PHFS: Progressive Hierarchical Feature Selection Based on Adaptive Sample Weighting

Hong Zhao, *Member, IEEE*, Jie Shi, and Yang Zhang

***Abstract*— Hierarchical feature selection is considered an effective technique to reduce the dimensionality of data with complex hierarchical label structures. Incorrect labels are a common and challenging issue in complex hierarchical data. However, the existing hierarchical methods often struggle to dynamically adapt to label noise and lack the flexibility to adjust sample weights. Therefore, their effectiveness in managing complex data with many classes and mitigating label noise is significantly limited. To address these issues, in this article, an adaptive sample weighting-based progressive hierarchical feature selection (PHFS) method was proposed, which dynamically adjusts the sample weights to focus on high-quality data. PHFS integrates progressive sample selection and hierarchical feature selection into a unified framework, thus enhancing its effectiveness in reducing the impact of label noise and achieving optimal performance. The progressive selection process is divided into initial and subsequent stages, focusing on correct and incorrect samples. In the initial stage, PHFS selects valuable and correct samples based on the adaptive weights calculated through hierarchical classification feedback, maximizing the guiding effect of the correctly labeled examples. In the subsequent stages, PHFS uses matrix factorization to preserve the structure of the correctly labeled samples, preventing the forgetting of the early selected samples and minimizing the negative impact of the mislabelled samples. The superiority of PHFS over 13 state-of-the-art methods was demonstrated by performing extensive experiments on eight real-world datasets, highlighting its effectiveness in reducing label noise and achieving optimal performance.**

***Index Terms*— Adaptive sample weighting, hierarchical feature selection, matrix factorization, sample selection.**

I. Introduction

With the rapid development of data generation and collection, the complexity of classification tasks is exploding, which is reflected in the dimensions of the features and classes [1]. The high-dimensional features of classification induce high computational and spatial costs to data processing, which hinders the generalization performance of machine learning [2]. Feature selection is proposed to select features relevant to classification from high-dimensional feature spaces, which is an important preprocessing tool in machine learning to reduce the computational and storage requirements [3]. Meanwhile, each class requires different combinations of discriminative features. As a result, features that are effective for certain classes are ineffective for most other classes. The traditional methods cannot select a shared feature subset for all classes simultaneously as the number of classes increases. Inspired by [4], humans can easily process amounts of classes, summarize abstract knowledge hierarchically to learn object concepts, and organize this knowledge into hierarchical structures. Therefore, feature selection utilizes hierarchical class structures to divide a complex classification into a set of related simple subclassifications, significantly reducing the need for one-time recognition of features for each class.

In recent years, hierarchical feature selection has been widely focused on hierarchical management to simplify classification. The majority of the existing methods [5], [6] utilize hierarchical structures generated by semantic relationships to provide additional information and optimize dependencies between the classes in the classification process. These methods require manual intervention during the feature selection process, making it difficult to ensure that the final selected feature is optimal and fully utilizes all information in the data. The matrix decomposition method automatically extracts the most useful features by directly transforming space from high dimension to low dimension, which reduces subjectivity and complexity and provides comprehensive data description [7], [8]. However, the low-dimensional representations obtained through direct transformation lack interpretability. To address this drawback, Wang et al. [9] combined matrix decomposition and feature selection to improve the interpretability during the process of space decomposition. This basic framework was subsequently improved by many works in the literature to ensure that a few low-dimensional data contain sufficient information and make it more efficient and accurate to discover hidden data relationships with lower complexity [10]. These methods are based on the assumption that all samples in a dataset are correctly labeled, where an accurate class label is assigned to each sample.

Numerous samples in real-world datasets exhibit labeling inaccuracies, often attributable to human error during the labeling and data collection processes including the widely recognized ImageNet dataset. The learned information bias predictions are as important as correctly labeled samples during training if both correctly labeled and noisy samples are treated equally. Illustratively, Fig. 1 shows a training process based on samples with both correct and noisy labels, including an instance where an image intended to be labeled "dingo" is erroneously labeled as "dhole." Equal treatment of all samples can lead to difficulties in the measurer and learning errors [11], diminishing the benefits of accurately

Received 19 July 2024; revised 5 November 2024; accepted 31 December 2024. This work was supported in part by the National Natural Science Foundation of China under Grant 62376114 and in part by the Fujian Province Undergraduate Universities Teaching Reform Research Project under Grant FBJY20240103. *(Corresponding author: Hong Zhao.)*

The authors are with the School of Computer Science and the Key Laboratory of Data Science and Intelligence Application, Minnan Normal University, Zhangzhou, Fujian 363000, China (e-mail: hongzhaocn@163.com; shijie_jay@126.com; zhang_yangcn@163.com).

Digital Object Identifier 10.1109/TNNLS.2025.3525643

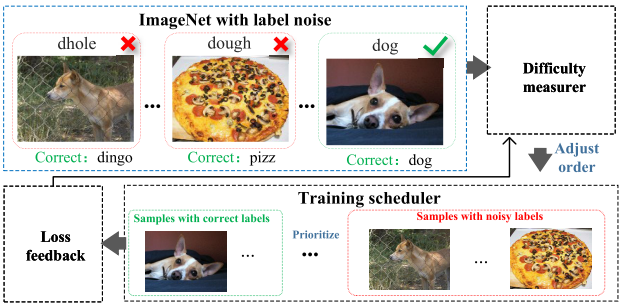

Fig. 1. Training process based on samples with both correct and noisy labels.

labeled samples and exacerbating the negative effects of label noise.

Handling label noise in machine learning is a critical challenge, as noisy labels can significantly degrade the performance of the learning models. To address this issue, recent research has focused on two main approaches: graph-based regularization and loss functions, as well as adversarial and contrastive learning methods. These approaches aim to improve the robustness of the models by leveraging the graph structure and advanced training techniques. Several works have explored the use of graph-based regularization and loss functions to handle label noise. For example, Lerner and Meir [12] proposed a graph-based approach with a regularization term to correct noisy labels using the graph structure. In addition to graph-based regularization, adversarial and contrastive learning methods have shown promise in improving robustness against label noise. For instance, You et al. [13] used contrastive learning with data augmentation to improve the robustness of the graph models to label noise.

Feature selection based on sample selection strategy assigns different weights to samples to mitigate the impact of the label errors. According to [14], these algorithms can be divided into progressive selection strategies based on the fixed sample order [15], [16], [17] and self-adjusting sample order [18], [19], [20]. Although the progressive sample selection strategies are effective in handling label errors, they may still overemphasize incorrectly labeled samples. The integration of feedback mechanisms that allow for adaptive selection could enhance the robustness of these methods.

However, the existing methods often struggle to dynamically adapt to the evolving nature of the label noise. They typically rely on predefined criteria or fixed sample orders, lacking the flexibility to dynamically adjust. In addition, hierarchical feature selection is not incorporated in these methods, which is essential for managing complex datasets with many classes. Moreover, the lack of a comprehensive framework that integrates both sample selection and feature selection limits their effectiveness in mitigating label noise and achieving optimal performance.

To effectively address these limitations, we propose a progressive hierarchical feature selection (PHFS) method based on adaptive sample weighting. PHFS adaptively adjusts sample weights and leverages hierarchical feature selection to improve both robustness and performance. PHFS handles samples with correct and incorrect labels in hierarchical classification using initial and subsequent stages. In the initial stage of PHFS, progressive sample selection strategies are adopted by evaluating the value of the samples to give higher weights to those samples with correct labels that are useful for the current classification. In the sample selection process, samples with relatively higher weights are gradually selected preferentially until all samples are selected, which maximizes the guiding effect of the correctly labeled examples on classification. In the subsequent stage of PHFS, matrix decomposition ensures that valuable information is retained in the labeled samples of each subtask in the hierarchical classification task. Matrix decomposition effectively prevents forgetting early selected samples and minimizes the negative impact of the samples with incorrect labels.

The main contributions can be described as follows.

1) A PHFS framework based on self-paced learning was proposed. This framework prioritizes correctly labeled samples that are particularly relevant to the current subclassification task, progressively selecting and emphasizing them to focus on high-quality data.
2) An optimization algorithm that combines seven sample selection strategies and matrix factorization techniques was successfully integrated. The training scheduler dynamically adjusts sample weights based on difficulty and correctness, allowing the model to gradually focus on more challenging and potentially mislabelled samples. Matrix factorization protects valuable information and reduces the impact of incorrect samples in each subtask.
3) Extensive additional experiments were performed on eight real-world datasets to demonstrate the superiority of the proposed method compared with 13 state-of-the-art methods.

The remainder of this work is organized as follows. Section II elaborates on the related works. Section III describes the framework. Section IV illustrates the optimization objectives. Section V explains and analyzes the experimental results. Section VI concludes and proposes further work.

II. Related Research and Challenges

The related work is divided into methods using equal treatment sample strategies, methods using progressive sample selection strategies, and challenges.

A. Methods Using Equal Treatment Sample Strategies

The traditional feature selection method treats all samples equally important without considering the correctness of labels and has received widespread attention due to its relative simplicity and ease of implementation. Many existing methods utilize hierarchical information to handle complex class relationships in classification tasks. Zhao et al. [4] first explored the sibling and parent–child relationships to select and categorize features by imposing constraints on the independence and correlation between the class labels. Based on this, Zheng et al. [5] used label information-based measurement techniques to expand the interclass distance and solved

the problem of approximate classes that were difficult to distinguish in hierarchical classification. Subsequently, Shi et al. [6] utilized similar class constraints to expand interclass independence while reducing intraclass redundancy through feature correlation constraints. However, these methods cannot capture the complex relationships in high-dimensional data space, thus interfering with the extraction of structural information and failing to fully mine data structure information.

With the application of the feature selection method based on nonnegative matrix decomposition, the correlation and importance between different features are automatically learned to obtain more compact and interpretable feature subspaces. Wang et al. [9] first introduced matrix factorization into feature selection, which decomposes the original space into two low-dimensional subspaces, namely, an indicator and coefficient subspace. This matrix decomposition preserves important information while reducing the data dimension and realizes efficient reconstruction of the original spatial information. This basic framework provides a research foundation for many subsequent works. Pujahari and Sisodia [21] restored the global feature information by applying high-sparsity and low-redundancy constraints on the indicator subspace. Song and Song [10] combined low-rank, sparse, and self-representation strategies to preserve potential data in the indicator spaces. These methods rely on the information of the indicator subspace, which contains some linear combinations of original features. However, the coefficient subspace, which is not taken into account by the abovementioned methods, reflects the weight of these linear combinations and plays an important role in evaluating the low-dimensional index space and analyzing the structural information of the original data [7].

Inspired by this effect, PHFS integrates the information of two low-dimensional subspaces based on a hierarchical structure, which provides a stronger performance advantage when dealing with classification tasks by deeply exploring and retaining structural space information.

B. Methods Using Progressive Sample Selection Strategies

Feature selection based on sample selection strategies assigns different weights to different samples to reduce the impact of label errors on classification performance. According to [14], these methods can be divided into progressive selection strategies based on fixed and self-adjusting sample order. Curriculum learning, as a representative of fixed selection strategies, has become a research hotspot due to its denoising and guidance functions [22]. Yang et al. [15] improved the parallel convolution model based on the feature selection module, where the curriculum learning loss achieves denoising by dynamically adjusting the difficult sample weights. Xia et al. [16] designed a feature embedding block based on the implementation of curriculum learning strategies to achieve the guiding role of the reward sparsity problem. Sun and Wang [17] improved the instance-level feature selection by combining curriculum learning with iterative operations to guide training from simpler to more difficult tasks. These methods gradually select more complex samples in a data-driven manner and ignore real-time feedback on the algorithm state during this process.

