig. 6. Hierarchy of *Cifar100* dataset.

The augmentation of D and $\hat{D}$ is a crucial step that attempts to capture the hierarchical relationships between classes. There are different measures based on different ways of augmenting the sets of predicted and true classes. We select a measure that augments the sets with the ancestors of the true and predicted classes [42], [43] as follows: $D_{aug} = D \cup anc(D)$, and $\hat{D}_{aug} = \hat{D} \cup anc(\hat{D})$.

Hierarchical precision and recall are defined as follows:

$$P_H = \frac{|\hat{D}_{aug} \cap D_{aug}|}{|\hat{D}_{aug}|}, \qquad R_H = \frac{|\hat{D}_{aug} \cap D_{aug}|}{|D_{aug}|}, \tag{27}$$

where $|\cdot|$ denotes the count of elements. The hierarchical F_1-measure is defined as follows:

$$F_H = \frac{2 \cdot P_H \cdot R_H}{P_H + R_H}. \tag{28}$$

4.3 Experiment Setup

To evaluate the performance of our hierarchical feature selection algorithms, we compare them with other hierarchical feature selection algorithms. We set up our experimental study as follows:

1) Baseline: All original features are selected.
2) The proposed algorithms are compared with three hierarchical feature selection algorithms. (1) HierMRMR [44] is a hierarchical feature selection method based on mRMR [45], which can be run independently for each classifier. This method actually selects different sets of features for each sub-classifier. (2) HierFisher is a hierarchical feature selection method modified from the Fisher score [46]. (3) HierFSNM is a hierarchical feature selection method modified from FSNM [47]. We take FSNM algorithm and HierFSNM algorithm as examples to introduce the difference between the flat algorithm and the hierarchical classification algorithm. FSNM selects a feature subset for all classes. The hierarchical class structure divides a large classification task into a set of relatively small and easy sub-classification tasks. In HierFSNM, we recursively execute FSNM on the hierarchical classification from root classification task until the leaf

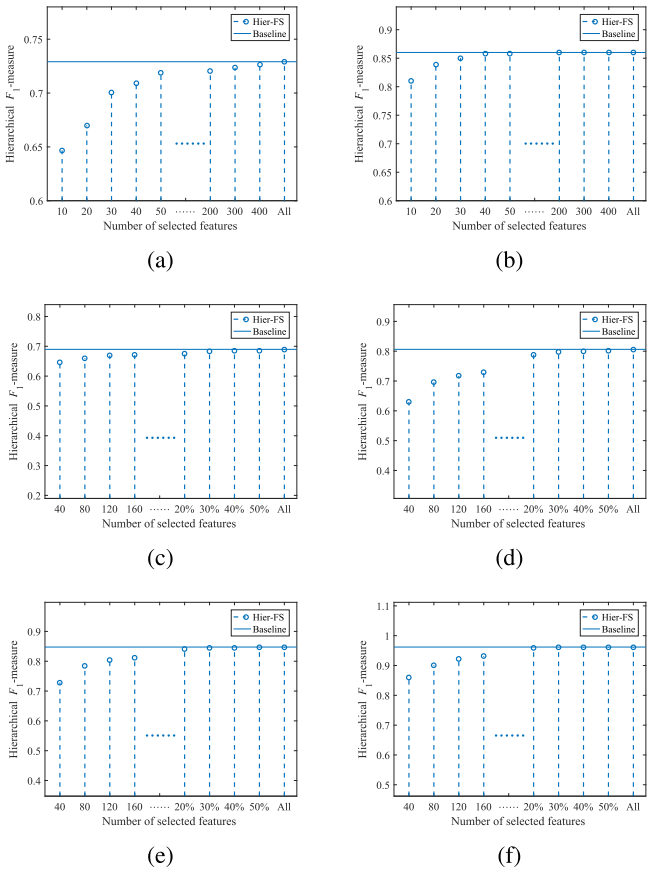

Fig. 7. Hierarchical F_1-measure on different features on each internal node. (a) *F194*; (b) *DD*; (c) *VOC*; (d) *Cifar100*; (e) *SUN*; (f) *ILSVRC65*.

node. We can select the features for current sub-classification task.

3) We use a top-down SVM classifier to test the effectiveness of our algorithms. For the SVM classifier, 10-fold cross-validation is performed using a linear kernel and $c = 1$. For all algorithms, we use different feature subsets for each internal node. Additionally, we use the TIE and hierarchical F_1-measure to evaluate the feature selection algorithms.
4) There are four algorithms: Hier-FS, HiRRpar-FS, HiRRsib-FS, and HiRRfam-FS. We set $\lambda = 10$ for all the algorithms, and set $\alpha = 1$ for HiRRpar-FS and HiRRfam-FS. We set $\beta = 1$ for the HiRRsib-FS and HiRRfam-FS. For parameter sensitivity analysis, we vary α and β over the set $\{0.01, 0.1, 1, 10, 100\}$.

4.4 Results and Analysis

We use seven metrics to evaluate the performance of the proposed feature selection algorithms: (1) a performance comparison with the baseline method (all features are selected); (2) a performance comparison using TIE and the hierarchical F_1-measure evaluation; (3) a performance comparison for the regularization terms; (4) the running time for obtaining the feature subset; (5) parameter sensitivity analysis; (6) results on text dataset; and (7) convergence analysis.

