Pattern Recognition 179 (2026) 113817

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

# Enhance and reuse: A dual-mechanism approach to boost deep forest for label distribution learning

Jia-Le Xu [a,b], Shen-Huan Lyu [a,b,c,d,*], Yu-Nian Wang [a,b], Ning Chen [a,b], Zhihao Qu [a,b], Bin Tang [a,b], Baoliu Ye [d]

[a] Key Laboratory of Water Big Data Technology of Ministry of Water Resources, Hohai University, Nanjing, 211100, China
[b] College of Computer Science and Software Engineering, Nanjing, 211100, China
[c] Department of Computer Science, City University of Hong Kong, 999077, Hong Kong, China
[d] National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, 210023, China

ARTICLE INFO

*Keywords:*
Label distribution learning
Deep forest
Label correlation

ABSTRACT

Label distribution learning (LDL) requires the learner to predict the degree of correlation between each sample and each label. To achieve this, a crucial task during learning is to leverage the correlation among labels. Deep Forest (DF) is a deep learning framework based on tree ensembles, whose training phase does not rely on backpropagation. DF performs in-model feature transform using the prediction of each layer and achieves competitive performance on many tasks. However, its exploration in the field of LDL is still in its infancy. The few existing methods that apply DF to the field of LDL do not have effective ways to utilize the correlation among labels. Therefore, we propose a method named **E**nhanced and **R**eused Feature **D**eep **F**orest (ERDF). It mainly contains two mechanisms: feature enhancement exploiting label correlation and measure-aware feature reuse. The first one is to utilize the correlation among labels to enhance the original features, enabling the samples to acquire more comprehensive information for the task of LDL. The second one performs a reuse operation on the features of samples that perform worse than the previous layer on the validation set, in order to ensure the stability of the training process. This kind of Enhance-Reuse pattern not only enables samples to enrich their features but also validates the effectiveness of their new features and conducts a reuse process to prevent the noise from spreading further. Experiments show that our method outperforms other comparison algorithms on six metrics.

## 1. Introduction

In multi-label learning (MLL) [1], the label corresponding to an instance is a subset of labels, rather than a single label as in traditional single-label learning (SLL). For example, in movie classification, a movie can be labeled with "comedy", "action", and "romance" simultaneously; in news text annotation, an article can involve both "technology" and "finance" topics. MLL significantly enhances the model's ability to describe complex objects and has been widely applied in various fields such as text classification [2,3], multimedia analysis [4,5] and image content annotation [6–8].

Although MLL has significantly improved the expressive power compared to SLL, it shares the same fundamental assumption as SLL: labels describe instances in a binary manner, i.e., a label is either relevant or irrelevant for an instance. This binary assumption ignores a crucial fact that the contribution or importance of different labels in describing the same instance may vary from each other. For example, a movie might mainly be an action film but also contain a few comedy elements. Traditional learning paradigms cannot capture this semantic ambiguity and uncertainty. To more precisely quantify the relative importance of different labels, label distribution learning (LDL) [9] is proposed as a more general learning paradigm. The core idea of LDL is to associate a label distribution with an instance. This distribution is a real-valued vector, where each element represents the description degree of the corresponding label for the instance, and the sum of all dimensions is 1. In this way, LDL extends the qualitative "yes or no" judgment of label descriptions to a quantitative "degree" measurement, thereby enabling more precise and comprehensive capture of the semantic information of the object. Due to its ability to address the more general scenario of label ambiguity, it has seen wide application in diverse tasks such as facial expression recognition [10–12], object detection [13],

* Corresponding author at: Key Laboratory of Water Big Data Technology of Ministry of Water Resources, Hohai University, Nanjing, 211100, China.
*E-mail addresses:* xujl@hhu.edu.cn (J.-L. Xu), lvsh@hhu.edu.cn (S.-H. Lyu), wangyunian@hhu.edu.cn (Y.-N. Wang), che-n-ing@hhu.edu.cn (N. Chen), quzhihao@hhu.edu.cn (Z. Qu), cstb@hhu.edu.cn (B. Tang), yebl@nju.edu.cn (B. Ye).

https://doi.org/10.1016/j.patcog.2026.113817
Received 24 February 2026; Received in revised form 2 April 2026; Accepted 18 April 2026
Available online 25 April 2026

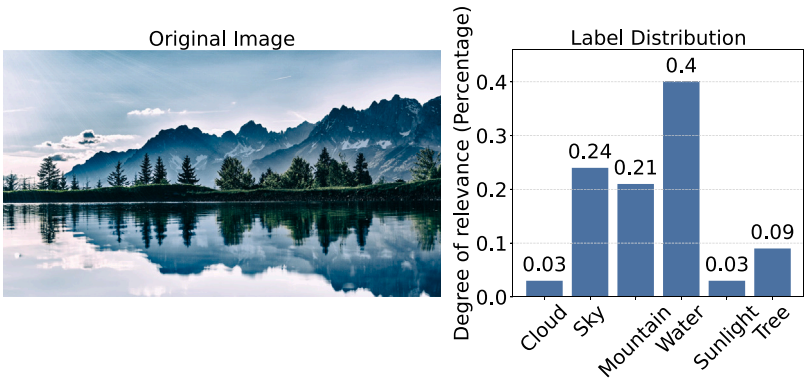



**Fig. 1.** This is an example sample of label distribution learning. On the left is a landscape picture as an input, which contains multiple elements. On the right is the degree of relevance of each element to the picture, which together form the label of this picture: $\boldsymbol{d}_{\text{image}} = \{0.03, 0.24, 0.21, 0.4, 0.03, 0.09\}$. Mountain and water tend to appear together, so the values of these two dimensions are both large. When cloud is present, the values describing the degree of the sky tend to be large. These are all manifestations of the correlations between the labels.

medical care [14,15], and other related fields. Some recent articles innovatively use label distribution to assist in the process of multi-label learning [16].

Whether in the MLL or LDL paradigm, the labels are not independent of each other. Therefore, making use of the correlations between labels is an important task during learning. In LDL, the labels with greater correlations tend to be assigned larger values simultaneously. For example, in the image classification task, the pair of labels "seawater" and "beach" tends to be highly correlated with the image. This situation is less likely to occur between labels with smaller correlations. For instance, in the movie classification task, labels like "love" and "horror", "technology" are basically unlikely to be assigned larger values simultaneously. A more concrete example is illustrated in Fig. 1. Making good use of the correlations between labels can bring improvements in both effectiveness and performance for the algorithm. Current work mainly utilizes the correlation of labels through two methods. One is external regularization such as statistical information [17,18], graph methods [19–21], or other special regularization items [22–24]. This is also the most commonly used method. The other is latent space learning such as low-rank methods [25,26]. The former adds regularization terms to the loss function based on the label correlations to constrain the training of the model; the latter directly alters the label space and learns the mapping from the input space to the new label space, then projects the prediction back to the original label space. These methods predominantly focus on modeling correlations within the output label space, whereas few works concentrate on taking advantage of label correlations to enrich the input feature space.

Deep Forest [27] is a deep learning framework that does not rely on backpropagation. Its implementation draws on the key success factors of deep neural networks: layer-by-layer processing, in-model feature transformation, and sufficient model complexity. Each layer of its cascade structure is an ensemble of multiple forests. Through this structure, features are extracted at each layer and passed to the next layer for processing. Due to its strong feature representation ability and fewer hyperparameters compared to neural networks, DF has achieved success in many tasks such as image classification [28,29]. It has also developed many different variations. Chen et al. [30] utilize high-dimensional feature interactions as the internal representation features, further enhancing its representational ability; Utkin [31] focuses on improving its robustness on small datasets by incorporating imprecise probabilities to handle the uncertainty in class predictions. Additionally, assigning different weights to the trees in the forest [32] has been used to adapt the framework for metric learning. Some studies have also provided theoretical proofs of its performance, for example [33, 34]. However, its exploration in non-single-label learning tasks (such as MLL, LDL) is still at an early stage. A major reason for this is that its characteristic of lacking a loss function makes it unable to effectively utilize traditional methods to leverage the correlation between labels. Yang et al. [35] apply DF to the MLL task, using a feature reuse mechanism to ensure the stability of the training. Ilidio et al. [36] conduct exploration in the task of weak label learning. Although they both utilize the correlations between labels to some extent, this process is implicit rather than explicit.