Sample selection combined with self-learning adaptively selects complex samples gradually. Kumar et al. [18] first introduced self-paced learning, and subsequent works have designed various self-paced regularization constraints within this framework. For example, Li et al. [23] designed a hard regularization to consider the cost of feature acquisition and misclassification. Moreover, Wang et al. [19] proposed a self-paced strategy based on hard regularization processes of the difficult samples. Hard regularization fails to distinguish two samples with different importance. Therefore, Miao et al. [24] proposed a soft regularization. Inspired by this, Wan et al. [20] designed a self-paced framework based on linear soft regularization using label similarity-based propagation. However, Li et al. [25] proved that linear regularization distinguishes the sample importance and fails to tolerate subtle errors. Various mixed soft regularization terms have been proposed to avoid excessive sample differentiation while maintaining the flexibility of sample differentiation [26]. Therefore, the above self-paced learning-based strategies were applied to perform hierarchical feature selection and incorporate feedback information more flexibly.

C. Corresponding Challenges

Although much effort has been dedicated to feature selection, the noted methods suffer from the following limitations and challenges.

1) Dealing with incorrect labels in machine learning becomes increasingly complex, leading to classification errors due to the presence of inaccurately labeled samples. Therefore, equal treatment of all samples disrupts the classification process. To address this issue, an adaptive weighting approach prioritizes samples with correct labels during the selection process, reducing the impact of those with incorrect labels. This shift toward more reliable samples in the hierarchical classification process can enhance classification accuracy.
2) The methods using progressive strategies overemphasize samples with incorrect labels, neglecting the samples with correct labels in the later selection stages. To address this issue, matrix factorization for each subtask preserves the structure of samples with correct labels. The focus shifts toward samples with correct labels, preventing misguidance by incorrectly labeled ones in the later selection process, which flexibly utilizes samples with correct labels for each subtask requirement.

III. Proposed Method

This section first describes the framework of PHFS. Sample selection was also proposed using progressive strategies and feature selection using matrix factorization. Finally, more details on PHFS are provided.

A. Framework of PHFS

The framework of PHFS is shown in Fig. 2, which can be mainly divided into the following two parts.

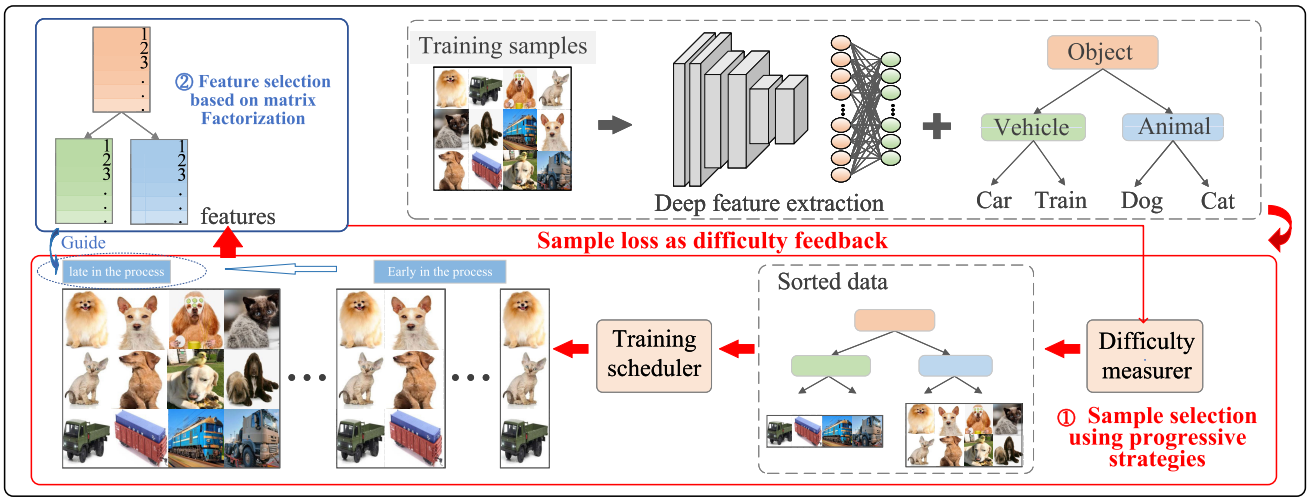

Fig. 2. Framework for PHFS.

1) *Sample Selection Using Progressive Strategy:* This framework consists of the difficulty measurer and the training scheduler.
   a) *Difficulty Measurer:* This step identifies and selects simple samples, distinguishing them from difficult ones. The identified simple samples are then sent to the training scheduler.
   b) *Training Scheduler:* During the training phase, the training scheduler sorts the samples according to their difficulty. As the training progresses, the model learns from and overcomes more complex samples.
2) *Feature Selection Using Matrix Factorization:* Each subtask of hierarchical classification employs matrix factorization, ensuring continued utilization of information from correct samples, which leverages information from samples with correct labels while mitigating the negative influence of mislabelled samples.

B. Sample Selection Using Progressive Strategies

Advanced data collection technologies create complexity in classification tasks involving numerous classes and features. This complexity arises due to redundant features, which degrade the performance in datasets with a large number of classes [27]. To address this issue, feature selection has been proposed to select classification-related features and improve classification accuracy [28]. The traditional methods process all classes simultaneously, which becomes infeasible as the class number grows, leading to resource-intensive computations and compromising real-time performance. Hierarchical feature selection addresses the challenge of managing data with thousands of classes by organizing objects into multigranularity abstract knowledge. It leverages hierarchical structures to divide a complex global task (classifying all classes simultaneously) into easily classified subtasks. Coarse-grained tasks group similar classes for easier differentiation, while fine-grained tasks handle subtle class differences. Hierarchical feature selection is mathematically expressed as follows:

$$J = \sum_{i=0}^{N} \left\| \mathbf{X}_i \mathbf{W}_i^l - \mathbf{Y}_i \right\|_F^2 + R(\cdot) \tag{1}$$

where $N+1$ is the nonleaf node number. Matrixes $\mathbf{Y}_i = [\mathbf{y}_1^i; \ldots; \mathbf{y}_j^i; \ldots; \mathbf{y}_{n_i}^i] \in \mathbf{R}^{n_i \times m}$, $\mathbf{X}_i = [\mathbf{x}_1^i; \ldots; \mathbf{x}_j^i; \ldots; \mathbf{x}_{n_i}^i] \in \mathbf{R}^{n_i \times d}$, and $\mathbf{W}_i^l \in \mathbf{R}^{d \times m}$ are the label, data, and weight matrixes of the ith nonleaf node, where n_i is the sample number in subtree of ith nonleaf node (ith subtask), m is the maximum class number in all subtrees, and d is the feature number. $\mathbf{X}_0$ is global data, which is a data matrix containing all samples in the dataset. $\mathbf{X}_i$ is a data matrix containing samples of the ith subtask. The loss term $||\mathbf{X}_i \mathbf{W}_i^l - \mathbf{Y}_i||_F^2$ measures the error between the predicted and true labels. The regularization term $R(\cdot)$ limits the algorithm complexity and prevents overfitting.

Hierarchical methods assume that all labels are correct when classifying samples. However, real-world datasets frequently contain errors stemming from manual labeling or data complexity. To address label bias, strategies such as human learning gradually integrate samples into the training process to identify and select incorrect patterns or features. One commonly used strategy is self-paced selection based on sample weighting, where sample weight is determined by sample loss and gradually decreases. This prioritizes samples with smaller loss values, indicating high-confidence samples with correct labels. The mathematical expression of hierarchical feature selection based on sample weighting strategies is as follows:

$$J = \sum_{i=0}^{N} \sum_{j=1}^{n_i} p_j^i l_j^i + \mathrm{SP}(\cdot) + R(\cdot) \tag{2}$$

where p_j^i means the sample weight and $\mathbf{P}_i = [p_j^i] \in \mathbf{R}^{1 \times n_i}$. Loss $l_j^i = ||\mathbf{x}_j^i \mathbf{W}_i \mathbf{H}_i - \mathbf{y}_j^i||_F^2$ measures the error between each sample's real and predicted label. The sample selection regularization term $\mathrm{SP}(\cdot)$ controls the increasing step of the selected sample size.

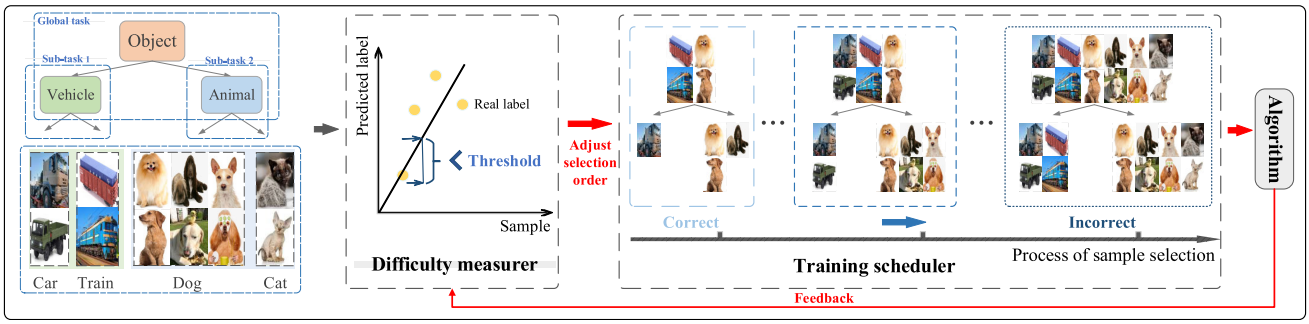

Fig. 3. Flowchart of the sample selection process using progressive strategies. The difficulty in measurer determines the prediction difficulty by the sample error between the predicted and the true label. The training scheduler sorts the training samples from easy to difficult as the training progresses.