4.4.1 Performance Comparison With the Baseline Method

To verify the effectiveness of the proposed methods, comparison experiments are performed between our algorithm and the baseline. We select the feature subset for each internal node, which is the sub-classification task. A top-down classifier is used to test the selected features for each sub-classification from the different numbers of selected features. Fig. 7 is used to compare the hierarchical F_1-measure of the feature selection method with that of all features (baseline algorithm). From these figures, we have the following observations. Hier-FS reduces the dimensions of data without changing the hierarchical F_1-measure. For *DD*, 40 features (approximately 10 percent) selected by Hier-FS achieve the same hierarchical F_1-measure as all features. For four image datasets, 20 percent of features achieve the same hierarchical F_1-measure as the baseline algorithm.

TABLE 3
Running Time (s) of the Classification Between With and Without Feature Selection

Dataset	20% features	ALL features
F194	0.709	0.729
DD	0.858	0.860
VOC	256	1041
Cifar100	1172	6552
SUN	779	3925
ILSVRC65	4855	27232

Furthermore, Table 3 shows the running time of the classification between the approaches with and without feature selection. For two protein datasets, this table lists the running time comparison of the classifier based on 40 features (approximately 10 percent) and all features. For four image datasets, this table lists the running time comparison of the classifier based on 20 percent of all features and all features. The classification complexity substantially reduces after hierarchical feature selection. Specifically, the running time of classification after feature selection on *VOC*, *SUN*, and *ILSVRC65* is an order of magnitude faster than that of the baseline method. From these results, we select 20 percent of features for four image datasets and 10 percent of features for two protein datasets.

4.4.2 Performance Comparison of Different Algorithms

First, we compare the effectiveness of the three hierarchical different feature selection algorithms with the Hier-FS and HiRRfam-FS algorithms based on TIE and hierarchical F_1-measure evaluation. TIE results are affected by the number of samples in the testing dataset, which is not helpful to the comparison among different datasets. Therefore, we removed the influence of the number of samples and computed the normalized TIE results on the testing datasets. It is helpful to reduce the range changes substantially among the datasets.

The normalized TIE results of different feature selection algorithms on different datasets are listed in Table 4. The best results are highlighted in bold, and the smaller the better. The experiments show that our algorithm, with the family relationship, dramatically outperforms other hierarchical algorithms for all datasets. HiRRfam-FS performs better and stabler than Hier-FS in most cases.

The hierarchical F_1-measure results of different feature selection algorithms on different datasets are listed in Table 5. Regarding the hierarchical F_1-measure, the higher the better. We can draw the same conclusion as that for TIE. Finally, we

TABLE 4
Normalized TIE Results of Different Feature Selection Algorithms on Different Datasets

Dataset	HierFisher	HierFSNM	HierMRMR	Hier-FS	HiRRfam-FS
F194	0.1945	0.2123	0.1800	0.1746	**0.1730**
DD	0.1355	0.0886	0.0919	0.0850	**0.0836**
VOC	0.2271	0.2144	0.2188	0.2143	**0.2138**
Cifar100	0.1285	—	0.1273	**0.1269**	0.1272
SUN	0.1341	—	0.1322	0.1280	**0.1271**
ILSVRC65	0.0336	0.0350	0.0335	**0.0328**	0.0329

TABLE 5
Hierarchical F_1-Measure Results of Different Feature Selection Algorithms on Different Datasets

Dataset	HierFisher	HierFSNM	HierMRMR	Hier-FS	HiRRfam-FS
F194	0.6758(4)	0.6462(5)	0.7000(3)	0.7089(2)	**0.7117(1)**
DD	0.7741(5)	0.8524(3)	0.8468(4)	0.8584(2)	**0.8606(1)**
VOC	0.6576(5)	0.6739(3)	0.6669(4)	0.6754(2)	**0.6758(1)**
Cifar100	0.7859(4)	—(5)	0.7879(3)	**0.7885(1)**	0.7880(2)
SUN	0.8324(4)	—(5)	0.8348(3)	0.8400(2)	**0.8411(1)**
ILSVRC65	0.9580(4)	0.9563(5)	0.9581(3)	**0.9591(1)**	0.9588(2)
Avg. Rank	4.33	4.33	3.33	1.67	1.33

introduce statistical methods to evaluate the performance of these algorithms. Demšar [48] advised statistically comparing algorithms on multiple datasets using the Friedman test [49] followed by the Bonferroni-Dunn test [50]. We perform these two tests to explore the statistical significance of the results. The Friedman test is a non-parametric test that can be used to compare k algorithms on N datasets by ranking each algorithm separately on each dataset. The best performing algorithm is given a rank of 1, the second best is given a rank of 2, and so on. In the case of ties (equal data values), average ranks are assigned.

Table 5 shows the hierarchical F_1-measure ranks for different algorithms. The average rank of each algorithm on all the datasets is computed. The null-hypothesis of the Friedman test is that all the feature algorithms are equivalent in terms of the hierarchical F_1-measure. Under the null-hypothesis, we compute the value $F_F = 21.94$, which is distributed according to an F distribution with $k-1$ and $(k-1)(N-1)$ degrees of freedom, where k is the number of methods, and N is the number of datasets. With five algorithms and six datasets, the critical value of $F(5-1,(5-1)\times(6-1)) = F(4,20)$ for $\alpha = 0.05$ is 2.866, so we reject the null-hypothesis. Thus, the five methods under comparison are not equivalent and there are significant differences between the methods.