This paper proposes the ERDF method, which explicitly utilizes the inherent correlation between labels through a feature enhancement mechanism. This mechanism is seamlessly integrated with the cascade structure of DF, taking full advantage of its layer-by-layer in-model feature extraction and transformation capabilities. At the same time, to guarantee the quality of the newly generated features and maintain the overall stability of the training phase, ERDF uses the feature reuse mechanism to identify the poor features and replace them with better ones. The main contributions of this paper can be summarized as follows:

- We propose a new and simple approach to exploring the correlations between labels during the learning process. Unlike most previous methods, it focuses on the input feature space.
- We apply this new mechanism to DF, supplemented by the feature reuse mechanism, and conduct a successful exploration of DF in the field of LDL.
- Experiments on multiple public datasets demonstrate the superiority of our algorithm. The ablation experiments demonstrate the effectiveness of our mechanism.

The remainder of the paper is organized as follows. Section 2 introduces some preliminaries about LDL. Section 3 details our ERDF method, including two designed mechanisms. Section 4 reports the experimental results followed by the conclusion of the paper in Section 5.

## 2. Preliminary

In LDL, $\mathcal{X} \subseteq \mathcal{R}^d$ is a $d$-dimensional feature space and $\mathcal{Y} = \{y_1, y_2, \dots, y_c\}$ is the set of $c$ possible labels. For the $i$th instance $x_i \in \mathcal{X}$, its corresponding label is a label distribution, denoted as $\boldsymbol{d_i} = \{d_i^1, d_i^2, \dots d_i^c\}$. Here $d_i^j$ represents the degree of correlation between $x_i$ and $y_j$. It should be noted that $\boldsymbol{d_i}$ must satisfy two constraints:

- **Non-negativity:** $d_i^j \geq 0, \quad \forall j \in \{1, \dots, c\}$.
- **Normalization:** $\sum_{j=1}^{c} d_i^j = 1$.

Fig. 1 is an example sample. The learning task is to learn a mapping function $f : \mathcal{X} \to D$, which optimizes the performance of unseen instances on some specific metrics. $\mathcal{X}$ here is the input space and $D$ is the space of all possible label distributions.

**Table 1**
Six evaluation metrics for label distribution learning are divided into two types. The arrow indicates the direction for better performance: ↑ signifies that a higher value is better, and ↓ signifies that a lower value is better.

| Type | Measure | Formula |
|---|---|---|
| Distance ↓ | Chebyshev | $Dis_1(\boldsymbol{d},\hat{\boldsymbol{d}}) = \max_j \lvert d_j - \hat{d}_j \rvert$ |
| | Clark | $Dis_2(\boldsymbol{d},\hat{\boldsymbol{d}}) = \sqrt{\sum_{j=1}^{c} \frac{(d_j-\hat{d}_j)^2}{(d_j+\hat{d}_j)^2}}$ |
| | Canberra | $Dis_3(\boldsymbol{d},\hat{\boldsymbol{d}}) = \sum_{j=1}^{c} \frac{\lvert d_j-\hat{d}_j \rvert}{d_j+\hat{d}_j}$ |
| | Kullback–Leibler | $Dis_4(\boldsymbol{d},\hat{\boldsymbol{d}}) = \sum_{j=1}^{c} d_j \ln \frac{d_j}{\hat{d}_j}$ |
| Similarity ↑ | Cosine | $Sim_1(\boldsymbol{d},\hat{\boldsymbol{d}}) = \frac{\sum_{j=1}^{c} d_j \hat{d}_j}{\sqrt{\sum_{j=1}^{c} d_j^2}\sqrt{\sum_{j=1}^{c} \hat{d}_j^2}}$ |
| | Intersection | $Sim_2(\boldsymbol{d},\hat{\boldsymbol{d}}) = \sum_{j=1}^{c} \min(d_j, \hat{d}_j)$ |

Unlike the evaluation metrics for traditional machine learning tasks, such as accuracy for classification and mean square error for regression, metrics for label distribution learning are aimed at describing the relationship between two probability distributions. Widely-used metrics [9], which are divided into two types, are listed in Table 1.

The first type is distance metrics. For this kind of metric, the larger the number, the greater the distance between the two distributions, and the less accurate the prediction result will be. The Chebyshev distance describes the maximum difference in all dimensions between two distributions, capturing the most extreme errors in the prediction. The Clark distance and Canberra distance are the weighted forms of the Euclidean distance and the Manhattan distance, respectively. Under the condition where the differences in a certain dimension are the same, they are more sensitive to the smaller values. The KL divergence measures the information loss that occurs when approximating one distribution with another. It should be noted that it is not symmetric.

The second type is similarity metrics, and it is easy to understand that the larger the number, the better the result. Cosine similarity describes the consistency of the directions of two vectors in a multi-dimensional space. It does not consider the magnitude of the values in each dimension, but only focuses on whether the trends of the vectors are similar. The Intersection similarity describes the area of overlap between two distributions. Contrary to cosine similarity, it mainly focuses on the magnitude of the values in each dimension and is used to reflect the degree of consistency between the two distributions in terms of numerical values.

## 3. The proposed method

### 3.1. Feature enhancement exploiting label correlation

This mechanism is specifically designed to enable DF to take advantage of the inherent correlations among the labels, thereby facilitating better performance in label distribution learning. Given the DF's powerful ability to extract and utilize new features layer by layer, we leverage the correlations among labels to enhance the features at each layer. This mechanism can be divided into two stages: the learning stage and the enhancement stage. The detailed pseudo-code of the feature enhancement algorithm is presented in Algorithm 1.

During the learning phase, it can be divided into three steps. The first step is to extract the relevance. We calculate the Pearson correlation matrix of the label matrix to obtain the correlation relationships between the labels. The value in the $i$th row and $j$th column of the correlation matrix, which represents the correlation coefficient between the $i$th label and the $j$th one, is calculated according to Eq. (1). $N$ is the total number of samples, $d_s^i$ is the description degree of the $i$th label for the $s$th sample, and $\bar{d}^i$ is the mean description degree for the $i$th label over all samples. The resulting matrix $\boldsymbol{C} \in \mathbb{R}^{c\times c}$, where $c$ is the number of classes, serves as an explicit representation of the global label correlations. This matrix is like a social network diagram, revealing positive or negative correlations between those labels with positive or negative numbers ranging from −1 to 1.

$$r_{ij} = \frac{\sum_{s=1}^{N}(d_s^i - \bar{d}^i)(d_s^j - \bar{d}^j)}{\sqrt{\sum_{s=1}^{N}(d_s^i - \bar{d}^i)^2 \sum_{s=1}^{N}(d_s^j - \bar{d}^j)^2}} \tag{1}$$

The second step is to extract the patterns. The information in the correlation matrix is vast and may be redundant. We use principal component analysis (PCA) to extract $k$ of the most representative relationship patterns from it. These patterns constitute the matrix $\boldsymbol{V}$, which consists of $k$ eigenvectors as shown in Eq. (2). Each pattern vector is the same dimensions as the label distribution of each sample, corresponding to one dimension of the enhanced feature later.

$$\boldsymbol{V} = \mathrm{PCA}(\boldsymbol{C}, k) = [\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_k] \tag{2}$$

The third step is to calculate the target relationship values and train the feature enhancers. For the $j$th relationship pattern vector $\boldsymbol{v}_j$, we perform a dot product between the global label matrix $\boldsymbol{D}$ and $\boldsymbol{v}_j$ to obtain the target vector $\boldsymbol{s}_j$, as shown in Eq. (3), where $\boldsymbol{D} \in \mathbb{R}^{N\times c}$ is the label matrix of all training samples. The resulting vector $\boldsymbol{s}_j \in \mathbb{R}^{N\times 1}$ represents the degree of similarity between all samples and the $j$th relationship pattern. Subsequently, we train the $j$th feature enhancer to learn the mapping from the input features $\boldsymbol{X}$ to this target vector $\boldsymbol{s}_j$.

$$\boldsymbol{s}_j = \boldsymbol{D}\boldsymbol{v}_j, \quad j \in \{1, \ldots, k\} \tag{3}$$

During the subsequent enhancement stage, we utilize all $k$ feature enhancers to generate k-dimensional enhanced features for each sample. These enhanced features are then concatenated with the original features to ultimately obtain the final feature vector used for learning.