TABLE I
CORRESPONDING EXPRESSIONS AND CLOSED-FORM SOLUTIONS OF SP-REGULARIZERS

SP-regularizer	Corresponding Expression	Close-formed Solution
Hard [29]	$-\lambda_1 \sum_{i=1}^{n_i} p_j^i$	$\begin{cases} 1, & \text{if } l_j^i < \lambda_1 \\ 0, & \text{otherwise} \end{cases}$
Linear [30]	$\frac{1}{2}\lambda_1 \sum_{i=1}^{n_i}((p_j^i)^2 - 2p_j^i)$	$\begin{cases} 1-\frac{l_j^i}{\lambda_1}, & \text{if } l_j^i < \lambda_1 \\ 0, & \text{otherwise} \end{cases}$
Logarithmic [31]	$\sum_{i=1}^{n_i}\left(\zeta p_j^i - \frac{\zeta^{p_j^i}}{\log\zeta}\right) \quad \zeta = 1-\lambda_1, 0<\lambda_1<1$	$\begin{cases} \frac{\log(l_j^i+\zeta)}{\log\zeta}, & \text{if } l_j^i < \lambda_1 \\ 0, & \text{otherwise} \end{cases}$
Logistic [32]	$\sum_{i=1}^{n_i} \ln(v_i^j)^{v_i^j} + \ln(p_j^i)^{p_j^i} - \lambda_1 p_j^i$ $v_i^j = 1 - e^{-\lambda_1} - p_j^i \quad \lambda_1 > 0$	$\frac{1+e^{-\lambda_1}}{1+e^{l_j^i-\lambda_1}}$
Mixture 1 [33]	$-\zeta \sum_{i=1}^{n_i} \log\left(p_j^i + \frac{\zeta}{\lambda_1}\right) \quad \zeta = \frac{\lambda_1\lambda_2}{\lambda_1-\lambda_2}, \lambda_1 > \lambda_2 > 0$	$\begin{cases} 1, & \text{if } l_j^i \le \lambda_2 \\ 0, & \text{if } l_j^i \ge \lambda_1 \\ \zeta\left(\frac{1}{l_j^i} - \frac{1}{\lambda_1}\right), & \text{otherwise} \end{cases}$
Mixture 2 [34]	$\frac{\gamma^2}{p_j^i + \frac{\gamma}{\lambda_1}}, \quad \gamma > 0$	$\begin{cases} 1, & \text{if } l_j^i \le (\frac{\lambda_1\gamma}{\lambda_1+\gamma})^2 \\ 0, & \text{if } l_j^i \ge \lambda_1^2 \\ \gamma\left(\frac{1}{\sqrt{l_j^i}} - \frac{1}{\lambda_1}\right), & \text{otherwise} \end{cases}$
Mixture 3 [35]	$\sum_{i=1}^{n_i}(\frac{\lambda_1-\lambda_2}{t} p_j^i - \lambda_1 p_j^i) \quad \lambda_1 > \lambda_2, t > 0$	$\begin{cases} 1, & \text{if } l_j^i \le \lambda_2 \\ (\frac{\lambda_1 - l_j^i}{\lambda_1-\lambda_2})^{\frac{1}{t-1}}, & \text{if } \lambda_2 \le l_j^i \le \lambda_1 \\ 0, & \text{if } l_j^i \ge \lambda_1 \end{cases}$

The sample selection strategies select samples suitable for the current subtask more finely according to the different needs of the subtask. The process is depicted in Fig. 3, which consists of two main steps: the difficulty measurer and the training scheduler. The training scheduler adjusts the order in which samples are selected by dynamically adjusting the sample weights. The difficulty measurer determines the difficulty based on the algorithm's loss feedback for each sample.

Table I lists seven types of difficulty measurer strategies SP(·), their corresponding expressions, and closed-form solutions designed to address different scenario requirements. These strategies are classified into three categories.

1) *Basic Regularization:* (Hard, Linear) provides fundamental regularization operations suited for scenarios requiring simple models and fast implementation.
2) *Probability Adjustment Regularization:* (Logarithmic, Logistic) incorporates more sophisticated mathematical functions to handle probability values, making it suitable for models needing finer control over probability distributions.
3) *Advanced Mixture Regularization:* (Mixture 1, Mixture 2, Mixture 3) offers the greatest flexibility, allowing for adjustments across multiple dimensions, ideal for advanced applications that require highly customized regularization strategies. We select the appropriate regularization method based on the specific task demands.

These strategies select a portion of samples with correct labels according to the parameters of the current stage, where the parameters are gradually changed as the learning progresses to increase the number of samples with incorrect labels. In this progressive process, samples with sample loss value $l_j^i <$ threshold are classified as samples with correct labels, while samples with $l_j^i \geq$ threshold are considered samples with incorrect labels. These strategies enable more flexible and efficient handling of possible mislabelling when dealing with classification tasks. The negative impact of samples with

incorrect labels grows in the later stages of sample selection as correct samples are preferentially chosen.

C. Feature Selection Using Matrix Factorization

Feature selection based on matrix factorization transforms the high-dimensional space of the original data into a low-dimensional subspace while still being able to effectively express the structure and relationship of the data [9]. Hierarchical classification requires that the subspace through which selected features pass represents the space through which all features pass to be represented as follows:

$$J = \sum_{i=0}^{N} ||\mathbf{X}_i\mathbf{W}_i\mathbf{H}_i - \mathbf{Y}_i||_F^2 + R(\cdot)$$
$$\text{s.t. } \mathbf{W}_i \geq 0, \quad \mathbf{W}_i^T\mathbf{W}_i = \mathbf{E}_k \tag{3}$$

where $\mathbf{E}_k$ is a $k \times k$ unit matrix and k is the subspace number. Matrixes $\mathbf{H}_i = [\mathbf{h}_1^i; \ldots; \mathbf{h}_j^i; \ldots; \mathbf{h}_k^i] \in \mathbf{R}^{k\times m}$ and $\mathbf{W}_i = [\mathbf{w}_1^i; \ldots; \mathbf{w}_j^i; \ldots; \mathbf{w}_{m_i}^i] \in \mathbf{R}^{d\times k}$ represent the coefficient and indicator matrixes of the ith nonleaf node, where $\mathbf{h}_j^i \in \mathbf{R}^{k\times 1}$ and $\mathbf{w}_j^i \in \mathbf{R}^{1\times k}$. Selected feature submatrix $\widetilde{\mathbf{X}}_i$ is represented as $\widetilde{\mathbf{X}}_i = \mathbf{X}_i\mathbf{W}_i = \mathbf{x}_1^i\mathbf{w}_1^i + \cdots + \mathbf{x}_j^i\mathbf{w}_j^i + \cdots + \mathbf{x}_{n_i}^i\mathbf{w}_{n_i}^i$, which is considered as dimensionality reduction. It can be concluded that $\mathbf{X}_i$ is closely related to $\widetilde{\mathbf{X}}_i$, where the corresponding term $\mathbf{x}_j^i\mathbf{w}_j^i$ plays a more effective role in $\widetilde{\mathbf{X}}_i$ if $\mathbf{w}_j^i$ is larger. Therefore, $\mathbf{X}_i$ plays an important role in dimensionality reduction, as explained in Example 1.

Example 1: We assume $\mathbf{X}_1 = [\mathbf{d}_1^1, \mathbf{d}_2^1, \mathbf{d}_3^1, \mathbf{d}_4^1, \mathbf{d}_5^1, \mathbf{d}_6^1]$ as

$$\mathbf{X}_1 = \begin{bmatrix} 0.6 & 0.1 & 0.4 & 0.3 & 1.0 & 0.3 \\ 0.2 & 0.8 & 0.1 & 0.6 & 0.9 & 0.8 \\ 0.5 & 0.4 & 0.2 & 0.3 & 0.7 & 1.0 \end{bmatrix} \in \mathbf{R}^{3\times 6}. \tag{4}$$

We suppose the subspace number $k = 3$ to obtain indictor matrix $\mathbf{W}_1 \in \mathbf{R}^{6\times 3}$ as

$$\mathbf{W}_1 = \begin{bmatrix} \mathbf{w}_1^1 \\ \mathbf{w}_2^1 \\ \mathbf{w}_3^1 \\ \mathbf{w}_4^1 \\ \mathbf{w}_5^1 \\ \mathbf{w}_6^1 \end{bmatrix} = \begin{bmatrix} 0.0 & 0.4 & 0.0 \\ 0.0 & 0.0 & 0.0 \\ 0.9 & 0.0 & 0.1 \\ 0.0 & 0.0 & 0.0 \\ 0.2 & 1.0 & 0.0 \\ 0.0 & 0.0 & 0.0 \end{bmatrix}. \tag{5}$$

It is easy with simple calculations to see that $||\mathbf{w}_2^1||_2 = ||\mathbf{w}_4^1||_2 = ||\mathbf{w}_6^1||_2 = 0$, $||\mathbf{w}_1^1||_2 = 0.4$, $||\mathbf{w}_3^1||_2 = 0.9055$, and $||\mathbf{w}_5^1||_2 = 1.0198$. Therefore, we obtain $\widetilde{\mathbf{X}}_1 = \mathbf{X}_1\mathbf{W}_1 = \mathbf{d}_1^1\mathbf{w}_1^1 + \underbrace{\mathbf{d}_2^1\mathbf{w}_2^1}_{=0} + \mathbf{d}_3^1\mathbf{w}_3^1 + \underbrace{\mathbf{d}_4^1\mathbf{w}_4^1}_{=0} + \mathbf{d}_5^1\mathbf{w}_5^1 + \underbrace{\mathbf{d}_6^1\mathbf{w}_6^1}_{=0} = \mathbf{d}_1^1\mathbf{w}_1^1 + \mathbf{d}_3^1\mathbf{w}_3^1 + \mathbf{d}_5^1\mathbf{w}_5^1$. Therefore, we obtain the selected features $\widetilde{\mathbf{X}}_1 = \{\mathbf{x}_1^1, \mathbf{x}_3^1, \mathbf{x}_5^1\}$ based on indicator matrix $\mathbf{W}_1$, where $||\mathbf{w}_j^1||_2 = 0$ has no affect on feature selection.

The nonnegative condition $\mathbf{W}_i \geq 0$ and the orthogonal condition $\mathbf{W}_i^T\mathbf{W}_i = \mathbf{E}_k$ in (3) cannot guarantee that $\mathbf{W}_i$ is an indicator matrix. Specifically, an indicator matrix satisfies these conditions, while meeting these conditions may not be an indicator matrix. For example, $\mathbf{W}_i = \begin{bmatrix} (\sqrt{3}/2) & 0 & (1/2) \\ 0 & 1 & 0 \\ -(1/2) & 0 & (\sqrt{3}/2) \end{bmatrix}$ satisfies these conditions but is not an index matrix. To strengthen indicator generation capabilities, sparse learning is applied in the establishment of indicator attributes [36], such as the ℓ_2- and ℓ_1-norms. Inspired by [37], we adopt $\sum_{j=1}^{d}\sum_{k=1,k\neq j}^{d} |\langle \mathbf{w}_j^i, \mathbf{w}_k^i\rangle|$ to enhance the ability to construct the index matrix

$$J = \sum_{i=0}^{N}\left(||\mathbf{X}_i\mathbf{W}_i\mathbf{H}_i - \mathbf{Y}_i||_F^2 + \lambda\left(||\mathbf{W}_i\mathbf{W}_i^T||_1 - ||\mathbf{W}_i||_2^2\right)\right)$$
$$\text{s.t. } \mathbf{W}_i \geq 0, \quad \mathbf{W}_i^T\mathbf{W}_i = \mathbf{E}_k \tag{6}$$

which has the function of both ℓ_1- and ℓ_2-norms. Some weight vectors are forced to be zero during the regularization process, thus eliminating irrelevant or useless features and retaining features related to the target variable.

However, (6) ignores the role of coefficient matrix $\mathbf{H}_i$. We use Example 2 to explain the role of matrix $\mathbf{H}_i$.

Example 2: We expand $||\mathbf{X}_i\mathbf{W}_i\mathbf{H}_i - \mathbf{Y}_i||_F^2$ as

$$\begin{cases} \tilde{\mathbf{y}}_1^i \simeq \mathbf{X}_i\mathbf{W}_i\mathbf{h}_1^i = \mathbf{H}_{11}^i\mathbf{f}_{l_1}^i + \cdots + \mathbf{H}_{k1}^i\mathbf{f}_{l_k}^i \\ \tilde{\mathbf{y}}_2^i \simeq \mathbf{X}_i\mathbf{W}_i\mathbf{h}_2^i = \mathbf{H}_{12}^i\mathbf{f}_{l_1}^i + \cdots + \mathbf{H}_{k2}^i\mathbf{f}_{l_k}^i \\ \vdots \\ \tilde{\mathbf{y}}_{n_i}^i \simeq \mathbf{X}_i\mathbf{W}_i\mathbf{h}_{n_i}^i = \mathbf{H}_{1n_i}^i\mathbf{f}_{l_1}^i + \cdots + \mathbf{H}_{kn_i}^i\mathbf{f}_{l_k}^i \end{cases} \tag{7}$$

where $\mathbf{X}_i\mathbf{W}_i = \widetilde{\mathbf{X}}_i = [\widetilde{\mathbf{x}}_j^i]$ and $\tilde{\mathbf{y}}_j^i \in \mathbf{R}^{n_i\times 1}$ represents the sample numbers of the ith nonleaf node.