The Bonferroni-Dunn post-hoc test is used to detect whether the proposed method is better than the existing methods. The performance of the two compared algorithms are significantly different if the distance between the average ranks exceeds the critical distance (CD) $CD_\alpha = q_\alpha\sqrt{\frac{k(k+1)}{6N}}$, where q_α is given in Table 5 in [48]. Note that $q_{0.1} = 2.241$ when $k = 5$. Therefore, $CD_{0.1} = q_{0.1}\sqrt{\frac{5\times6}{6\times6}} = 2.046$.

Fig. 8 shows the results of the Bonferroni-Dunn post-hoc test for $\alpha = 0.1$ on the six datasets. The results indicate that the hierarchical F_1-measure for HiRRfam-FS is statistically better than those for HierFSNM and HierFisher. There is no consistent evidence to indicate statistical differences between

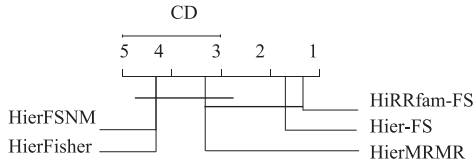

Fig. 8. Performance comparison of Hier-FS algorithm against the others with the Bonferroni-Dunn test.

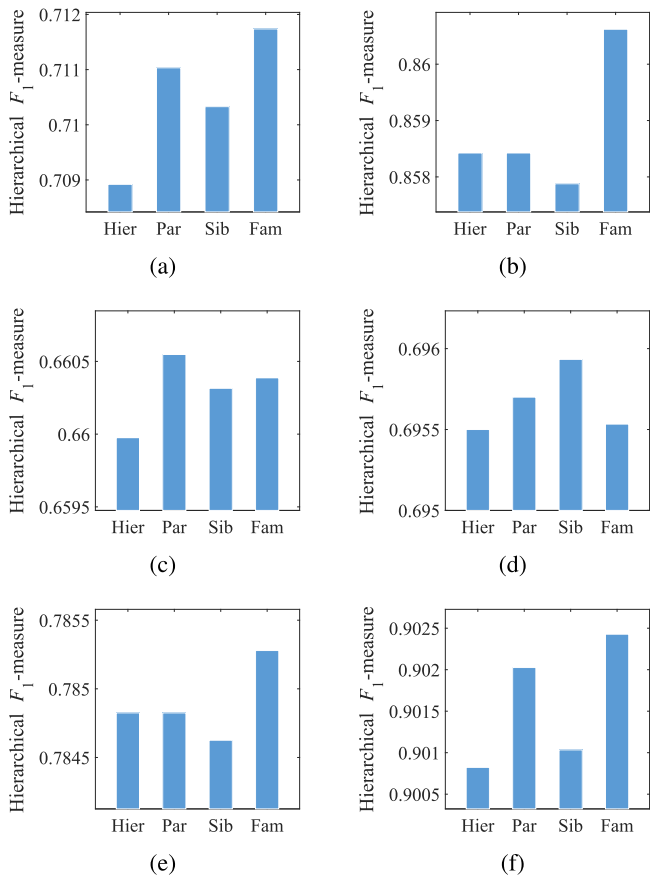

Fig. 9. Hierarchical F_1-measure on different algorithms. (a) *F194*; (b) *DD*; (c) *VOC*; (d) *Cifar100*; (e) *SUN*; (f) *ILSVRC65*.

the hierarchical F_1-measure among HiRRfam-FS, Hier-FS, and HierMRMR.

4.4.3 Performance Comparison for Regularization Terms

We investigate the performance of the regularization terms in our four algorithms experimentally. We fix $\lambda = 10$ and compare the effectiveness of Hier-FS, HiRRpar-FS ($\alpha = 1$), HiRRsib-FS ($\beta = 1$), and HiRRfam-FS ($\alpha = 1$ and $\beta = 1$) for all datasets.

Figs. 9a and 9b show the hierarchical F_1-measure using 40 features for the two protein datasets. The results for the two protein datasets demonstrate that HiRRfam-FS, which considers the regularization terms for parent-child and sibling relationships, works well. For the two protein datasets, the performance of the HiRRpar-FS algorithm is better than that of the HiRRsib-FS algorithm. The results are better when we consider family relationship rather than only sibling relationship.

TABLE 6
The Selected 24 Features from Each Non-Leaf Node of the *DD* Dataset

No.	Root	$Node_1$	$Node_2$	$Node_3$	$Node_4$	No.	Root	$Node_1$	$Node_2$	$Node_3$	$Node_4$
1	425	421	422	421	421	13	451	435	437	439	452
2	426	422	423	422	422	14	452	436	438	440	453
3	430	423	425	423	424	15	454	439	451	**446**	454
4	433	424	427	424	425	16	455	440	455	448	456
5	437	425	428	426	426	17	457	448	458	449	457
6	**441**	426	430	427	431	18	458	454	460	451	458
7	**443**	427	431	428	433	19	460	457	463	452	462
8	**444**	428	432	429	434	20	461	463	464	453	463
9	**445**	429	433	430	437	21	463	464	465	457	465
10	**446**	431	434	432	440	22	464	467	467	460	466
11	449	433	435	434	449	23	467	468	469	467	469
12	450	434	436	435	451	24	469	469	470	470	470

Figs. 9c, 9d, 9e, and 9f show the hierarchical F_1-measure with 80 features for four image datasets *VOC*, *Cifar100*, *SUN*, and *ILSVRC65*. The performance of sibling relationship is not good on most datasets, except *Cifar100*. However, the performance is best when we consider both the parent-child and the sibling relationships among the four algorithms. Thus, the independence among siblings is based on the dependence between parent and children.