**Algorithm 1** Feature enhancement exploiting label correlation

**Input:** Feature matrix $\boldsymbol{X} \in \mathbb{R}^{N\times D}$; Label matrix $\boldsymbol{D} \in \mathbb{R}^{N\times c}$; Number of relational features to generate $k$.
**Output:** A set of trained enhancers $\mathcal{E} = \{e_1, \ldots, e_k\}$; Enhanced relational feature matrix $\boldsymbol{E} \in \mathbb{R}^{N\times k}$.

**procedure** Fit($\boldsymbol{X}, \boldsymbol{D}, k$) *// Learning Phase*
  *// Compute label correlation matrix according to Eq. (1)*
  $\boldsymbol{C} \leftarrow$ CalculateCorrelationMatrix($\boldsymbol{D}$)
  $\boldsymbol{V} \leftarrow$ PCA($\boldsymbol{C}, k$) *// Extract k relationship vectors according to Eq. (2)*
  $\mathcal{E} \leftarrow \emptyset$ *// Initialize empty enhancer set*
  **for** $j = 1$ to $k$ **do**
    $\boldsymbol{s}_j \leftarrow \boldsymbol{D}\boldsymbol{v}_j$ *// Ideal scores calculated according to Eq. (3)*;
    $e_j \leftarrow$ RandomForestRegressor($\boldsymbol{X}, \boldsymbol{s}_j$); $\mathcal{E} \leftarrow \mathcal{E} \cup \{e_j\}$
  **end for**
  **return** $\mathcal{E}$
**end procedure**

**procedure** Transform($X, \mathcal{E}$) *// Enhancement Phase*
  $k \leftarrow |\mathcal{E}|$; $\boldsymbol{E} \leftarrow \boldsymbol{0}^{N\times k}$
  **for** $j = 1$ to $k$ **do**
    $\hat{\boldsymbol{s}}_j \leftarrow e_j$.predict($\boldsymbol{X}$)
    SetColumn($\boldsymbol{E}, j, \hat{\boldsymbol{s}}_j$) *// Set the j-th column*
  **end for**
  **return** $\boldsymbol{E}$
**end procedure**

Let us consider the movie classification task mentioned earlier. Assume that the label set $\mathcal{Y} = \{\text{action}, \text{comedy}, \text{romance}, \text{horror}, \text{science fiction}\}$. We first calculate the correlation matrix. Obviously, action and horror are likely to be positively correlated, while comedy and horror are likely to be negatively correlated. Through principal component analysis, we may obtain two distinct relationship patterns. The first relationship pattern represents the meaning of "blockbuster"; in this pattern, the values of action and science fiction are relatively large. The second relationship pattern represents the meaning of "youth"; in this pattern, the value of romance is relatively large. Subsequently, for each relationship pattern, we calculate the similarity between the

label distribution of all samples and the target relationship pattern, and then learn an enhancer to map from the original feature vector to the relationship pattern values, which is used to enhance the original features.

DF can extract new feature vectors to perform in-model feature transform layer by layer. In this mechanism, new features are enhanced at each layer, which is perfectly adaptive to the layer-by-layer feature transformation process in DF. This is a key factor for the success of this mechanism. It should be noted that the reason for training the feature enhancer instead of directly using the calculated relationship values as new features is that during generalization, the true label distribution of the samples is invisible.

### 3.2. Measure-aware feature reuse

Measure-aware feature reuse [35] is a mechanism that ensures the quality of features extracted by the cascade structure of DF and promotes the stability of layer-by-layer training. ERDF not only extracts new features like standard DF, but also enhances the features through the enhancers layer by layer. This results in more new features at each layer, which leads to a greater need for the feature reuse mechanism to ensure the stability of training and to perform reuse operations for new features of low quality. This is achieved by identifying samples with degraded performance at the current layer and replacing their newly generated features with the more reliable features obtained from the preceding layer.

After the training on the $l$th layer is completed ($l > 1$), we will obtain the validation result matrix $\boldsymbol{H}_l$ for all samples. $\boldsymbol{H}_l$ is viewed as the first part of the new features, and we will concatenate it with the original feature, just as DF does. Then, ERDF will use the feature enhancement mechanism described in Section 3.1 to enhance the combined features, obtaining the second part of the new features $\boldsymbol{E}_l$. By concatenating $\boldsymbol{H}_l$ and $\boldsymbol{E}_l$ according to Eq. (4), we get all the new features $\boldsymbol{F}_{new}^{(l)}$ for this layer. At this time, we cannot guarantee that all the features in $\boldsymbol{F}_{new}^{(l)}$ are of high quality and beneficial for the subsequent training. If we directly pass it to the subsequent layers, it may cause instability in the training process. The noise in the new features will gradually expand as the layer deepens, causing a disastrous impact on the training.

$$\mathbf{F}_{\text{new}}^{(l)} = [\mathbf{H}_l, \mathbf{E}_l] \tag{4}$$

The measure-aware feature reuse mechanism comes into play in this scenario. First, based on actual requirements, we can select a measurement metric $\mathcal{M}$. By comparing the prediction results of the current layer ($\boldsymbol{H}_l$), with those of the previous layer ($\boldsymbol{H}_{l-1}$), we can quickly identify the sample set $S$ that has deteriorated in performance on metric $\mathcal{M}$ after the training of the $l$th layer. Subsequently, using a certain threshold $\tau$ as the boundary, we determine the specific subset of samples $S_r \subset S$ requiring to be reused, as formulated in Eq. (5):

$$S_r = \left\{ s \in \{1, \ldots, N\} \;\middle|\; \mathbb{I}(m_{l,s} > m_{l-1,s}) \cdot \mathbb{I}(m_{l,s} > \tau) = 1 \right\} \tag{5}$$

where $\mathbb{I}(\cdot)$ is the indicator function whose value is 1 when the expression is true. $m_{l,s} = \mathcal{M}(\boldsymbol{d}_s, \boldsymbol{h}_{l,s})$ denotes the calculated metric value obtained by comparing the validation prediction $\mathbf{h}_{l,s}$ of the $s$th sample at layer $l$ with its corresponding ground-truth label distribution $\boldsymbol{d}_s$. It is worth noting that the inequalities in Eq. (5) assume that $\mathcal{M}$ is a distance metric (e.g., KL divergence) where a lower value indicates better performance. If a similarity metric (e.g., cosine similarity) is used, where a higher value is better, the direction of these inequalities should be reversed.

ERDF then applies the reuse operation for the new features of samples in $S_r$, which are obtained through this layer's training. Concretely, ERDF replaces the rows of $\boldsymbol{F}_{new}^{(l)}$ which represent the samples in $S_r$, with the corresponding rows in matrix $\boldsymbol{G}_{l-1}$ which are the final new features of these samples in the previous layer. Now, new features in $\boldsymbol{F}_{new}^{(l)}$ are more reliable, and we record the newly obtained matrix as $\boldsymbol{G}_l$, which represents the final new features of all samples after this layer's training. Then, we concatenate $\boldsymbol{G}_l$ with the original features and pass it to the next layer for training.

As for the selection of the threshold $\tau$, we simply use the mean score of all the samples in $S$ which is calculated on $\mathcal{M}$ as the boundary for elimination. We opt for partial rather than complete elimination because the latter is an overly greedy approach, assuming current status guarantees the best. In practice, a deterioration in a specific metric does not always imply failure. Novel feature perspectives may induce temporary fluctuations which is sometimes a normal part of the learning dynamics.

### 3.3. The framework

The overall framework of ERDF is shown in Fig. 2. It mainly adds a feature enhancement module (described in Section 3.1) and a feature reuse module (described in Section 3.2) on the basis of traditional DF.

Before starting training, an initial feature enhancement is conducted, and the obtained enhancer $\mathcal{E}_0$ is stored. Then, the feature vector is input into the cascaded forest for training. Each layer contains two types of forests, random forest (RF) and extremely random forest (ERF), to ensure diversity.

After the training of the $l$th layer, the validation prediction result $\boldsymbol{H}_l$ of all samples is given as the first part of the new feature, and it is also used to evaluate the current layer's training performance. Then $\boldsymbol{H}_l$ is concatenated with the original feature, input into the feature enhancement module for enhancement. The second part of the new feature $\boldsymbol{E}_l$ and the current layer's enhancers $\mathcal{E}_l$ are obtained. The two parts of the new features are concatenated to form $\boldsymbol{F}_{new}^{(l)}$, which is input into the feature reuse module for reuse. After the reuse operation is finished, we can get this layer's final new feature matrix $\boldsymbol{G}_l$ and the elimination threshold $\tau_l$ of this layer.