From Example 2, we observe that: 1) the label set for each class can be represented as a linear combination of selected features and 2) $\tilde{\mathbf{y}}_j^i = \mathbf{H}_{1j}^i\widetilde{\mathbf{x}}_1^i + \mathbf{H}_{2j}^i\widetilde{\mathbf{x}}_2^i + \cdots + \mathbf{H}_{kj}^i\widetilde{\mathbf{x}}_k^i$ indicates that $\mathbf{h}_j^i$ plays an inevitable role in the feature selection process. Matrix $\mathbf{H}_i$ captures correlations between samples and implicitly considers the correlation between features to enhance the algorithm's recognition ability for features. Therefore, we apply the ℓ_1- and ℓ_2-norms in $\mathbf{H}_i$ on (6) to consider highly correlated features as follows:

$$J = \sum_{i=0}^{N}\left(||\mathbf{X}_i\mathbf{W}_i\mathbf{H}_i - \mathbf{Y}_i||_F^2 + \lambda\left(||\mathbf{W}_i\mathbf{W}_i^T||_1 - ||\mathbf{W}_i||_2^2\right)\right.$$
$$\left. + \mu\left(||\mathbf{H}_i\mathbf{H}_i^T||_1 - ||\mathbf{H}_i||_2^2\right)\right)$$
$$\text{s.t. } \mathbf{W}_i, \mathbf{H}_i \geq 0, \quad \mathbf{W}_i^T\mathbf{W}_i = \mathbf{E}_k \tag{8}$$

where the sparse combination of the ℓ_1- and ℓ_2-norms improves the recognition ability for features.

The matrix decomposition process is illustrated in Fig. 4, which includes dimension reduction and feature selection. Dimension reduction is the reduction of dimensions based on the indicator matrix in the features of the original dataset. Feature selection is the prediction label based on the reconstruction and reduced matrix. After matrix factorization, the approach investigates $\mathbf{W}_i$ and $\mathbf{H}_i$ for complete structural data insight.

D. Proposal and Discussion of PHFS

The final objective equation combines feature selection based on matrix factorization and progressive selection strategies based on self-spaced learning for hierarchical classification called PHFS as follows:

$$J = \sum_{i=0}^{N}\left(\sum_{j=1}^{n_i} p_j^i||\mathbf{x}_j^i\mathbf{W}_i\mathbf{H}_i - \mathbf{y}_j^i||_F^2 + \text{SP}(\cdot)\right)$$
$$+ \lambda\left(||\mathbf{W}_i\mathbf{W}_i^T||_1 - ||\mathbf{W}_i||_2^2\right) + \mu\left(||\mathbf{H}_i\mathbf{H}_i^T||_1 - ||\mathbf{H}_i||_2^2\right)$$
$$\text{s.t. } \mathbf{W}_i, \mathbf{H}_i \geq 0, \quad \mathbf{W}_i^T\mathbf{W}_i = \mathbf{E}_k. \tag{9}$$

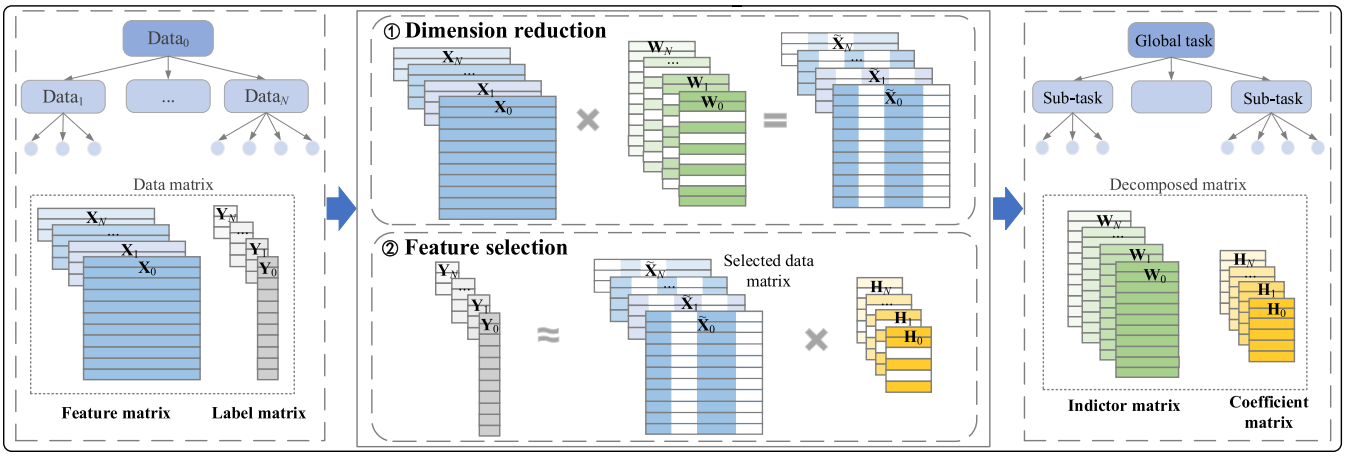

Fig. 4. Flowchart of feature selection using matrix factorization.

PHFS has the following comparative advantages.

1) PHFS dynamically weights samples based on difficulty, transitioning from easy to challenging samples. Progressive selection prioritizes accurate labels for each subtask, which reduces the impact of samples with incorrect labels.
2) PHFS retains correct sample structures via subtask matrix factorization, preventing their loss and ensuring reliable feature selection results in the presence of samples with incorrect labels.

PHFS degenerates into simple algorithms after specifying certain parameters.

1) *Removing the Sample Selection Strategies:* $p_j^i = 1$ and removing SP(·) means that sample selection is not performed and the impact of samples on the algorithm is not distinguished. Consequently, all samples are used for training when the sample selection strategy is removed.
2) *Disabling Matrix Factorization Regularization:* Setting λ and μ to zero removes the regularization penalties associated with matrix factorization.

This simplification reduces the complexity of the algorithm, allowing it to be trained without the need for sample selection and matrix factorization.

IV. Optimization

A. Optimization Algorithm

Each variable in (9) can be treated as convex when the other variables are fixed, which performs individual optimization for each variable as a separate convex optimization problem, given other variable values. Therefore, each variable in (9) is iteratively optimized to find the optimal solution using alternative optimization methods, and the Lagrangian function form of (9) is rewritten as follows:

$$\begin{aligned} J = &\sum_{i=0}^{N}\left(\sum_{j=1}^{n_i} p_j^i ||\mathbf{x}_j^i \mathbf{W}_i \mathbf{H}_i - \mathbf{y}_j^i||_F^2 + \mathrm{SP}(\cdot)\right) \\ &+ \lambda\left(||\mathbf{W}_i \mathbf{W}_i^T||_1 - ||\mathbf{W}_i||_2^2\right) + \mu\left(||\mathbf{H}_i \mathbf{H}_i^T||_1 - ||\mathbf{H}_i||_2^2\right) \\ &+ \frac{\alpha}{2}||\mathbf{W}_i^T \mathbf{W}_i - \mathbf{E}_k||_2 + \mathrm{tr}(\mathbf{a}_i \mathbf{W}_i) + \mathrm{tr}(\mathbf{b}_i \mathbf{H}_i)) \end{aligned} \tag{10}$$

where $\mathbf{a}_i = [a_{jl}^i] \in \mathbf{R}^{d\times k}$ and $\mathbf{b}_i = [b_{jl}^i] \in \mathbf{R}^{k\times m}$ are the Lagrange multipliers. Specifically, the three variable optimizations in (10) are transformed into three subproblems with fixed other two variables. The specific steps for optimization are as follows.

1) *Update* $\mathbf{P}_i$ *With Fixed* $\mathbf{W}_i$ *and* $\mathbf{H}_i$*:* The terms related to $\mathbf{W}_i$ and $\mathbf{H}_i$ are regarded as constants when $\mathbf{W}_i$ and $\mathbf{H}_i$ are fixed. We assume $l_j^i = ||\mathbf{x}_j^i \mathbf{W}_i \mathbf{H}_i - \mathbf{y}_j^i||_2$ to obtain

$$J_P = \sum_{i=0}^{N}\sum_{j=1}^{n_i} p_j^i l_j^i + \mathrm{SP}(\cdot), \quad \text{s.t. } 0 \le p_j^i \le 1. \tag{11}$$

The closed-form solutions of SP(·) are listed in Table I.

2) *Update* $\mathbf{H}_i$ *With Fixed* $\mathbf{W}_i$ *and* $\mathbf{P}_i$*:* The terms related to $\mathbf{W}_i$ and $\mathbf{P}_i$ are regarded as constants when $\mathbf{W}_i$ and $\mathbf{P}_i$ are fixed, and (10) is converted to

$$\begin{aligned} J_H = &\sum_{i=0}^{N}\left(\sum_{j=1}^{n_i} p_j^i ||\mathbf{x}_j^i \mathbf{W}_i \mathbf{H}_i - \mathbf{y}_j^i||_F^2\right) + \mu\left(||\mathbf{H}_i \mathbf{H}_i^T||_1\right. \\ &\left. - ||\mathbf{H}_i||_2^2\right) + \mathrm{tr}(\mathbf{b}_i \mathbf{H}_i). \end{aligned} \tag{12}$$

To facilitate the optimization, we assume that $\mathbf{G}_i = \mathbf{U}_i \mathbf{Y}_i$ and $\mathbf{Q}_i = \mathbf{U}_i \mathbf{X}_i$, where $\mathbf{U}_i = [u_i] = [\mathrm{diag}((p_j^i)^{1/2})]$. Therefore, (10) is rewritten as

$$\begin{aligned} J_H = &\sum_{i=0}^{N}\left(||\mathbf{Q}_i \mathbf{W}_i \mathbf{H}_i - \mathbf{G}_i||_F^2 + \mu\left(||\mathbf{H}_i \mathbf{H}_i^T||_1 - ||\mathbf{H}_i||_2^2\right)\right. \\ &\left. + \mathrm{tr}(\mathbf{b}_i \mathbf{H}_i)\right). \end{aligned} \tag{13}$$

By taking the derivative of $\mathbf{H}_i$ in (13), we obtain

$$\begin{aligned} \frac{\partial J_H}{\mathbf{H}_i} = &-2\mathbf{W}_i^T \mathbf{Q}_i^T (\mathbf{G}_i - \mathbf{Q}_i \mathbf{W}_i \mathbf{H}_i) \\ &+ 2\mu(\mathbf{1}_{k\times k} - \mathbf{E}_k)\mathbf{H}_i + \mathbf{b}_i. \end{aligned} \tag{14}$$

According to the Karush–Kuhn–Tucker (KKT) condition $b_{jl}^i H_{jl}^i = 0$, the updating rule for $\mathbf{H}_i$ is obtained

$$H_{jk}^i \leftarrow H_{jk}^i \frac{\left(\mathbf{W}_i^T \mathbf{Q}_i^T \mathbf{G}_i + \mu \mathbf{E}_k \mathbf{H}_i\right)_{jk}}{\left(\mathbf{W}_i^T \mathbf{Q}_i^T \mathbf{Q}_i \mathbf{W}_i \mathbf{H}_i + \mu \mathbf{1}_{k\times k} \mathbf{H}_i\right)_{jk}}. \tag{15}$$