The proposed methods select the feature subset for each internal node, which is the sub-classification task. To introduce the selected features of each node, we give a semantic interpretation of the selected features for *DD* dataset according to family-relationship regularization. We select 24 features for each node from *DD* as our example. The results of selected features are listed in Table 6. In *DD*, there are 473 features, where features from 441 to 446 are global features. From Table 6, the selected features include {441, 443, 444, 445, 446} which are almost all global features for the root node. There are from 9 to 12 same features between root node and each internal node. For the 1st and 2nd internal nodes, there are 10 different features to classify their subclasses, where 6 features are 2-gram features and 4 features are local features.

4.4.4 Efficiency Comparison

We next compare the efficiency of the four different feature selection algorithms with the Hier-FS and HiRRfam-FS algorithms using running time. All experiments were executed on an Intel Core i7-3770 running at 3.40 GHz with 16.0 GB memory on a 64-bit Windows 7 operating system.

The results presented in Table 7 demonstrate that the Hier-FS algorithm has a significantly shorter running time than the other algorithms, except for the HierFisher algorithm. HierFSNM cannot process the *Cifar100* and *SUN* datasets with running out of memory. Although HierFisher is faster than our methods, the performance is not good, which is shown in Tables 4 and 5.

TABLE 7
Running Time (s) of the Five Feature Selection Algorithms

Dataset	HierFisher	HierFSNM	HierMRMR	Hier-FS	HiRRfam-FS
F194	5.83(3)	112.6(5)	50.93(4)	0.96(1)	5.59(2)
DD	0.95(2)	27.24(5)	23.87(4)	0.52(1)	1.66(3)
VOC	0.73(1)	122.1(4)	251.9(5)	6.42(2)	13.4(3)
Cifar100	14.8(1)	—(5)	24162(4)	597(2)	8763(3)
SUN	29.7(1)	—(5)	32953(4)	525(2)	2195(3)
ILSVRC65	6.37(1)	1042(4)	9574(5)	227(2)	450(3)
Ave. Rank	1.5	4.67	4.33	1.67	2.83

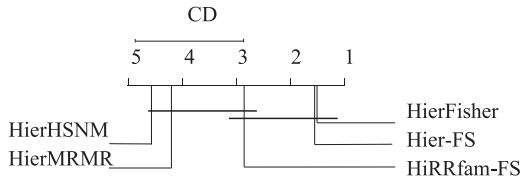

Fig. 10. Running time comparison of Hier-FS algorithm against the others with the Bonferroni-Dunn test.

To further explore whether the running time of the five feature selection algorithms are significantly different, we performed a Friedman test. The null-hypothesis of the Friedman test is that the feature algorithms are equivalent in terms of running time. Under the null-hypothesis, we compute the value $F_F = 30.86$; thus, we reject the null-hypothesis. Thus, the five feature selection algorithms are different in terms of running time. A post-hoc Bonferroni-Dunn test was also conducted.

From Fig. 10, we observe that the running time of Hier-FS is statistically better than those of HierMRMR and HierFSNM. There is no consistent evidence to indicate a statistical difference in the running time between Hier-FS and HierFisher.

4.4.5 Parameter Sensitivity Analysis

We analyze parameter sensitivity using the average hierarchical F_1-measure for different parameters as listed in Table 8. The hierarchical F_1-measure is computed based on 20 percent of selected features on each dataset. A "grid-search" is used to tune the parameters α and β within certain ranges. It is noted that the results reported in Table 8 are achieved using the fixed value $\lambda = 10$. The parameters α and β are chosen from the set $\{0.01, 0.1, 1, 10, 100\}$.

From Table 8, we can observe the followings.

1) For *F194* dataset, the hierarchical F_1-measure is 0.7171 when $\alpha = 0.01$ and $\beta = 0.01$. However, the hierarchical F_1-measure reduces to 0.6329 when $\alpha = 100$ and $\beta = 0.01$. The results demonstrate that there is not a

TABLE 8
Parameter Sensitivity Evaluation on Different Datasets

(a) *F194*

β \ α	0.01	0.1	1	10	100
0.01	0.7171	0.7176	0.7176	0.7176	0.6329
0.1	0.7176	0.7178	0.7176	0.7171	0.6481
1	0.7176	0.7181	0.7176	0.7169	0.6974
10	0.7171	0.7176	0.7176	0.7169	0.7190
100	0.7174	0.7178	0.7178	0.7164	0.7169

(b) *DD*

β \ α	0.01	0.1	1	10	100
0.01	0.8601	0.8601	0.8601	0.8601	0.8601
0.1	0.8601	0.8601	0.8601	0.8601	0.8601
1	0.8601	0.8601	0.8601	0.8601	0.8601
10	0.8601	0.8601	0.8601	0.8601	0.8601
100	0.5851	0.5851	0.5851	0.7731	0.8292

(c) *VOC*

β \ α	0.01	0.1	1	10	100
0.01	0.6752	0.6756	0.6752	0.6757	0.6759
0.1	0.6752	0.6749	0.6750	0.6756	0.6754
1	0.6737	0.6738	0.6740	0.6745	0.6749
10	0.6714	0.6711	0.6705	0.6713	0.6714
100	0.6751	0.6695	0.6721	0.6707	0.6705