Note that the feature reuse module of the first layer cannot perform a reuse operation due to the lack of former information, and will directly record $F_{new}^{(0)}$ as the final new feature matrix $\boldsymbol{G}_0$ for subsequent use. $\boldsymbol{G}_l$ will be concatenated with the original feature, passed to the next layer for training, and also passed to the feature reuse module of the next layer for judgment on whether the performance has declined. The number of layers of the cascaded forest is self-adapting. Training will stop automatically when there is no significant improvement in performance. It should be noted that we place the feature reuse mechanism after the feature enhancement mechanism. This fixed order cannot be changed. If reuse is performed before enhancement, only the first part of the new features $\boldsymbol{H}_l$ will be reused, and it cannot ensure that the quality of the second part of the new features $\boldsymbol{E}_i$ remains stable.

For unseen samples, the initialization enhancement is first performed with $\mathcal{E}_0$, and then input into the cascade for prediction. Similar to the training process, after obtaining the validation prediction result $\boldsymbol{H}_l$ of the $l$th layer, it will be processed by two modules. The difference is that in the feature enhancement module, the trained enhancer is directly used for enhancement, and in the feature reuse module, the threshold stored during the training stage is directly used to screen the reused samples. When the layer with the highest validation performance during the training process gives the prediction, the transmission is terminated, and it is viewed as the final prediction.

## 4. Experiments

In this section, we conduct experiments on our ERDF algorithm on multiple public LDL datasets to validate its performance. We compare it with several commonly used LDL algorithms and demonstrate the superiority of ERDF. We also conduct ablation experiments on the two mechanisms of ERDF to demonstrate their necessity.

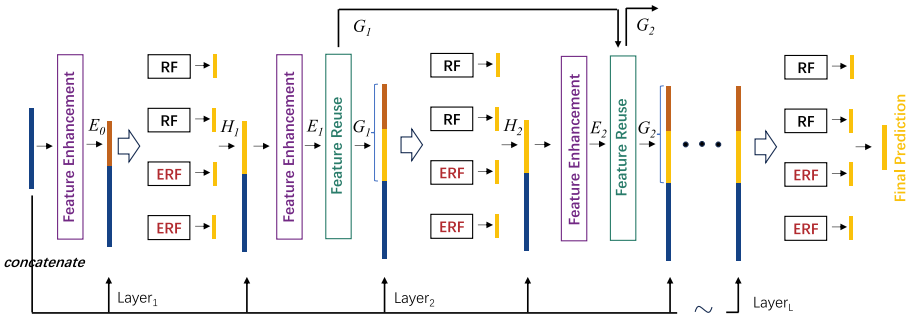



**Fig. 2.** This figure clearly presents the overall framework of ERDF. Each layer is composed of RF and ERF. After the training of the $l$th layer, the first part of the new features $\boldsymbol{H}_l$ is obtained. These new features are composed of the validation prediction results of all forests in this layer. Then, the feature enhancement mechanism is conducted to obtain the second part of the new features $\boldsymbol{E}_l$. These two parts are concatenated to form $\boldsymbol{F}_{new}^{(l)}$, which is input into the feature reuse module. After reuse according to $\boldsymbol{G}_{l-1}$, the final new features $\boldsymbol{G}_l$ of this layer are obtained and concatenated with the original features before being passed to the next layer.

**Table 2**
Information about the five datasets we used in our experiments.

| Dataset | Samples | Features | Classes |
|---|---|---|---|
| Movie | 7755 | 1869 | 5 |
| Natural_Scene | 2000 | 294 | 9 |
| SBU_3DFE | 2500 | 243 | 6 |
| emotion6 | 1980 | 168 | 7 |
| SCUT_FBP | 1500 | 300 | 5 |

### 4.1. Dataset and configuration

Benchmark and ablation experiments are conducted on five commonly used public datasets. Table 2 shows the detailed information with sample sizes ranging from 1500 to 7755, feature numbers ranging from 168 to 1869, and the number of classes (i.e., the dimension of label distribution) ranging from 5 to 9. In the experiments, we randomly divide the datasets into training sets and test sets in the ratio of 8:2. To reduce the randomness of the experiments, for each dataset, we split the samples using three different random seeds, and finally take the mean and standard deviation of the results as the final reported result.

We use the six metrics shown in Table 1 to conduct a comprehensive evaluation of the model from both the distance and similarity perspectives. We compare ERDF with four commonly used LDL algorithms. They are AA-KNN [9], StructRF [37], LDL-SCL [38], and RG4LDL [39] respectively. AA-KNN predicts the label distribution of a new sample by finding its $k$ nearest neighbors in the feature space and averaging their label distributions. StructRF extends the traditional random forest framework to the LDL task. It treats the label distribution of each sample as a whole, enabling the model to directly predict the complete label distribution at the leaf nodes instead of making separate predictions for each dimension. LDL-SCL is an advanced algorithm that utilizes subspace clustering to leverage label correlations. It first clusters instances based on their labels and then enhances the original features with a local correlation vector representing the influence of these clusters. RG4LDL employs a restricted Boltzmann machine (RBM) to extract low-dimensional latent representations from high-dimensional features, and then uses the BFGS algorithm for supervised optimization to predict the final label distributions.

Hyperparameters of ERDF are set as follows. We use a decision tree similar to StructTree [37] as the base learner which considers the label distribution of each sample as a whole. For each unseen sample that falls under a certain leaf node, the mean of the label distributions of all samples at that leaf node is taken as the final prediction result. The difference is that we use the KL divergence as the splitting criterion. The maximum depth of each tree is set to 10, and the minimum sample number of leaf nodes is set to 2. Each forest contains 100 decision trees, and each layer of the cascade forest consists of two random forests and two completely random forests. The maximum number of layers in the cascade forest is 10, and the early stopping tolerance is set to 1, which means that the training will be terminated in advance if there is no performance improvement for two consecutive layers before reaching the maximum number of layers. In the feature reuse mechanism, we use the KL divergence for performance measurement. In the feature enhancement mechanism, $k$ is set to 5, which means extracting five relationship patterns from the relationship matrix to generate 5-dimensional enhanced features for each instance.

The parameters of the comparison algorithms are listed as follows: in AA-KNN, $k$ is set to 5, and the Minkowski distance metric is used to find the $k$ nearest neighbor of an instance. In StructRF, the number of trees and the maximum depth are 100 and 10, respectively, which are consistent with our ERDF. In LDL-SCL, for each dataset, $\lambda_1$, $\lambda_2$, and $\lambda_3$ are tuned from $\{10^{-1}, 10^{-2}, 10^{-3}\}$, and the number of clusters $c$ is tuned in the set $\{5, 10\}$. In RG4LDL, the number of hidden units is selected from $\{10, 20, 50, 100, 200, 500, 800\}$, the maximum number of iterations is 10, and the learning rate is tuned from $\{10^{-2}, 10^{-3}\}$.

### 4.2. Benchmark experiment

As shown in Table 3, our proposed ERDF achieves state-of-the-art performance with a top average rank of 1.20. The foundational LDL algorithm, AA-KNN, which does not explicitly model label correlations, ranks the lowest. This outcome empirically validates the central premise that leveraging label correlations is critical for success in LDL tasks. StructRF performs competitively (average rank 2.27) by implicitly capturing correlations among labels. Its efficacy can be attributed to the non-linear modeling power of tree-based ensembles. Nevertheless, its performance is surpassed by ERDF, which benefits from a more potent and explicit mechanism for exploiting label correlations within a more powerful ensemble architecture. An insightful comparison can be drawn with LDL-SCL. Similar to ERDF, it explicitly enhances features using label correlations. However, LDL-SCL exhibits high sensitivity to its hyperparameters, requiring dataset-specific tuning to reach its optimal performance. As a new proposed method based on neural networks in recent years, RG4LDL performs similarly to LDL-SCL. Although it is found in experiments that sometimes it is not very sensitive to hyperparameters, a certain degree of parameter tuning is still necessary to achieve optimal performance. In contrast, ERDF demonstrates remarkable robustness, achieving superior results with a consistent set of hyperparameters across all datasets, which highlights the stability and practical advantages of our proposed architecture.