3) *Update* $\mathbf{W}_i$ *With Fixed* $\mathbf{H}_i$ *and* $\mathbf{P}_i$*:* The terms related to $\mathbf{H}_i$ and $\mathbf{P}_i$ are regarded as constants when $\mathbf{H}_i$ and $\mathbf{P}_i$ are fixed, and (10) is modified to

$$J_W = \sum_{i=0}^{N}\left(\sum_{j=1}^{n_i} p_j^i ||\mathbf{y}_j^i - \mathbf{x}_j^i \mathbf{W}_i \mathbf{H}_i||_F^2\right) + \lambda\big(||\mathbf{W}_i\mathbf{W}_i^T||_1 - ||\mathbf{W}_i||_2^2\big) + \frac{\alpha}{2}||\mathbf{W}_i^T\mathbf{W}_i - \mathbf{E}_k||_2 + \mathrm{tr}(\mathbf{a}_i\mathbf{W}_i). \tag{16}$$

We assume that $\mathbf{Q}_i = \mathbf{U}_i\mathbf{X}_i$, and (16) is rewritten as

$$J_W = \sum_{i=0}^{N}\big(||\mathbf{G}_i - \mathbf{Q}_i\mathbf{W}_i\mathbf{H}_i||_F^2 + \lambda\big(||\mathbf{W}_i\mathbf{W}_i^T||_1 - ||\mathbf{W}_i||_2^2\big) + \frac{\alpha}{2}||\mathbf{W}_i^T\mathbf{W}_i - \mathbf{E}_k||_2 + \mathrm{tr}(\mathbf{a}_i\mathbf{W}_i)\big). \tag{17}$$

By taking the derivative of $\mathbf{W}_i$ in (17), we obtain

$$\frac{\partial J_W}{\mathbf{W}_i} = -2\mathbf{Q}^T(\mathbf{G}_i - \mathbf{Q}_i\mathbf{W}_i\mathbf{H}_i)\mathbf{H}_i^T + 2\lambda(\mathbf{1}_{d\times d} - \mathbf{E}_d)\mathbf{W}_i + 2\alpha\mathbf{W}_i\mathbf{W}_i^T\mathbf{W}_i - 2\alpha\mathbf{W}_i + \mathbf{a}_i. \tag{18}$$

According to the KKT condition $a_{jl}^i W_{jl}^i = 0$, the updating rule for $\mathbf{W}_i$ is obtained

$$W_{jk}^i \leftarrow W_{jk}^i \frac{(\mathbf{Q}_i^T\mathbf{G}_i\mathbf{H}_i^T + \lambda\mathbf{E}_d\mathbf{W}_i + \alpha\mathbf{W}_i)_{jk}}{(\mathbf{Q}_i^T\mathbf{Q}_i\mathbf{W}_i\mathbf{H}_i\mathbf{H}_i^T + \lambda\mathbf{1}_{d\times d}\mathbf{W}_i + \alpha\mathbf{W}_i\mathbf{W}_i^T\mathbf{W}_i)_{jk}}. \tag{19}$$

We select the top h features based on the $\mathbf{W}_i$ matrix sorted in descending order.

B. Time Complexity Analysis

The pseudocode of PHFS is summarized in Algorithm 1, and the symbol O(·) indicates the time complexity. The PHFS time complexity mainly focuses on the calculation of $\mathbf{W}_i$ and $\mathbf{H}_i$. The time complexity in (15) is mainly affected by $\mathbf{W}_i^T\mathbf{Q}_i^T\mathbf{G}_i$ and $\mathbf{W}_i^T\mathbf{Q}_i^T\mathbf{Q}_i\mathbf{W}_i\mathbf{H}_i$, where each requires $kdn_i + kmn_i$ and $kdn_i + k^2d + k^2m$ operations. The time complexity in (15) is O($kdn_i + kmn_i + k^2d + k^2m$). The time complexity in (19) is mainly affected by $\mathbf{Q}_i^T\mathbf{G}_i\mathbf{H}_i^T$ and $\mathbf{Q}_i^T\mathbf{Q}_i\mathbf{W}_i\mathbf{H}_i\mathbf{H}_i^T$, where each requires $dmn_i + dmk$ and $d^2n_i + d^2k$ operations. The time complexity in (15) is O($dmn_i + dmk + d^2n_i + d^2k$). Therefore, the time complexity of PHFS is O($R(d^3 + d^2n_i + d^2m + dmn_i)$) because $k = 15\%d$, where R is the operation number.

V. Experimental Studies

In this section, the performance of PHFS is evaluated from five aspects: 1) performance comparison with varying subspace numbers; 2) performance comparison with different methods; 3) experimental results on the document dataset; 4) ablation experiments of different regularization terms; and 5) comparison experiments of matrix factorization.

All experiments are conducted on a PC with Intel Core i7-3770, 3.40-GHz CPU, and 64-bit Windows 10 operating system. The code and datasets used in the proposed model will be opened to https://github.com/fhqxa/PHFS.git. Except for special instructions, PHFS uses the SP-regularizer term of Mixture 2. Table II lists the running time of PHFS on different datasets. As can be seen, the running time increases with the complexity and size of the datasets, with ILSVRC having the longest training time due to its large scale and high-dimensional features.

Algorithm 1 Algorithm of PHFS

Input: (1) Feature matrix $\mathbf{X}_i \in \mathbf{R}^{n_i\times d}$; (2) Label matrix $\mathbf{Y}_i \in \mathbf{R}^{n_i\times m}$; (3) Parameters for selection strategies: λ_1, λ_2, and γ; (4) Parameters for matrix factorization: μ, α, and the subspace number k; and (5) Maximal iteration number T.
Output: (1) Sample weight $\mathbf{P}_i \in \mathbf{R}^{1\times n_i}$; (2) Indicator matrix $\mathbf{W}_i \in \mathbf{R}^{d\times k}$; and (3) Coefficient matrix $\mathbf{H}_i \in \mathbf{R}^{k\times m}$.

```
Set iteration number t = 0;
Initialize W = ones (d, k), H_i = rand (k, m), and W = [W_0, W_1, ..., W_N];
while t < T do
    for i = 0 : N do
        Fix W_i^(t) and H_i^(t), update P_i^(t) according to Table I;
        Update Q_i^(t) by Q_i = diag(sqrt(P_i^(t)))X_i;
        Update G_i^(t) by G_i = diag(sqrt(P_i^(t)))Y_i;
        Fix H_i^(t) and P_i^(t), update W_i^(t) according to (19);
        Fix W_i^(t) and P_i^(t), update H_i^(t) according to (15);
    end for
    UpdateP^(t+1) = [P_0^(t+1), P_1^(t+1), ..., P_N^(t+1)];
    UpdateH^(t+1) = [H_0^(t+1), H_1^(t+1), ..., H_N^(t+1)];
    UpdateW^(t+1) = [W_0^(t+1), W_1^(t+1), ..., W_N^(l+1)];
    t = t + 1;
end while
return P, H, and W;
```

TABLE II
Running Time of PHFS on Different Datasets (s)

Dataset	Running time
DD	2.90
F194	9.83
AWA	6.68
VOC	42.13
ILSVRC	120.28
Car	221.92
Cifar	1186.04
SUN	1418.27

A. Experimental Setup

We introduce the comparison methods, experience dataset, evaluation metrics, and experimental parameter settings before conducting the experiments.

1) Comparison Methods: They are briefly introduced as follows. HFisher [38] improves a label distance in hierarchical datasets. HFSNM [39] employs $\ell_{2,1}$-norm sparse learning for loss and regularization term optimization. HMRMR [40] selects feature subsets minimizing redundancy and maximizing correlation. HiFSRR [4] exploits class dependency relationships in hierarchical feature selection. HFSDK [41] uses parent–child relationships to mitigate outlier impact

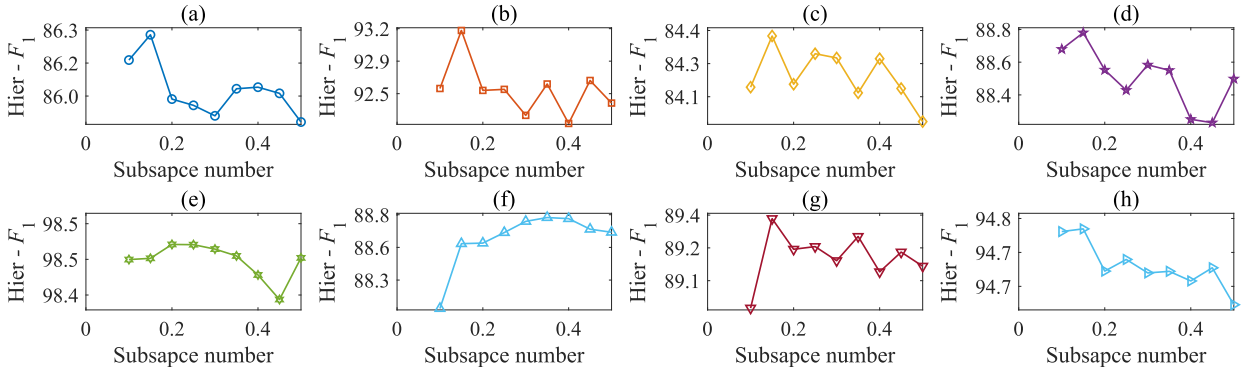

Fig. 5. Hier-F_1 of PHFS for selecting different numerical subspaces (SVM). (a) *AWA*, (b) *DD*, (c) *F194*, (d) *VOC*, (e) *ILSVRC*, (f) *Car*, (g) *Cifar*, and (h) *SUN*.

TABLE III
DESCRIPTIONS OF HIERARCHICAL DATASETS

Dataset	Type	Feature	Training	Test	Class
DD	Protein	473	3,020	605	27
F194	Protein	473	7,105	1,420	194
AWA	Images	252	6,405	3,202	10
VOC	Images	4,096	3,437	3,539	20
ILSVRC	Images	4,096	12,346	11,845	57
Car	Images	4,096	8,144	7,541	196
Cifar	Images	4,096	50,000	10,000	100
SUN	Images	4,096	45,109	22,556	324

on hierarchical classification. HFSCN [42] applies an upper limit ℓ_2-norm to resist noisy data. HFSGR [43] constructs a subtree graph to consider the directionality of class dependencies. LCCSHFS [44] embeds label correlation analysis within different hierarchical structures in the selection process. HFSCF [45] accounts for consistency within class groups and differences across tasks to select features. HFSLDL [46] solves the classification task with class imbalance by converting the stratified labels into label distribution and exploiting the correlation between classes. HFS-MIMR [6] emphasizes task independence to reduce redundancy among features within tasks in a hierarchical structure to make the boundaries clearer. HFSLE [47] overcomes semantic disparities in the hierarchical framework through the enhancement of labels. DMFFS [48] decomposes large-scale tasks into smaller subtasks and proposes a dynamic, multitask feature selection system designed for feature selection in new tasks.

2) Experiment Dataset: They are eight real-world hierarchical datasets, and the descriptions are listed in Table III.

3) Evaluation Indicators: They include the Hier-F_1 [49], F_{LCA} (lowest common ancestors, LCAs) [50], and tree induced error (TIE) [51] indicators. We select feature subsets from the training data and assess them on the test data using fivefold cross validation with identical parameter settings, deploying SVM and KNN classifiers for performance evaluation.