(d) *Cifar100*

β \ α	0.01	0.1	1	10	100
0.01	0.7883	0.7885	0.7886	0.7884	0.7881
0.1	0.7883	0.7884	0.7887	0.7883	0.7881
1	0.7880	0.7877	0.7875	0.7880	0.7876
10	0.7881	0.7877	0.7879	0.7876	0.7879
100	0.7883	0.7884	0.7885	0.7883	0.7886

(e) *SUN*

β \ α	0.01	0.1	1	10	100
0.01	0.8403	0.8407	0.8408	0.8414	0.8412
0.1	0.8403	0.8406	0.8408	0.8412	0.8414
1	0.8409	0.8411	0.8412	0.8412	0.8418
10	0.8400	0.8405	0.8406	0.8407	0.8407
100	0.8374	0.8388	0.8387	0.8392	0.8396

(f) *ILSVRC65*

β \ α	0.01	0.1	1	10	100
0.01	0.9590	0.9590	0.9591	0.9591	0.9589
0.1	0.9589	0.9589	0.9589	0.9588	0.9589
1	0.9587	0.9587	0.9588	0.9589	0.9590
10	0.9592	0.9591	0.9592	0.9594	0.9592
100	0.9592	0.9592	0.9592	0.9591	0.9595

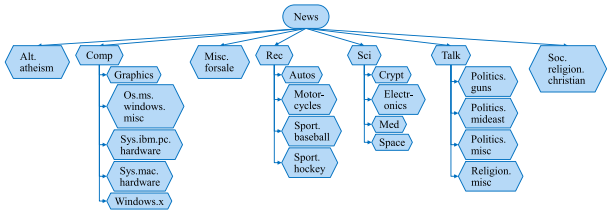

Fig. 11. Hierarchy of the *20 Newsgroup* dataset.

good result when a parent-child relationship is strongly restricted for this dataset.

2) The hierarchical F_1-measure is 0.8601 when $\alpha = 0.01$ and $\beta = 0.01$ for *DD* dataset. However, the hierarchical F_1-measure reduces 0.5851 when $\alpha = 0.01$ and $\beta = 100$. The results demonstrate that too restrictive sibling relationship is not suitable for *DD* dataset. There are similar results for the *SUN* and *VOC* dataset when $\beta = 100$ and $\alpha \in \{0.1, 1, 10, 100\}$.
3) Our method is not sensitive to parameters on *Cifar100* and *ILSVRC65* datasets.

4.4.6 Results on Text Dataset

To illustrate the advantage of the proposed method in dealing with large-scale features, we test the proposed method on *20 Newsgroups* dataset which contains 26,214 features [52]. The *20 Newsgroup* dataset was collected and originally used for document classification by Lang [51]. This dataset includes 18,446 messages collected from 20 different Netnews newsgroups. One thousand messages from each of the 20 newsgroups were chosen at random and partitioned by newsgroup name. The hierarchical structure of the class is shown in Fig. 11. We use the "bydate" version, which contains 951 documents evenly distributed across 20 classes.

We experimentally compare Hier-FS algorithm, Hier-Fisher and HierFSNM, and demonstrate that our algorithm significantly outperforms other algorithms. The hierarchical F_1-measure results are shown in Fig. 12, we observe the followings.

1) The experiments show that Hier-FS algorithm can dramatically outperform the other two algorithms. The method presented in this paper shows good classification ability especially when the number of features is less than 1,000. The less the number of features is, the more obvious the advantage is, which shows that the preferential features have a good ability of classification discrimination.
2) The hierarchical F_1-measure result of Hier-FS when it selects 300 features exceeds that of all features.

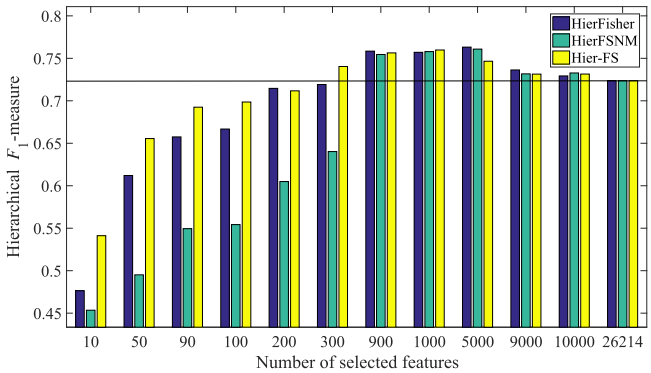

Fig. 12. Performance comparison of Hier-FS algorithm against the others on the *20 Newsgroup* dataset.