**Table 3**
This table presents the results of ERDF and four common LDL algorithms on five datasets. All results are reported as mean ± standard deviation over three random splits. The arrow beside the metric indicates the direction for better performance. The optimal value of each row is displayed in bold. We use • (or ∘) to indicate that ERDF is significantly better (or worse) than the corresponding method.

| Dataset | Metric | AA-KNN | StructRF | LDL-SCL | RG4LDL | ERDF(ours) |
|---|---|---|---|---|---|---|
| Movie | KL div ↓ | 0.1133 ± 0.0010 • | 0.0926 ± 0.0025 • | 0.1024 ± 0.0024 • | 0.1024 ± 0.0015 • | **0.0600 ± 0.0014** |
| | Chebyshev ↓ | 0.1227 ± 0.0004 • | 0.1108 ± 0.0010 • | 0.1189 ± 0.0014 • | 0.1170 ± 0.0008 • | **0.0939 ± 0.0011** |
| | Clark ↓ | 0.5463 ± 0.0037 • | 0.5035 ± 0.0039 • | 0.5309 ± 0.0105 • | 0.5289 ± 0.0073 • | **0.4289 ± 0.0070** |
| | Canberra ↓ | 1.0462 ± 0.0050 • | 0.9597 ± 0.0069 • | 1.0143 ± 0.0170 • | 1.0152 ± 0.0126 • | **0.8026 ± 0.0118** |
| | Cosine ↑ | 0.9251 ± 0.0006 • | 0.9391 ± 0.0017 • | 0.9321 ± 0.0011 • | 0.9326 ± 0.0006 • | **0.9604 ± 0.0006** |
| | Intersection ↑ | 0.8246 ± 0.0004 • | 0.8424 ± 0.0015 • | 0.8317 ± 0.0020 • | 0.8317 ± 0.0014 • | **0.8715 ± 0.0013** |
| Natural | KL div ↓ | 1.0357 ± 0.0575 • | 0.6689 ± 0.0295 • | 0.8283 ± 0.0118 • | 0.7634 ± 0.0143 • | **0.5216 ± 0.0155** |
| | Chebyshev ↓ | 0.3134 ± 0.0100 • | 0.2794 ± 0.0172 • | 0.3329 ± 0.0083 • | 0.3106 ± 0.0138 • | **0.2221 ± 0.0049** |
| | Clark ↓ | **1.9085 ± 0.0214** ∘ | 2.3465 ± 0.0175 | 2.4682 ± 0.0116 • | 2.4124 ± 0.0449 | 2.3595 ± 0.0209 |
| | Canberra ↓ | **4.5686 ± 0.0701** ∘ | 6.2559 ± 0.0761 ∘ | 6.7833 ± 0.0468 • | 6.6179 ± 0.1570 • | 6.3066 ± 0.0777 |
| | Cosine ↑ | 0.7150 ± 0.0022 • | 0.7820 ± 0.0089 • | 0.7284 ± 0.0036 • | 0.7401 ± 0.0054 • | **0.8343 ± 0.0040** |
| | Intersection ↑ | 0.5621 ± 0.0058 • | 0.5927 ± 0.0151 • | 0.4994 ± 0.0047 • | 0.5646 ± 0.0209 • | **0.6704 ± 0.0062** |
| SBU | KL div ↓ | 0.0802 ± 0.0011 • | 0.0573 ± 0.0004 • | 0.0634 ± 0.0005 • | 0.0679 ± 0.0004 • | **0.0414 ± 0.0005** |
| | Chebyshev ↓ | 0.1273 ± 0.0022 • | 0.1104 ± 0.0013 • | 0.1199 ± 0.0015 • | 0.1199 ± 0.0018 • | **0.0865 ± 0.0013** |
| | Clark ↓ | 0.4012 ± 0.0017 • | 0.3444 ± 0.0009 • | 0.3695 ± 0.0017 • | 0.3793 ± 0.0015 • | **0.2746 ± 0.0020** |
| | Canberra ↓ | 0.8309 ± 0.0039 • | 0.7276 ± 0.0060 • | 0.7960 ± 0.0043 • | 0.7997 ± 0.0050 • | **0.5772 ± 0.0044** |
| | Cosine ↑ | 0.9219 ± 0.0011 • | 0.9432 ± 0.0006 • | 0.9376 ± 0.0006 • | 0.9337 ± 0.0008 • | **0.9587 ± 0.0006** |
| | Intersection ↑ | 0.8487 ± 0.0010 • | 0.8691 ± 0.0007 • | 0.8575 ± 0.0007 • | 0.8562 ± 0.0010 • | **0.8968 ± 0.0007** |
| SCUT | KL div ↓ | 0.5241 ± 0.0049 • | 0.3337 ± 0.0094 • | 0.5647 ± 0.0056 • | 0.3707 ± 0.0164 • | **0.2949 ± 0.0134** |
| | Chebyshev ↓ | 0.2537 ± 0.0035 • | 0.2303 ± 0.0024 • | 0.3418 ± 0.0073 • | 0.2520 ± 0.0042 • | **0.2092 ± 0.0049** |
| | Clark ↓ | **1.3043 ± 0.0116** ∘ | 1.3675 ± 0.0032 • | 1.4561 ± 0.0050 • | 1.3781 ± 0.0034 • | 1.3523 ± 0.0027 |
| | Canberra ↓ | **2.3985 ± 0.0411** | 2.5220 ± 0.0037 • | 2.8076 ± 0.0234 • | 2.5591 ± 0.0140 • | 2.4651 ± 0.0122 |
| | Cosine ↑ | 0.8294 ± 0.0045 • | 0.8624 ± 0.0045 • | 0.7598 ± 0.0053 • | 0.8433 ± 0.0056 • | **0.8782 ± 0.0063** |
| | Intersection ↑ | 0.6942 ± 0.0058 • | 0.7210 ± 0.0025 • | 0.5750 ± 0.0064 • | 0.6959 ± 0.0049 • | **0.7552 ± 0.0054** |
| emotion6 | KL div ↓ | 0.9038 ± 0.0264 • | 0.5893 ± 0.0093 • | 0.5862 ± 0.0230 • | 0.5989 ± 0.0254 • | **0.5111 ± 0.0252** |
| | Chebyshev ↓ | 0.3326 ± 0.0113 • | 0.3137 ± 0.0051 • | 0.3108 ± 0.0090 • | 0.3206 ± 0.0093 • | **0.2869 ± 0.0056** |
| | Clark ↓ | 1.7087 ± 0.0146 • | 1.6569 ± 0.0210 • | 1.6503 ± 0.0221 | 1.6529 ± 0.0219 • | **1.6495 ± 0.0230** |
| | Canberra ↓ | 3.8728 ± 0.0475 • | 3.7285 ± 0.0502 | 3.7000 ± 0.0510 | 3.7203 ± 0.0535 | **3.6858 ± 0.0684** |
| | Cosine ↑ | 0.6561 ± 0.0120 • | 0.7125 ± 0.0026 • | 0.7152 ± 0.0079 • | 0.7072 ± 0.0106 • | **0.7496 ± 0.0078** |
| | Intersection ↑ | 0.5534 ± 0.0079 • | 0.5793 ± 0.0042 • | 0.5833 ± 0.0072 • | 0.5735 ± 0.0101 • | **0.6238 ± 0.0089** |
| Avg. Rank | | 4.27 | 2.33 | 3.67 | 3.53 | 1.20 |

### 4.3. Ablation study

To deeply investigate the internal working mechanism of our proposed method, we conduct extensive ablation studies to evaluate the distinct contributions of the two core components of ERDF. Specifically, we compare the full ERDF model with (i) ERDF without the feature enhancement module (w/o fe), (ii) ERDF without the feature reuse module (w/o fr), and (iii) the original DF without either module (w/o fe & fr). The results are summarized in Table 4. Across all datasets and most evaluation metrics, the full ERDF consistently achieves the best performance, demonstrating the complementary benefits of the two modules. Relative to w/o fe & fr, w/o fe yields noticeable improvements, with average ranks of 2.12. However, its performance ceiling is limited due to the failure to exploit inter-label correlations explicitly. In stark contrast, w/o fr exhibits significant instability. Although it outperforms the baseline from the overall ranking perspective (average rank 3.27), on certain datasets such as SBU, its performance even drops below that of the baseline w/o fe & fr. This degradation is attributed to the uncontrolled diffusion of noise within the newly generated features, which leads to severe overfitting. This critical phenomenon will be further analyzed in the subsequent subsection. Nevertheless, the superior performance of the full ERDF model vindicates the critical value of the feature enhancement mechanism. It demonstrates that to fully unlock the potential of label correlations without suffering from noise, feature enhancement must be inextricably coupled with feature reuse. Through ablation study it can be seen that the dual-mechanism strategy proposed in this paper offers a robust paradigm for leveraging label correlations within deep forest framework.