Hier-F_1 incorporates ancestor and descendant classes, while F_{LCA} employs graph theory to represent hierarchical label distances via the least common ancestor. They are calculated as follows:

$$\frac{2 \cdot P_H \cdot R_H}{P_H + R_H} \tag{20}$$

where hierarchical precision (P_H) and recall (R_H) are defined as

$$P_H = \frac{|\hat{D}_{\text{aug}} \cap D_{\text{aug}}|}{|\hat{D}_{\text{aug}}|} \quad \text{and} \quad R_H = \frac{|\hat{D}_{\text{aug}} \cap D_{\text{aug}}|}{|D_{\text{aug}}|}.$$

Both Hier-F_1 and F_{LCA} consider the entire set of predicted and true classes, but they differ in how the sets are augmented. For Hier-F_1, the sets D (true classes) and $\hat{D}$ (predicted classes) are augmented with their ancestors: $D_{\text{aug}} = D \cup \text{anc}(D)$ and $\hat{D}_{\text{aug}} = \hat{D} \cup \text{anc}(\hat{D})$. For F_{LCA}, the function anc$(\cdot)$ represents the least common ancestors.

TIE quantifies error edges between two nodes d_u (correct label) and d_v (predicted label) in a hierarchical tree structure H is given by the length of the unique path connecting these nodes: $\text{TIE}(d_u, d_v) = |E_H(d_u, d_v)|$, where $E_H(d_u, d_v)$ is the set of edges along the path from d_u to d_v and $|\cdot|$ denotes the number of edges.

4) Parameter Settings: They mainly involve the feature selection ratio, the number of subspaces, and three optimization parameters, which are used in progressive learning strategies and matrix factorization. They are given as follows.

1) *Feature Selection Ratio:* For image datasets, 20% of the features are selected, and for protein datasets, 10% of the features are selected. This ratio is chosen based on the approach described in [4].
2) *Number of Subspaces:* Extensive experiments are performed to determine the optimal number of subspaces. The detailed results and analysis are presented in Section V-B.
3) *Optimization Parameters:* The parameters related to matrix factorization are set to fixed values based on experimental validation.
   a) $\alpha = 10$*:* This parameter enhances the column vector orthogonality of the basis matrix, ensuring basis vector independence and simplifying calculations.
   b) $\lambda = 1$ *and* $\mu = 1$*:* These parameters balance the sparsity and orthogonality of the matrices $\mathbf{W}_i$ and $\mathbf{H}_i$ by encouraging independence, thereby reducing feature redundancy.

B. Subspace Number Impact on Performance

The impact of subspace number k shown in Fig. 5 are validated, which helps us determine the optimal allocation of

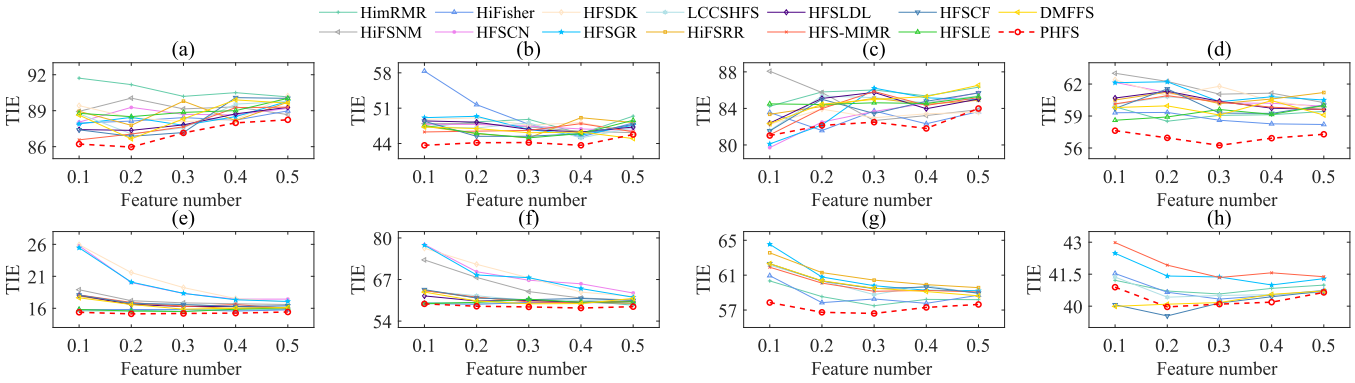

Fig. 6. All methods on eight datasets using the TIE indicator (KNN). (a) *AWA*, (b) *DD*, (c) *F194*, (d) *VOC*, (e) *ILSVRC*, (f) *Car*, (g) *Cifar*, and (h) *SUN*.

k to optimize classification results. The following conclusions are drawn by analyzing the overall trend of Fig. 5.

1) The accuracy of the overall dataset shows a trend of first increasing and then stabilizing as the subspace numbers increased. The accuracy gradually improves when $k \leq 15\%d$, while the accuracy tends to stabilize or slightly decrease when $k > 15\%d$. The existence of too many subspaces resulted in feature redundancy, meaning that the selected features contain similar information when representing the data, wasting computational resources and storage space. On the contrary, too few subspaces loss important information, where crucial features are excluded, causing the model to inaccurately describe the data.
2) There are certain differences in overall trends among different datasets. The accuracy curve of each dataset shows that when $k \approx 15\%d$, the accuracy of most datasets reaches its peak or remains stable. However, a few datasets exhibit slightly different trends under different subspace numbers due to the characteristics of the dataset, such as data distribution and noise level. Therefore, subsequent experiments follow $k = 15\%d$, which avoids dimensional disasters while maintaining high accuracy.

C. Performance Comparison of Different Methods

We compare the performance of PHFS with other methods measured with the TIE, F_{LCA}, and Hier-F_1 indicators. First, Fig. 6 shows the TIE metrics of different methods on different datasets.

By analyzing the results in Fig. 6, the following conclusions can be drawn.

1) *Performance of PHFS Across Datasets:* PHFS consistently exhibits robust performance across various datasets, particularly those with complex hierarchical structures, such as *VOC*, *ILSVRC*, *Car*, and *Cifar*. Its ability to dynamically adjust its feature selection strategy based on the evolving characteristics of label noise allows for greater flexibility and adaptability. This dynamic adjustment enables PHFS to better capture the underlying data structure, resulting in improved feature selection accuracy and overall performance compared to other methods.
2) *Performance With Varying Feature Ratios:* PHFS demonstrates consistent and robust performance across varying feature ratios on multiple datasets, particularly on *VOC* and *Cifar*. This is due to its dynamic feature selection mechanism, which adjusts the selection of relevant features according to the underlying noise patterns in the data. By incorporating matrix factorization techniques, PHFS can leverage historical information from the previously selected samples, ensuring that it maintains high accuracy in selecting discriminative features even with fewer features.

Second, Table IV lists the F_{LCA} and Hier-F_1 values on the different classifiers. The following conclusions are drawn from the analysis of the results in Table IV.

1) *Performance of PHFS on Different Domain Datasets:* PHFS demonstrates superior performance across various domains, including protein datasets (e.g., *DD* and *F194*) and image datasets (e.g., *Cifar*). For example, PHFS achieves F_{LCA} of 85.82 in *Cifar* and F_{LCA} of 89.09 in *DD* when evaluated using the KNN classifier. These results significantly surpass those of other methods, indicating that PHFS effectively identifies key features through an adaptive sample weighting strategy, thereby enhancing its resilience to mislabelled and noisy samples.
2) *Robustness of PHFS Across Datasets of Different Scales:* PHFS exhibits superior performance on small-scale datasets (e.g., *AWA* and *DD*) and large-scale datasets (e.g., *ILSVRC* and *Cifar*). For example, PHFS achieves Hier-F_1 of 93.22 in *DD* when evaluated using the SVM classifier, and it yields Hier-F_1 of 87.91 in *Cifar* when evaluated using the KNN classifier. These results suggest that PHFS can flexibly adjust its feature selection strategy according to the characteristics of the data, demonstrating strong robustness and high classification performance across datasets of varying sizes.
3) *Adaptability of PHFS to Different Classifiers:* PHFS exhibits significant advantages when applied to both KNN and SVM classifiers. Its progressive sample selection and feature weighting strategies can effectively accommodate the diverse requirements of these classifiers. For KNN, PHFS enhances model robustness by removing mislabelled samples, thereby improving the

TABLE IV
HIER-F_1 AND F_{LCA} USING DIFFERENT METHODS ON DIFFERENT CLASSIFIERS. (A) KNN. (B) SVM

(a)

Method	Metric	*DD*	*F194*	*AWA*	*VOC*	*ILSVRC*	*Car*	*Cifar*	*SUN*
HmRMR	F_{LCA}	88.19	78.94	77.21	85.37	96.10	85.09	85.37	89.82
	Hier-F_1	90.84	83.87	83.46	88.28	97.98	87.97	87.54	93.82
HFSNM	F_{LCA}	88.02	77.98	77.49	84.44	95.70	83.09	–	–
	Hier-F_1	90.66	82.64	83.75	87.49	97.76	85.98	–	–
HiFisher	F_{LCA}	85.42	79.11	77.97	85.17	96.05	85.19	85.53	89.84
	Hier-F_1	88.95	83.85	84.10	88.10	97.95	88.05	87.69	93.83
HFSCN	F_{LCA}	88.05	80.07	77.68	84.70	94.99	82.64	–	–
	Hier-F_1	90.60	84.76	83.85	87.71	97.41	85.58	–	–
HFSDK	F_{LCA}	87.81	79.44	77.90	84.78	94.60	82.06	–	–
	Hier-F_1	90.34	84.25	84.07	87.77	97.22	84.94	–	–
HFSGR	F_{LCA}	87.72	79.97	77.89	84.45	94.98	82.89	84.80	89.65
	Hier-F_1	90.45	84.74	83.96	87.50	97.41	85.84	87.11	93.69
LCCSHFS	F_{LCA}	88.27	79.60	78.23	84.80	95.82	84.52	84.91	89.89
	Hier-F_1	90.83	84.41	84.34	87.85	97.83	87.40	87.20	93.86
HFSLDL	F_{LCA}	88.14	79.16	78.05	84.69	95.83	84.68	–	–
	Hier-F_1	90.73	84.10	84.21	87.67	97.84	87.60	–	–
HiFSRR	F_{LCA}	87.88	79.40	78.16	84.67	95.82	84.90	84.93	89.92
	Hier-F_1	90.47	84.27	84.19	87.69	97.83	87.75	87.19	93.89
HFSCF	F_{LCA}	87.90	79.62	78.28	84.61	95.80	84.70	84.92	**90.11**
	Hier-F_1	90.60	84.50	84.30	87.69	97.83	87.59	87.23	**93.97**
HFS-MIMR	F_{LCA}	88.43	79.72	78.24	84.77	95.77	84.63	84.98	89.52
	Hier-F_1	91.00	84.56	84.28	87.85	97.80	87.52	87.26	93.68
HFSLE	F_{LCA}	88.11	78.88	77.87	85.27	96.11	85.03	–	–
	Hier-F_1	90.69	83.79	83.99	88.24	97.97	87.88	–	–
DMTFS	F_{LCA}	88.18	79.42	78.34	85.01	95.85	84.93	84.91	89.98
	Hier-F_1	90.77	84.34	84.34	88.00	97.85	87.73	87.19	93.89
PHFS	F_{LCA}	**89.09**	**79.74**	**78.51**	**85.76**	**96.15**	**85.34**	**85.82**	90.01
	Hier-F_1	**91.61**	**84.45**	**84.50**	**88.61**	**98.04**	**88.17**	**87.91**	93.94