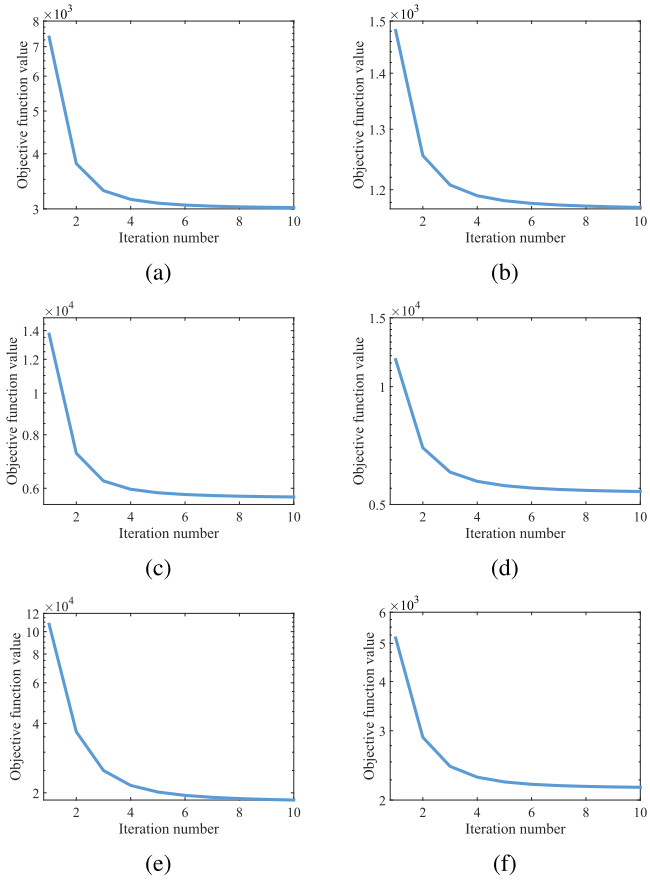

Fig. 13. Convergence curves of the objective function value. (a) *F194*; (b) *DD*; (c) *VOC*; (d) *Cifar100*; (e) *SUN*; and (f) *ILSVRC65*.

4.4.7 Convergence Analysis for HiRRfam-FS

We study the convergence of the proposed HiRRfam-FS algorithm presented in Algorithm 4. Fig. 13 shows the convergence curves based on the objective function value in Equation (20) for all datasets. In our experiments, we set the maximal iteration number $T = 10$ for all datasets. This figure demonstrates that the objective function value decreases monotonically and converges within no more than ten iterations for all datasets.

5 Conclusions and Future Work

We have proposed a hierarchical feature selection framework based on recursive regularization for hierarchical classification. We have exploited parent-child, sibling, and family relationships in hierarchical class structures to optimize our model. The sibling relationship performs poorly for some datasets. However, in conjunction with the parent-child relationship, the sibling relationship can improve effectiveness. In contrast to existing feature selection approaches, we take advantage of the hierarchical class structure, which provides significant information for classification learning. We have also provided efficient Hier-FS, HiRRpar-FS, HiRRsib-FS, and HiRRfam-FS algorithms to select different feature

subsets for each node in a hierarchical tree structure. Compared with the other three hierarchical feature selection approaches, our algorithms achieve competitive results in terms of hierarchical classification accuracy.

The current implementation of the algorithm only deals with a tree structure for class labels in which each node (class) has a single parent. In the future, we will design feature selection approaches for general graph structures where a node can have multiple parents. We use classical least-squares linear regression as loss function in this paper. In the future, we will try other no-linear regression to design the optimization objective function.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under Grant Nos. 61703196, 61432011, 61925602, 61732011, and 91746107, the Natural Science Foundation of Fujian Province under Grant No. 2018J01549, and the President's Fund of Minnan Normal University under Grant No. KJ19021.

References

[1] I. Partalas, I. Partalas, E. Gaussier, and M. R. Amini, "Learning taxonomy adaptation in large-scale classification," *J. Mach. Learn. Res.*, vol. 17, no. 1, pp. 3350–3386, 2016.

[2] J. Deng, W. Dong, R. Socher, L. J. Li, K. Li, and L. Fei Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2009, pp. 248–255.

[3] S. Bengio, J. Weston, and D. Grangier, "Label embedding trees for large multi-class tasks," in *Proc. 23rd Int. Conf. Neural Inf. Process. Syst.*, 2010, pp. 163–171.

[4] C. N. Silla and A. A. Freitas, "A survey of hierarchical classification across different application domains," *Data Mining Knowl. Discovery*, vol. 22, no. 1/2, pp. 31–72, 2011.

[5] Z. Q. Yuan, C. S. Xu, J. T. Sang, S. C. Yan, and M. S. Hossain, "Learning feature hierarchies: A layer-wise tag-embedded approach," *IEEE Trans. Multimedia*, vol. 17, no. 6, pp. 816–827, Jun. 2015.

[6] Q. Y. Wu, M. K. Tan, H. J. Song, J. Chen, and M. K. Ng, "ML-FOREST: A multi-label tree ensemble method for multi-label classification," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 10, pp. 2665–2680, Oct. 2016.

[7] A. Esuli, T. Fagni, and F. Sebastiani, "Boosting multi-label hierarchical text categorization," *Inf. Retrieval*, vol. 11, no. 4, pp. 287–313, 2008.

[8] G. Griffin and P. Perona, "Learning and using taxonomies for fast visual categorization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2008, pp. 1–8.

[9] Y. J. Chang, N. Kim, Y. Lee, J. Lim, J. B. Seo, and Y. K. Lee, "Fast and efficient lung disease classification using hierarchical one-against-all support vector machine and cost-sensitive feature selection," *Comput. Biol. Medicine*, vol. 42, no. 12, pp. 1157–1164, 2012.

[10] H. X. Wang, X. T. Shen, and W. Pan, "Large margin hierarchical classification with mutually exclusive class membership," *J. Mach. Learn. Res.*, vol. 12, pp. 2721–2748, 2011.