### 4.4. Layer-wise dynamics and mechanism analysis

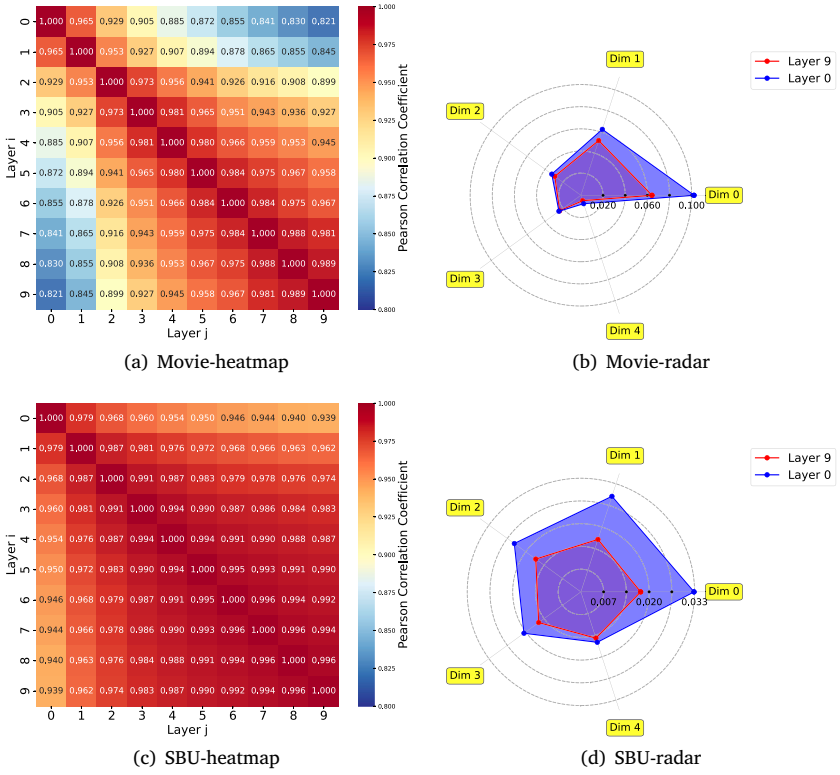
While the foregoing ablation study quantitatively confirms the complementary nature of the dual-mechanism design, in this section, we leverage a series of visualization experiments on representative datasets to deeply investigate the intrinsic relationship between the two mechanisms proposed in ERDF and the cascade structure of DF. This analysis explicitly elucidates the distinct roles and dynamic evolution of these mechanisms throughout the layer-wise structure.

*Interplay between feature enhancement and layer structure.* To investigate the dynamic evolution of the feature enhancement mechanism throughout the cascading process, we conduct a joint analysis combining inter-layer enhanced feature correlation heatmaps (Figs. 3(a) and 3(c)) and enhanced feature error radar charts (Figs. 3(b) and 3(d)). It is crucial to clarify first that, since the label distribution of the training samples is fixed, the $k$ ideal values of $k$ latent relationship patterns (i.e., the regression targets for the enhancers) extracted via PCA on the label matrix and then calculated by dot product remain invariant across all layers. It is also the reason why the correlation coefficients of the enhanced features between layers are universally high, consistently exceeding 0.8. As the layer depth increases, it is not the regression targets that evolve, but rather the feature representations input into the enhancers. Each layer of DF extracts new features with enhanced discriminability. These are concatenated with the original features to serve as input for the enhancer. This facilitates the learning of an optimal mapping from the feature space to the latent label space, allowing the generated enhanced features to progressively approximate the ideal scores layer by layer.

Experimental results clearly reveal that feature enhancement acts as two distinct roles during the evolutionary process, contingent upon the specific dataset characteristics. **(1) Structural Reconstruction and Directional Adjustment.** Taking the Movie dataset as an instance, Fig. 3(a) shows that the correlation coefficient between Layer 0 and Layer 9 is only 0.821. This relatively lower inter-layer correlation implies that the model has undergone significant adjustments to the internal

**Table 4**
This table presents the results of the ablation experiments. All results are reported as mean ± standard deviation over three random splits. The arrow beside the metric indicates the direction for better performance. The optimal value of each row is displayed in bold. We use • (or ∘) to indicate that ERDF is significantly better (or worse) than the corresponding method.