(b)

Method	Metric	*DD*	*F194*	*AWA*	*VOC*	*ILSVRC*	*Car*	*Cifar*	*SUN*
HmRMR	F_{LCA}	90.58	79.10	80.49	86.44	97.27	87.24	87.44	91.99
	Hier-F_1	92.73	84.21	86.06	88.98	98.58	89.66	89.79	95.02
HFSNM	F_{LCA}	90.08	76.38	81.74	86.16	97.05	84.08	–	–
	Hier-F_1	92.29	81.81	86.77	88.84	98.46	86.53	–	–
HiFisher	F_{LCA}	86.37	77.19	81.55	86.48	97.25	87.31	88.49	91.99
	Hier-F_1	89.73	83.02	86.61	89.04	98.56	89.72	90.15	95.01
HFSCN	F_{LCA}	90.50	77.95	81.69	86.55	96.70	82.42	–	–
	Hier-F_1	92.70	83.47	86.74	89.14	98.29	84.89	–	–
HFSDK	F_{LCA}	90.53	78.62	81.94	86.57	96.49	80.61	–	–
	Hier-F_1	92.67	83.91	86.88	89.09	98.19	83.02	–	–
HFSGR	F_{LCA}	90.62	78.96	81.81	86.45	96.67	82.92	88.44	92.18
	Hier-F_1	92.77	84.22	86.81	89.07	98.27	85.39	90.14	95.14
LCCSHFS	F_{LCA}	90.48	**79.75**	**82.01**	86.61	97.24	86.79	88.57	92.19
	Hier-F_1	92.75	**84.82**	**86.92**	89.15	98.57	89.22	90.23	95.16
HFSLDL	F_{LCA}	90.62	79.64	80.34	86.63	97.27	86.84	–	–
	Hier-F_1	92.81	84.69	85.99	89.21	98.59	89.19	–	–
HiFSRR	F_{LCA}	90.33	79.66	81.87	86.43	97.23	86.88	88.70	92.21
	Hier-F_1	92.55	84.78	86.81	89.01	98.56	89.23	90.32	95.16
HFSCF	F_{LCA}	91.03	79.69	82.15	86.98	97.22	86.81	88.58	92.33
	Hier-F_1	93.07	84.79	86.99	89.46	98.56	89.19	90.23	95.23
HFS-MIMR	F_{LCA}	90.49	78.17	82.09	86.83	97.24	86.75	88.66	92.04
	Hier-F_1	92.64	83.62	86.97	89.35	98.56	89.10	90.28	95.09
HFSLE	F_{LCA}	90.45	79.17	81.53	86.69	97.27	87.43	–	–
	Hier-F_1	92.53	84.36	86.61	89.24	98.58	89.84	–	–
DMTFS	F_{LCA}	90.34	79.68	82.29	86.91	97.28	87.17	88.45	**92.37**
	Hier-F_1	92.58	84.81	87.07	89.42	98.59	89.49	90.33	**95.25**
PHFS	F_{LCA}	**91.16**	79.73	81.93	**86.96**	**97.34**	**87.27**	**88.77**	92.15
	Hier-F_1	**93.22**	84.77	86.87	**89.43**	**98.63**	**89.68**	**90.36**	95.12

discriminability of the feature space. In the case of SVM, PHFS accurately selects representative samples, enhancing its decision-making capabilities in high-dimensional data. Consequently, PHFS effectively integrates the characteristics of various classifiers, ensuring adaptability across a range of tasks and datasets.

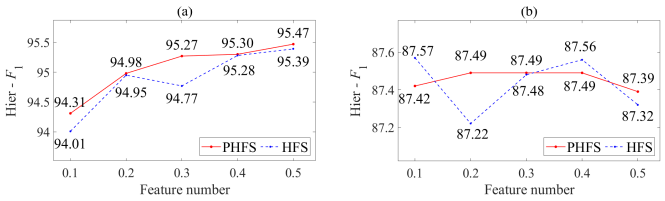

Fig. 7. Hier-F_1 comparison between Hier-FS and PHFS on 20 Newsgroups. (a) SVM. (b) KNN.

D. Experimental Results on Document Dataset

To assess the performance of PHFS in practical applications, it was applied to the 20 Newsgroups dataset, which comprises 26 214 features and is utilized for document classification. This dataset contains 18 446 discussions across 20 different newsgroups. The "Bydate" version of the dataset in this experiment is adopted, which consists of 951 documents that are evenly distributed in 20 classes.

The experimental comparison of Hier-F_1 between Hier-FS and PHFS, as determined by KNN and SVM classifiers, is presented in Fig. 7. After performing a thorough analysis, the following results are obtained. PHFS consistently improved performance across various feature subset sizes. Specifically, PHFS achieved an accuracy of 94.31% using only 10% of the dataset's features with SVM. The accuracy of PHFS shows an upward trend as the proportion of the selected feature subset increased and reached a peak accuracy of 95.47% with 50% of the selected features on SVM. These results highlight the advantage of PHFS in feature selection, accurately identifying and utilizing key features for classification tasks. This approach helps to avoid the performance degradation associated with high-dimensional data and significantly boosts the model performance.

E. Ablation Experiments of Different Regularization Terms

Ablation experiments are conducted to evaluate the contributions of sample selection strategies and matrix factorization in PHFS. The impact on the algorithm's performance is assessed under various settings by removing different regularization terms from SFSPS to understand which aspects significantly improved the overall performance. Fig. 8 shows the Hier-F_1 indicators for methods using different strategies. *Base* uses original features as a performance baseline. *Subspace* enhances *Base* with matrix factorization, while *Mixture* further integrates sample selection strategies. The following conclusions can be derived by analyzing Fig. 8.

1) *Subspace* demonstrates enhanced performance on each dataset over *Base* by exploring subspaces or potential structures that contain relevant features within high-dimensional data. This improvement is achieved through the utilization of matrix factorization, which allows the algorithm to extract more informative and discriminative feature subsets from the original feature space.
2) *Mixture* outperforms the other two methods on each dataset by accurately selecting the most informative

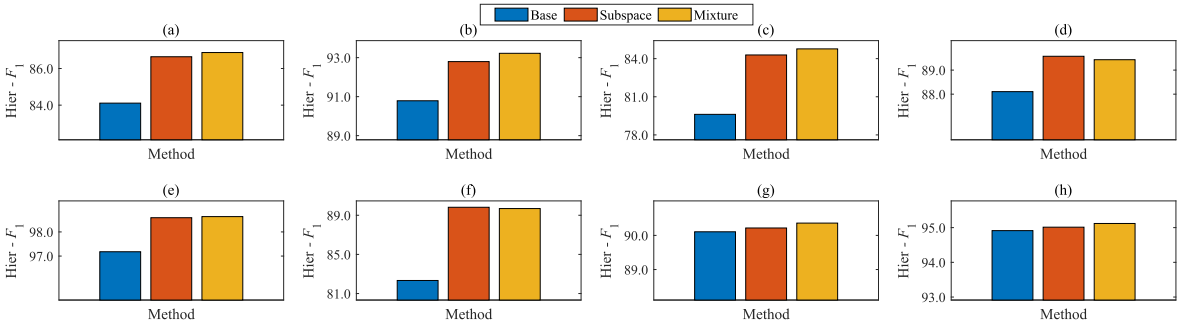

Fig. 8. Hier-F_1 indicators of different methods (SVM). (a) *AWA*, (b) *DD*, (c) *F194*, (d) *VOC*, (e) *ILSVRC*, (f) *Car*, (g) *Cifar*, and (h) *SUN*.

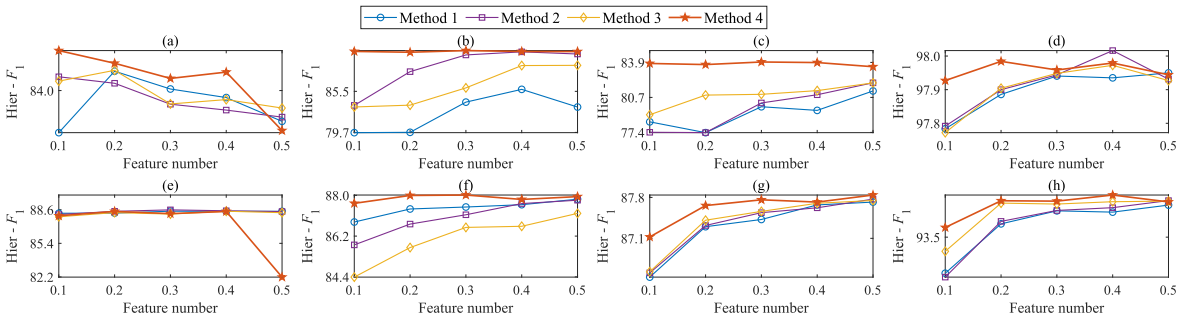

Fig. 9. Performance of regularization terms related to subspaces (KNN). (a) *AWA*, (b) *DD*, (c) *F194*, (d) *VOC*, (e) *ILSVRC*, (f) *Car*, (g) *Cifar*, and (h) *SUN*.

features. The sample selection strategies are crucial in ensuring that the selected samples effectively represent the characteristics and distribution of the entire dataset. These strategies take into account the representativeness of the samples, employing various criteria to achieve this goal. Therefore, *Mixture* achieves superior classification results by utilizing both matrix factorization and effective sample selection strategies.

F. Comparison Experiment of Matrix Factorization

The role of matrix factorization in PHFS is explored. The specific experimental settings are as follows. *Method* 1 serves as a control without regularization. *Methods* 2 and 3 extend *Method* 1 by applying regularization to $\mathbf{W}_i$ and $\mathbf{H}_i$, respectively, while *Method* 4 combines both regularizations. Fig. 9 validates the role of the regularization terms related to $\mathbf{W}_i$ and $\mathbf{H}_i$. From Fig. 9, the following conclusions can be drawn.

1) *Method* 2 performs well on most datasets, achieving higher accuracy compared with *Method* 1. For example, the indicators of *ILSVRC* and *DD* are significantly higher than those of the *Base* strategy. *Method 2* handles the differences in the importance of different features or data subspaces and encourages more balanced feature and subspace representations.
2) *Method* 3 performs unevenly on different datasets, with some datasets having higher Hier-F_1 values, while others have slightly lower Hier-F_1 values. The Hier-F_1 value can be improved by applying inner product regularization to the representation matrix $\mathbf{H}_i$. For example, achieving higher accuracy on *ILSVRC* is an improvement compared with *Method* 1.
3) *Method* 4 is the combination strategy, which achieves higher accuracy than the *Base* strategy on most datasets and slightly decreases compared to using inner product regularization alone in some cases. This combination considers the influence of the $\mathbf{W}_i$ and $\mathbf{H}_i$ matrices by limiting their norms. This regularization strategy effectively controls the algorithm's complexity and promotes parameter smoothness, ultimately leading to better generalization ability and preventing overfitting. The slight decrease in performance in certain cases may be attributed to the interplay between regularization terms and the specific characteristics of individual datasets.