[11] S. Gopal and Y. M. Yang, "Hierarchical Bayesian inference and recursive regularization for large-scale classification," *ACM Trans. Knowl. Discovery Data*, vol. 9, no. 3, 2015, p. 18.

[12] W. Zhu, "Relationship among basic concepts in covering-based rough sets," *Inf. Sci.*, vol. 179, no. 14, pp. 2478–2486, 2009.

[13] B. Tang, S. Kay, and H. B. He, "Toward optimal feature selection in naive bayes for text categorization," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 9, pp. 2508–2521, Sep. 2016.

[14] Y. Yang, H. T. Shen, Z. G. Ma, Z. Huang, and X. F. Zhou, "$\ell_{2,1}$-norm regularized discriminative feature selection for unsupervised learning," in *Proc. 22nd Int. Joint Conf. Artif. Intell.*, 2011, pp. 1589–1594.

[15] H. S. Wang *et al.*, "Incremental subgraph feature selection for graph classification," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 1, pp. 128–142, Jan. 2017.

[16] B. Zhao, F. F. Li, and E. P. Xing, "Large-scale category structure aware image categorization," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2011, pp. 1251–1259.

[17] C. Freeman, D. Kulić, and O. Basir, "Feature-selected tree-based classification," *IEEE Trans. Cybern.*, vol. 43, no. 6, pp. 1990–2004, Dec. 2013.

[18] C. Freeman, D. Kulić, and O. Basir, "Joint feature selection and hierarchical classifier design," in *Proc. Int. Conf. Syst. Man Cybern.*, 2011, pp. 1728–1734.

[19] J. Song, P. Zhang, S. Qin, and J. Gong, "A method of the feature selection in hierarchical text classification based on the category discrimination and position information," in *Proc. Int. Conf. Ind. Inform.-Comput. Technol. Intell. Technol. Ind. Inf. Integration*, 2015, pp. 132–135.

[20] A. Gretton, O. Bousquet, A. Smola, and B. Schölkopf, "Measuring statistical dependence with hilbert-schmidt norms," in *Proc. Int. Conf. Algorithmic Learn. Theory*, 2005, pp. 63–77.

[21] H. Zhao, P. Zhu, P. Wang, and Q. Hu, "Hierarchical feature selection with recursive regularization," in *Proc. 26th Int. Joint Conf. Artif. Intell.*, 2017, pp. 3483–3489.

[22] J. H. Dai, Q. H. Hu, J. H. Zhang, H. Hu, and N. G. Zheng, "Attribute selection for partially labeled categorical data by rough set approach," *IEEE Trans. Cybern.*, vol. 47, no. 9, pp. 2460–2471, Sep. 2017.

[23] Q. H. Hu, L. J. Zhang, Y. C. Zhou, and W. Pedrycz, "Large-scale multi-modality attribute reduction with multi-kernel fuzzy rough sets," *IEEE Trans. Fuzzy Syst.*, vol. 26, no. 1, pp. 226–238, Feb. 2018.

[24] R. Tibshirani, "Regression shrinkage and selection via the lasso," *J. Roy. Statist. Soc.. Series B, Methodological*, vol. 58, pp. 267–288, 1996.

[25] G. Obozinski, B. Taskar, and M. I. Jordan, "Joint covariate selection and joint subspace selection for multiple classification problems," *Statist. Comput.*, vol. 20, no. 2, pp. 231–252, 2010.

[26] J. Liu and J. P. Ye, "Moreau-yosida regularization for grouped tree structure learning," in *Proc. 23rd Int. Conf. Neural Inf. Process. Syst.*, 2010, pp. 1459–1467.

[27] A. Argyriou, T. Evgeniou, and M. Pontil, "Multi-task feature learning," in *Proc. 20th Annu. Conf. Neural Inf. Process. Syst.*, 2006, pp. 41–48.

[28] A. Kosmopoulos, I. Partalas, E. Gaussier, G. Paliouras, and I. Androutsopoulos, "Evaluation measures for hierarchical classification: A unified view and novel approaches," *Data Mining Knowl. Discovery*, vol. 29, no. 3, pp. 820–865, 2015.

[29] D. Koller and M. Sahami, "Hierarchically classifying documents using very few words," in *Proc. 14th Int. Conf. Mach. Learn.*, 1997, pp. 170–178.

[30] A. X. Sun and E. P. Lim, "Hierarchical text classification and evaluation," in *Proc. IEEE Int. Conf. Data Mining*, 2001, pp. 521–528.

[31] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (VOC) challenge," *Int. J. Comput. Vision*, vol. 88, no. 2, pp. 303–338, 2010.

[32] F. R. Bach and M. I. Jordan, "Kernel independent component analysis," *J. Mach. Learn. Res.*, vol. 3, no. Jul, pp. 1–48, 2002.

[33] L. Wei, M. Liao, X. Gao, and Q. Zou, "An improved protein structural classes prediction method by incorporating both sequence and structure information," *IEEE Trans. Nanobiosci.*, vol. 14, no. 4, pp. 339–349, Jun. 2015.

[34] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *Comput. Sci.*, 2014, *arXiv:1409.1556*.

[35] D. P. Li, Y. Ju, and Q. Zou, "Protein folds prediction with hierarchical structured SVM," *Current Proteomics*, vol. 13, no. 2, pp. 79–85, 2016.