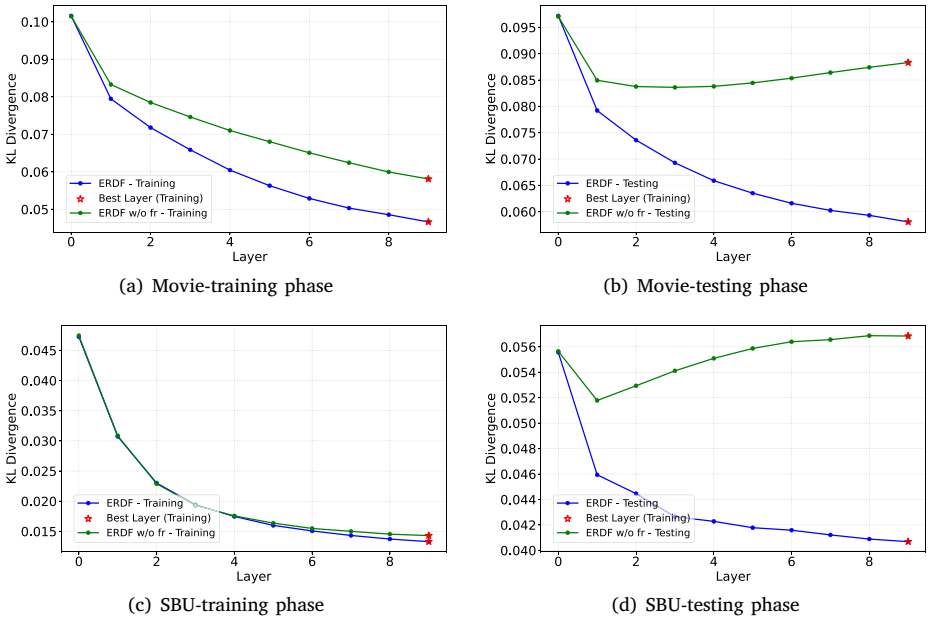
| Dataset | Metric | w/o fe & fr | w/o fe | w/o fr | ERDF (ours) |
|---|---|---|---|---|---|
| Movie | KL div ↓ | 0.0921 ± 0.0018 • | 0.0698 ± 0.0011 • | 0.0898 ± 0.0034 • | **0.0600 ± 0.0014** |
| | Chebyshev ↓ | 0.1120 ± 0.0010 • | 0.1018 ± 0.0007 • | 0.1082 ± 0.0020 • | **0.0939 ± 0.0011** |
| | Clark ↓ | 0.5129 ± 0.0070 • | 0.4640 ± 0.0053 • | 0.4917 ± 0.0074 • | **0.4289 ± 0.0070** |
| | Canberra ↓ | 0.9745 ± 0.0111 • | 0.8712 ± 0.0090 • | 0.9378 ± 0.0144 • | **0.8026 ± 0.0118** |
| | Cosine ↑ | 0.9395 ± 0.0009 • | 0.9537 ± 0.0002 • | 0.9408 ± 0.0024 • | **0.9604 ± 0.0006** |
| | Intersection ↑ | 0.8406 ± 0.0011 • | 0.8594 ± 0.0006 • | 0.8462 ± 0.0030 • | **0.8715 ± 0.0013** |
| Natural | KL div ↓ | 0.5912 ± 0.0139 • | 0.5617 ± 0.0139 • | 0.5866 ± 0.0150 • | **0.5216 ± 0.0155** |
| | Chebyshev ↓ | 0.2535 ± 0.0079 • | 0.2436 ± 0.0074 • | 0.2365 ± 0.0040 • | **0.2221 ± 0.0049** |
| | Clark ↓ | 2.4071 ± 0.0143 • | 2.4000 ± 0.0169 • | 2.3696 ± 0.0161 | **2.3595 ± 0.0209** |
| | Canberra ↓ | 6.4948 ± 0.0622 • | 6.4612 ± 0.0705 • | 6.3594 ± 0.0587 | **6.3066 ± 0.0777** |
| | Cosine ↑ | 0.8139 ± 0.0036 • | 0.8247 ± 0.0038 • | 0.8139 ± 0.0044 • | **0.8343 ± 0.0040** |
| | Intersection ↑ | 0.6194 ± 0.0074 • | 0.6332 ± 0.0073 • | 0.6561 ± 0.0053 • | **0.6704 ± 0.0062** |
| SBU | KL div ↓ | 0.0546 ± 0.0000 • | 0.0477 ± 0.0004 • | 0.0581 ± 0.0014 • | **0.0414 ± 0.0005** |
| | Chebyshev ↓ | 0.1083 ± 0.0013 • | 0.1008 ± 0.0014 • | 0.1019 ± 0.0004 • | **0.0865 ± 0.0013** |
| | Clark ↓ | 0.3343 ± 0.0004 • | 0.3123 ± 0.0002 • | 0.3138 ± 0.0007 • | **0.2746 ± 0.0020** |
| | Canberra ↓ | 0.7149 ± 0.0021 • | 0.6665 ± 0.0019 • | 0.6482 ± 0.0026 • | **0.5772 ± 0.0044** |
| | Cosine ↑ | 0.9459 ± 0.0001 • | 0.9525 ± 0.0005 • | 0.9414 ± 0.0013 • | **0.9587 ± 0.0006** |
| | Intersection ↑ | 0.8719 ± 0.0001 • | 0.8807 ± 0.0003 • | 0.8810 ± 0.0002 • | **0.8968 ± 0.0007** |
| SCUT | KL div ↓ | 0.3474 ± 0.0141 • | 0.3349 ± 0.0141 • | 0.4350 ± 0.0321 • | **0.2949 ± 0.0134** |
| | Chebyshev ↓ | 0.2401 ± 0.0048 • | 0.2382 ± 0.0044 • | 0.2455 ± 0.0098 • | **0.2092 ± 0.0049** |
| | Clark ↓ | 1.3681 ± 0.0034 • | 1.3639 ± 0.0040 | 1.4051 ± 0.0128 • | **1.3523 ± 0.0027** |
| | Canberra ↓ | 2.5261 ± 0.0123 • | 2.5128 ± 0.0136 | 2.6361 ± 0.0446 • | **2.4651 ± 0.0122** |
| | Cosine ↑ | 0.8548 ± 0.0064 • | 0.8590 ± 0.0063 • | 0.8263 ± 0.0140 • | **0.8782 ± 0.0063** |
| | Intersection ↑ | 0.7095 ± 0.0051 • | 0.7135 ± 0.0048 • | 0.7065 ± 0.0122 • | **0.7552 ± 0.0054** |
| emotion6 | KL div ↓ | 0.5700 ± 0.0205 • | 0.5338 ± 0.0173 | 0.7410 ± 0.0335 • | **0.5111 ± 0.0252** |
| | Chebyshev ↓ | 0.3121 ± 0.0074 • | 0.3011 ± 0.0065 • | 0.3355 ± 0.0029 • | **0.2869 ± 0.0056** |
| | Clark ↓ | 1.6457 ± 0.0234 ∘ | **1.6439 ± 0.0237** | 1.7267 ± 0.0226 • | 1.6495 ± 0.0230 |
| | Canberra ↓ | 3.6912 ± 0.0552 | **3.6754 ± 0.0524** | 3.9466 ± 0.0586 • | 3.6858 ± 0.0684 |
| | Cosine ↑ | 0.7234 ± 0.0081 • | 0.7418 ± 0.0065 • | 0.6560 ± 0.0067 • | **0.7496 ± 0.0078** |
| | Intersection ↑ | 0.5840 ± 0.0071 • | 0.5996 ± 0.0052 • | 0.5615 ± 0.0061 • | **0.6238 ± 0.0089** |
| Avg. Rank | | 3.50 | 2.13 | 3.27 | 1.10 |




(a) Movie-heatmap (b) Movie-radar

(c) SBU-heatmap (d) SBU-radar

**Fig. 3.** Visualization of the feature enhancement dynamics across layers on Movie and SBU. Figs. 3(a) and 3(c) are heatmaps showing the enhanced feature correlation between layers. Figs. 3(b) and 3(d) are radar charts, showing the average error between the predicted values of the enhancers and the ideal values in each dimension.




(a) Movie-training phase (b) Movie-testing phase

(c) SBU-training phase (d) SBU-testing phase

**Fig. 4.** Comparison of KL divergence trajectory between ERDF (blue) and ERDF w/o fr (green) on dataset Movie and SBU. The left column is the training phase, and the right is the testing phase. The red star indicates the optimal layer during the training phase, which is the layer that gives the final prediction in the testing phase. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

structure of the enhanced features as depth increased. Combining this with the radar chart (Fig. 3(b)), it is evident that while the error decreases across all dimensions, this improvement is not significant. This suggests that in complex semantic scenarios, the enhancement mechanism primarily acts as an "Explorer", seeking superior representations by continuously reconstructing the direction of the feature space. **(2) Magnitude Calibration and Numerical Refinement.** Conversely, on the SBU dataset, Fig. 3(c) displays a correlation coefficient of 0.939 between Layer 0 and Layer 9, demonstrating that the distributional trends of the enhanced features maintain high stability across layers. However, the radar chart in Fig. 3(d) reveals a remarkable error contraction (the red area is significantly smaller than the blue area), with the reduction in the most significant dimension (Dim 0) reaching up to fifty percent. This phenomenon indicates that the initial layers have already captured the correct semantic trends, and the enhancement mechanism in subsequent layers primarily functions as a "Calibrator". It focuses on fine-grained magnitude correction of feature values, substantially reducing prediction error while preserving structural stability.

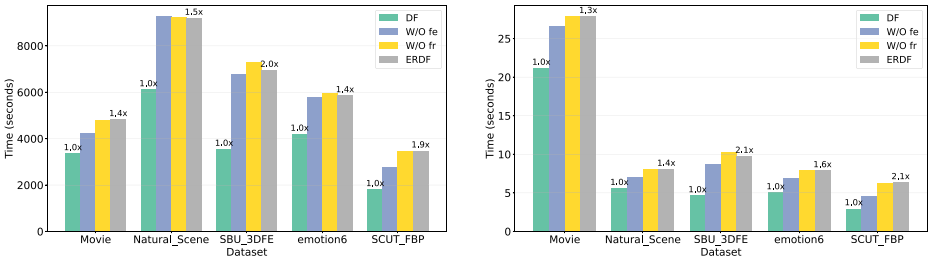
Consequently, the feature enhancement mechanism is not merely a static procedure but rather a dynamic optimization process that adaptively performs layer-wise structural reconstruction or numerical calibration depending on the specific statistical characteristics of the dataset.

*Interplay between feature reuse and layer structure.* To validate the robustness of the feature reuse mechanism throughout the cascading process, we compared the layer-wise performance evolution of the complete ERDF model against its variant without the feature reuse mechanism (ERDF w/o fr) on Movie and SBU. As illustrated in Fig. 4, we recorded the trajectory of KL divergence during both the training and testing phases. In the training phase (Figs. 4(a) and 4(c)), both curves exhibit a consistent layer-wise downward trend. This indicates that irrespective of the feature reuse mechanism, the cascade architecture of DF possesses potent feature fitting capabilities, enabling it to continuously approximate the training data distribution as the number of layers increases. However, results from the testing phase (Figs. 4(b) and 4(d)) reveal a fundamental disparity. ERDF w/o fr achieves optimality at the shallow layer, and as the depth further increases, the KL divergence begins to rise rather than decrease. This characteristic "U-shape" curve indicates that as the cascade depth grows, the model begins to overfit the noise inherent in the training data. Specifically, due to the feature enhancement mechanism, ERDF introduces higher-dimensional features at each layer compared to traditional DF, which inevitably introduces more potential noise. The noise is amplified layer by layer, ultimately compromising the model's generalization ability. In contrast, ERDF maintains a continuous downward and stable trend on the testing set, avoiding performance deterioration. This provides compelling evidence that the Feature Reuse mechanism effectively curbs the diffusion of noise within the features. Consequently, it mitigates the overfitting problem in deep cascade structures, ensuring that the model can safely leverage the representational power of deeper structure.