VI. Conclusion and Future Work

A hierarchical feature selection method with adaptive sample weighting was proposed in this work to improve the classification accuracy. The main idea is to prioritize correctly labeled samples and gradually introduce mislabelled samples, utilizing the structure information retained by matrix decomposition to reduce the impact of mislabelled data in the later stages. While PHFS excels in handling high-dimensional and noisy data, more computational resources and careful tuning of the various parameters may be required. In the future, automatic parameter adjustment technology will be explored to automatically search for the optimal parameter configuration.

References

[1] M. Xiao et al., "Hierarchical interdisciplinary topic detection model for research proposal classification," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 9, pp. 9685–9699, Sep. 2023.
[2] R. Jiao, B. Xue, and M. Zhang, "Learning to preselection: A filter-based performance predictor for multiobjective feature selection in classification," *IEEE Trans. Evol. Comput.*, early access, Mar. 6, 2024, doi: 10.1109/TEVC.2024.3373802.
[3] B. Hoai Nguyen, B. Xue, and M. Zhang, "A constrained competitive swarm optimizer with an SVM-based surrogate model for feature selection," *IEEE Trans. Evol. Comput.*, vol. 28, no. 1, pp. 2–16, Feb. 2024.
[4] H. Zhao, Q. Hu, P. Zhu, Y. Wang, and P. Wang, "A recursive regularization based feature selection framework for hierarchical classification," *IEEE Trans. Knowl. Data Eng.*, vol. 33, no. 7, pp. 2833–2846, Jul. 2021.

[5] J. Zheng, C. Luo, T. Li, and H. Chen, "A novel hierarchical feature selection method based on large margin nearest neighbor learning," *Neurocomputing*, vol. 497, pp. 1–12, Aug. 2022.

[6] J. Shi, Z. Li, and H. Zhao, "Feature selection via maximizing inter-class independence and minimizing intra-class redundancy for hierarchical classification," *Inf. Sci.*, vol. 626, pp. 1–18, May 2023.

[7] C. Sheng and P. Song, "Graph regularized virtual label regression for unsupervised feature selection," *Digit. Signal Process.*, vol. 123, Apr. 2022, Art. no. 103393.

[8] B. Jiang et al., "Semi-supervised multi-view feature selection with adaptive graph learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 35, no. 3, pp. 3615–3629, Mar. 2024.

[9] S. Wang, W. Pedrycz, Q. Zhu, and W. Zhu, "Subspace learning for unsupervised feature selection via matrix factorization," *Pattern Recognit.*, vol. 48, no. 1, pp. 10–19, Jan. 2015.

[10] Z. Song and P. Song, "Latent energy preserving embedding for unsupervised feature selection," *Digit. Signal Process.*, vol. 132, Jan. 2023, Art. no. 103794.

[11] J. Xia et al., "GNN cleaner: Label cleaner for graph structured data," *IEEE Trans. Knowl. Data Eng.*, vol. 36, no. 2, pp. 640–651, Feb. 2024.

[12] B. Lerner and R. Meir, "Graph-based semi-supervised learning with noisy labels," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 11, pp. 1783–1796, Aug. 2005.

[13] Y. You, T. Chen, Y. Shen, and Z. Wang, "Graph contrastive learning with augmentations," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 19496–19507.

[14] X. Wang, Y. Chen, and W. Zhu, "A survey on curriculum learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 9, pp. 4555–4576, Mar. 2021.

[15] L. Yang, G. Liu, W. Liu, H. Bai, J. Zhai, and Y. Dai, "Detecting multielement algorithmically generated domain names based on adaptive embedding model," *Secur. Commun. Netw.*, vol. 2021, pp. 1–20, May 2021.

[16] J. Xia, Y. Luo, Z. Liu, Y. Zhang, H. Shi, and Z. Liu, "Cooperative multi-target hunting by unmanned surface vehicles based on multi-agent reinforcement learning," *Defence Technol.*, vol. 29, pp. 80–94, Nov. 2023.

[17] H. Sun and T. Wang, "Toward causal-aware RL: State-wise action-refined temporal difference," in *Proc. Deep Reinforcement Learn. Workshop (NeurIPS)*, 2022, pp. 1–14.

[18] M. Kumar, B. Packer, and D. Koller, "Self-paced learning for latent variable models," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 23, 2010, pp. 1–9.

[19] Q. Wang, Y. Zhou, Z. Cao, and W. Zhang, "M2SPL: Generative multiview features with adaptive meta-self-paced sampling for class-imbalance learning," *Expert Syst. Appl.*, vol. 189, Mar. 2022, Art. no. 115999.

[20] L. Wan, C. Dong, and X. Pei, "Self-paced learning-based multi-graphs semi-supervised learning," *Multimedia Tools Appl.*, vol. 81, no. 5, pp. 7025–7046, Feb. 2022.

[21] A. Pujahari and D. S. Sisodia, "Item feature refinement using matrix factorization and boosted learning based user profile generation for content-based recommender systems," *Expert Syst. Appl.*, vol. 206, Nov. 2022, Art. no. 117849.

[22] Y. Bengio, J. Louradour, and R. Collobert, "Curriculum learning," in *Proc. Int. Conf. Mach. Learn.*, Aug. 2009, pp. 41–48.

[23] Y. Li, C. Ma, Y. Tao, Z. Hu, Z. Su, and M. Liu, "A robust cost-sensitive feature selection via self-paced learning regularization," *Neural Process. Lett.*, vol. 54, no. 4, pp. 2571–2588, Aug. 2022.

[24] J. Miao et al., "Self-paced non-convex regularized analysis–synthesis dictionary learning for unsupervised feature selection," *Knowl.-Based Syst.*, vol. 241, Apr. 2022, Art. no. 108279.

[25] W. Li, H. Chen, T. Li, J. Wan, and B. Sang, "Unsupervised feature selection via self-paced learning and low-redundant regularization," *Knowl.-Based Syst.*, vol. 240, Mar. 2022, Art. no. 108150.

[26] C. Zeng, H. Chen, T. Li, and J. Wan, "Robust unsupervised feature selection via sparse and minimum-redundant subspace learning with dual regularization," *Neurocomputing*, vol. 511, pp. 1–21, Oct. 2022.

[27] L. Li, Y. Zhang, Q. Lin, Z. Ming, C. A. C. Coello, and V. C. Leung, "Superpixel segmentation based evolutionary multitasking algorithm for feature selection of hyperspectral images," *IEEE Trans. Evol. Comput.*, early access, Apr. 23, 2024, doi: 10.1109/TEVC.2024.3392749.

[28] H. Xu, B. Xue, and M. Zhang, "Probe population based initialization and genetic pool based reproduction for evolutionary bi-objective feature selection," *IEEE Trans. Evol. Comput.*, early access, May 21, 2024, doi: 10.1109/TEVC.2024.3403655.

[29] Y. Li, K. Wu, and J. Liu, "Self-paced ARIMA for robust time series prediction," *Knowl.-Based Syst.*, vol. 269, Jun. 2023, Art. no. 110489.

[30] J. Peng, P. Wang, C. Desrosiers, and M. Pedersoli, "Self-paced contrastive learning for semi-supervised medical image segmentation with meta-labels," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 34, 2021, pp. 16686–16699.

[31] L. Liu, Z. Wang, J. Bai, X. Yang, Y. Yang, and J. Zhou, "Ensemble self-paced learning based on adaptive mixture weighting," *Electronics*, vol. 11, no. 19, p. 3154, Oct. 2022.

[32] C. Xu, D. Tao, and C. Xu, "Multi-view self-paced learning for clustering," in *Proc. Int. Joint Conf. Artif. Intell.*, Jul. 2015, pp. 3974–3980.

[33] H. Li, M. Gong, M. Zhang, and Y. Wu, "Spatially self-paced convolutional networks for change detection in heterogeneous images," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 14, pp. 4966–4979, 2021.

[34] X. Qu, D. Li, X. Zhao, and B. Gu, "GAGA: Deciphering age-path of generalized self-paced regularizer," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 35, 2022, pp. 32025–32038.

[35] H. Li, Y. Zhao, M. Gong, Y. Wu, and J. Liu, "Self-paced learning method with adaptive mixture weighting," *J. Softw.*, vol. 34, no. 5, pp. 2337–2349, May 2023, doi: 10.13328/j.cnki.jos.006438.

[36] M. Qi, T. Wang, F. Liu, B. Zhang, J. Wang, and Y. Yi, "Unsupervised feature selection by regularized matrix factorization," *Neurocomputing*, vol. 273, pp. 593–610, Jan. 2018.

[37] J. Han, Z. Sun, and H. Hao, "Selecting feature subset with sparsity and low redundancy for unsupervised learning," *Knowl.-Based Syst.*, vol. 86, pp. 210–223, Sep. 2015.

[38] P. Hart, D. Stork, and R. Duda, *Pattern Classification*. Hoboken, NJ, USA: Wiley, 2000.

[39] F. Nie, H. Huang, X. Cai, and C. Ding, "Efficient and robust feature selection via joint $\ell_{2,1}$-norms minimization," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 23, 2010, pp. 1813–1821.

[40] L. Grimaudo, M. Mellia, and E. Baralis, "Hierarchical learning for fine grained internet traffic classification," in *Proc. 8th Int. Wireless Commun. Mobile Comput. Conf. (IWCMC)*, Aug. 2012, pp. 463–468.

[41] X. Liu, Y. Zhou, and H. Zhao, "Robust hierarchical feature selection driven by data and knowledge," *Inf. Sci.*, vol. 551, pp. 341–357, Apr. 2021.

[42] X. Liu and H. Zhao, "Robust hierarchical feature selection with a capped ℓ_2-norm," *Neurocomputing*, vol. 443, pp. 131–146, Jul. 2021.

[43] Q. Tuo, H. Zhao, and Q. Hu, "Hierarchical feature selection with subtree based graph regularization," *Knowl.-Based Syst.*, vol. 163, pp. 996–1008, Jan. 2019.

[44] Y. Lin, S. Bai, H. Zhao, S. Li, and Q. Hu, "Label-correlation-based common and specific feature selection for hierarchical classification," *J. Softw.*, vol. 33, no. 7, pp. 2667–2682, Jul. 2022.

[45] H. Liu et al., "Hierarchical feature selection from coarse to fine," *Acta Electronica Sinica*, vol. 50, no. 11, pp. 2778–2789, 2020.

[46] Y. Lin, H. Liu, H. Zhao, Q. Hu, X. Zhu, and X. Wu, "Hierarchical feature selection based on label distribution learning," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 6, pp. 5964–5976, Jun. 2023.

[47] H. Liu, Y. Lin, C. Wang, L. Guo, and J. Chen, "Semantic-gap-oriented feature selection in hierarchical classification learning," *Inf. Sci.*, vol. 642, Sep. 2023, Art. no. 119241.

[48] Y. Zhang, J. Shi, and H. Zhao, "DMTFS-FO: Dynamic multi-task feature selection based on flexible loss and orthogonal constraint," *Expert Syst. Appl.*, vol. 255, Dec. 2024, Art. no. 124588.

[49] L. Cai and T. Hofmann, "Exploiting known taxonomies in learning overlapping concepts," in *Proc. Int. Joint Conf. Artif. Intell.*, 2007, pp. 708–713.

[50] B. Schieber and U. Vishkin, "On finding lowest common ancestors: Simplification and parallelization," *SIAM J. Comput.*, vol. 17, no. 6, pp. 1253–1262, Dec. 1988.

[51] O. Dekel, J. Keshet, and Y. Singer, "Large margin hierarchical classification," in *Proc. 21st Int. Conf. Mach. Learn. (ICML)*, 2004, p. 27.