[36] C. H. Ding and I. Dubchak, "Multi-class protein fold recognition using support vector machines and neural networks," *Bioinformatics*, vol. 17, no. 4, pp. 349–358, 2001.

[37] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," University of Toronto, Toronto, ON, 2009.

[38] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba, "Sun database: Large-scale scene recognition from abbey to zoo," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2010, pp. 3485–3492.

[39] J. Xiao, K. A. Ehinger, J. Hays, A. Torralba, and A. Oliva, "Sun database: Exploring a large collection of scene categories," *Int. J. Comput. Vision*, vol. 119, no. 1, pp. 3–22, 2016.

[40] J. Deng, J. Krause, A. C. Berg, and L. Fei-Fei, "Hedging your bets: Optimizing accuracy-specificity trade-offs in large scale visual recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2012, pp. 3450–3457.

[41] O. Dekel, J. Keshet, and Y. Singer, "Large margin hierarchical classification," in *Proc. Int. Conf. Mach. Learn.*, 2004, vol. 12, Art. no. 27.
[42] J. Struyf, S. Džeroski, H. Blockeel, and A. Clare, *Hierarchical Multi-Classification with Predictive Clustering Trees in Functional Genomics*. Berlin, Germany: Springer, 2005.
[43] L. Cai and T. Hofmann, "Exploiting known taxonomies in learning overlapping concepts," in *Proc. Int. Joint Conf. Artif. Intell.*, 2007, pp. 714–719.
[44] L. Grimaudo, M. Mellia, and E. Baralis, "Hierarchical learning for fine grained internet traffic classification," in *Proc. Int. Wireless Commun. Mobile Comput. Conf.*, 2012, pp. 463–468.
[45] H. C. Peng, F. H. Long, and C. Ding, "Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 8, pp. 1226–1238, Aug. 2005.
[46] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*. Hoboken, NJ, USA: Wiley, 2012.
[47] F. P. Nie, H. Huang, X. Cai, and C. H. Ding, "Efficient and robust feature selection via joint $\ell_{2,1}$-norms minimization," in *Proc. 23rd Int. Conf. Neural Inf. Process. Syst.*, 2010, pp. 1813–1821.
[48] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *J. Mach. Learn. Res.*, vol. 7, no. Jan, pp. 1–30, 2006.
[49] M. Friedman, "A comparison of alternative tests of significance for the problem of m rankings," *The Ann. Math. Statist.*, vol. 11, no. 1, pp. 86–92, 1940.
[50] O. J. Dunn, "Multiple comparisons among means," *J. Amer. Statist. Assoc.*, vol. 56, no. 293, pp. 52–64, 1961.
[51] K. Lang, "NewsWeeder: Learning to filter netnews," in *Proc. 12th Int. Conf. Mach. Learn.*, 1995, pp. 331–339.
[52] D. Cai, X. F. He, W. V. Zhang, and J. W. Han, "Regularized locality preserving indexing via spectral regression," in *Proc. 16th ACM Conf. Inf. Knowl. Manage.*, 2007, pp. 741–750.

Hong Zhao received the MS degree from Liaoning Normal University, Dalian, China, in 2006, and the PhD degree from Tianjin University, Tianjin, China, in 2019. She is currently a professor of the School of Computer Science and the Fujian Key Laboratory of Granular Computing and Application, Minnan Normal University, Zhangzhou, China. She has authored more than 40 journal and conference papers in the areas of granular computing based machine learning and cost-sensitive learning. Her current research interests include rough sets, granular computing, and data mining for hierarchical classification.

Qinghua Hu received the BS, MS, and PhD degrees from the Harbin Institute of Technology, Harbin, China, in 1999, 2002, and 2008, respectively. He was a postdoctoral fellow with the Department of Computing, Hong Kong Polytechnic University from 2009 to 2011. He is currently the dean of the School of Artificial Intelligence, the vice chairman of the Tianjin Branch of China Computer Federation, the vice director of the SIG Granular Computing and Knowledge Discovery, and the Chinese Association of Artificial Intelligence. He is currently supported by the Key Program, National Natural Science Foundation of China. He has published more than 200 peer-reviewed papers. His current research is focused on uncertainty modeling in big data, machine learning with multi-modality data, and intelligent unmanned systems. He is an associate editor of the *IEEE Transactions on Fuzzy Systems*, the *Acta Automatica Sinica*, and the *Energies*. He is a senior member of the IEEE.

Pengfei Zhu received the BS and MS degrees from the Harbin Institute of Technology, Harbin, China, in 2009 and 2011, respectively, and the PhD degree from The Hong Kong Polytechnic University, Hong Kong, China, in 2015. He is currently an associate professor with the School of Computer Science and Technology, Tianjin University. His research interests are focused on machine learning and computer vision.

Yu Wang received the BE and ME degrees from Tianjin University, Tianjin, China, in 2013 and 2016, respectively. He is currently working toward the PhD degree in the School of Computer Science and Technology, Tianjin University. His current research interests include hierarchical classification and hierarchical clustering in machine learning with big data and massive categories.

Ping Wang received the BS, MS, and PhD degrees from Tianjin University, Tianjin, China, in 1988, 1991, and 1998, respectively. She is currently a professor with the School of Mathematics, Tianjin University. She is also a PhD Supervisor with the School of Computer Software, Tianjin University. Her current research interests include pattern recognition and image processing.

▷ **For more information on this or any other computing topic, please visit our Digital Library at** www.computer.org/csdl.