### 4.5. Efficiency analysis

ERDF introduces two new modules, which inevitably lead to additional computational costs. We conduct ablation experiments to analyze the running time of each module. The experimental results are shown in Fig. 5. It is worth emphasizing that directly focusing on the absolute time values of the vertical axis is not meaningful. Since we need to use a custom decision tree splitting metric, a purely handcrafted implementation of the decision tree at the bottom layer is adopted. Although its algorithm is consistent with the scikit-learn library, the overall running time is relatively long due to not using acceleration methods such as Cython. Therefore, the focus of our analysis is on the relative time of each ablation variant compared to the baseline DF.

From the figure, a phenomenon worthy of discussion can be observed: whether a single module is introduced alone or both modules are combined simultaneously, the total time consumption is basically similar, and it is significantly longer than the original DF. The reason behind this phenomenon can be traced. In terms of the module itself, the additional time complexity introduced by feature reuse and feature enhancement is very low. The feature reuse mechanism directly uses the already calculated validation prediction results of each layer in DF, with almost no additional cost, while the feature enhancement mechanism only involves some lightweight statistical matrix calculations and training several small-scale forest models as enhancers. The




(a) time costs during the training phase (b) time costs during the testing phase

**Fig. 5.** Comparison of time costs during the training and testing phases among different ablation variants. The horizontal axis represents the datasets, and the vertical axis indicates the execution time in seconds. Each bar corresponds to a specific ablation variant. For each dataset, the time cost of the original DF is set as the baseline (1.0x).

**Table 5**
This table presents the results of the sensitivity experiment on the hyperparameter $k$. The column headers indicate the values of $k$, where "max" means that $k$ is equal to the number of classes. All results are reported as mean ± standard deviation over three random splits. The arrow beside the metric indicates the direction for better performance.

| Dataset | Metric | 1 | 3 | 5 | max |
|---|---|---|---|---|---|
| Natural | KL div ↓ | 0.5519 ± 0.0127 | 0.5271 ± 0.0116 | **0.5216 ± 0.0155** | 0.5235 ± 0.0156 |
| | Chebyshev ↓ | 0.2352 ± 0.0066 | 0.2261 ± 0.0039 | 0.2221 ± 0.0049 | **0.2201 ± 0.0073** |
| | Clark ↓ | 2.3871 ± 0.0188 | 2.3715 ± 0.0173 | 2.3595 ± 0.0209 | **2.3427 ± 0.0207** |
| | Canberra ↓ | 6.4163 ± 0.0743 | 6.3541 ± 0.0640 | 6.3066 ± 0.0777 | **6.2353 ± 0.0807** |
| | Cosine ↑ | 0.8258 ± 0.0027 | 0.8319 ± 0.0024 | 0.8343 ± 0.0040 | **0.8347 ± 0.0049** |
| | Intersection ↑ | 0.6445 ± 0.0068 | 0.6660 ± 0.0034 | 0.6704 ± 0.0062 | **0.6772 ± 0.0082** |
| SBU | KL div ↓ | 0.0439 ± 0.0007 | **0.0406 ± 0.0014** | 0.0414 ± 0.0005 | 0.0433 ± 0.0013 |
| | Chebyshev ↓ | 0.0889 ± 0.0009 | **0.0863 ± 0.0016** | 0.0865 ± 0.0013 | 0.0880 ± 0.0014 |
| | Clark ↓ | 0.2825 ± 0.0011 | 0.2754 ± 0.0026 | **0.2746 ± 0.0020** | 0.2808 ± 0.0025 |
| | Canberra ↓ | 0.5926 ± 0.0024 | 0.5781 ± 0.0056 | **0.5772 ± 0.0044** | 0.5897 ± 0.0037 |
| | Cosine ↑ | 0.9562 ± 0.0008 | **0.9593 ± 0.0014** | 0.9587 ± 0.0006 | 0.9566 ± 0.0014 |
| | Intersection ↑ | 0.8938 ± 0.0006 | 0.8965 ± 0.0012 | **0.8968 ± 0.0007** | 0.8943 ± 0.0012 |
| emotion6 | KL div ↓ | **0.4932 ± 0.0205** | 0.4981 ± 0.0128 | 0.5111 ± 0.0252 | 0.5265 ± 0.0244 |
| | Chebyshev ↓ | 0.2840 ± 0.0105 | **0.2829 ± 0.0031** | 0.2869 ± 0.0056 | 0.2896 ± 0.0055 |
| | Clark ↓ | 1.6599 ± 0.0225 | 1.6514 ± 0.0201 | **1.6495 ± 0.0230** | 1.6582 ± 0.0238 |
| | Canberra ↓ | 3.7148 ± 0.0660 | 3.6900 ± 0.0509 | **3.6858 ± 0.0684** | 3.7126 ± 0.0666 |
| | Cosine ↑ | **0.7596 ± 0.0071** | 0.7552 ± 0.0022 | 0.7496 ± 0.0078 | 0.7434 ± 0.0064 |
| | Intersection ↑ | 0.6277 ± 0.0108 | **0.6283 ± 0.0031** | 0.6238 ± 0.0089 | 0.6200 ± 0.0082 |
| Avg. Rank | | 3.44 | 2.06 | 1.89 | 2.61 |

fundamental reason that actually leads to the extra time consumption is that the model is trained to a deeper layer. Specifically, w/o fe improves the training stability by eliminating low-quality features, enabling the model to robustly progress to deeper layers. w/o fr is prone to overfitting due to the lack of feature selection. The validation set fails to detect the overfitting phenomenon and trigger early stopping, causing the model to iterate deeper. The consequence is the "U-shaped" curve shown in Fig. 4, where the training KL curve keeps declining, and the test KL curve first declines and then rises.

The complete ERDF integrates the advantages of both modules and naturally forms a deeper cascading structure compared to DF. However, its overall time cost does not show a cumulative increase. This result fully proves that the main computational cost increase of ERDF does not stem from the complex calculations of the newly introduced module itself, but from the training cost brought by the extra layers.

### 4.6. Parameter sensitivity analysis

Compared with algorithms based on neural networks, one of the major advantages of ERDF is that it does not require a cumbersome parameter tuning process. The hyperparameters of the deep forest itself are far fewer than those of neural networks, and ERDF only introduces a very small number of additional hyperparameters (such as the dimension $k$ of enhanced features, the evaluation metric $\mathcal{M}$ for feature reuse). Among them, the parameter that has the most significant impact on model performance is undoubtedly $k$. Therefore, we conduct a parameter sensitivity analysis on $k$, and the experimental results are shown in Table 5. To ensure that k has a sufficient value range, we select three datasets with more than 5 label categories for the experiment.

From the experimental results, it can be seen that when $k$ takes an intermediate value (such as 3 or 5), the overall performance of the algorithm is the most stable, which is in line with our expectations. At the same time, extreme values lead to poorer results. If $k$ is too small, the information contained in the enhanced features is insufficient, making it difficult to exert its intended effect; if $k$ is too large, it will cause the model to overly rely on the relationship patterns, and thus may fit the noise in the label matrix. Of course, the intrinsic characteristics of different datasets will also lead to different performances. For example, on the Natural Scene dataset, when $k$ takes the maximum value, the results are the best. This is because this dataset consists of natural landscape images, and the visual elements represented by the labels (such as beach, building, clouds) have very significant and clear co-occurrence correlations, and the relationship matrix contains less noise. Consequently, more feature dimensions can capture richer semantics. In contrast, for the SCUT dataset we used in the previous benchmark experiments, its labels represent a rating scale (such as Unattractive, Average, Attractive, etc.), and the correlation structure between such labels is relatively simple. Extracting too many relationship patterns does not bring additional benefits.

Overall, in our experiment, setting $k$ to 5 as the default value is a robust choice that balances performance and stability. In practice, the value of $k$ can be finely adjusted based on the characteristics of the specific dataset for better performance.

## 5. Conclusions

In this paper, we investigated the problem of label distribution learning and proposed ERDF, a dual-mechanism extension of DF that integrates feature enhancement and measure-aware feature reuse. The proposed framework jointly leverages label correlations to enrich representations while maintaining training stability through adaptive feature reuse. Extensive experiments across multiple benchmark datasets demonstrate that ERDF achieves significant improvements over existing methods, validating both the effectiveness of the two mechanisms.

For future work, two directions appear particularly promising. First, the feature enhancement mechanism is conceptually general and may benefit other tasks that rely on exploiting label dependencies, such as multi-label learning. Exploring these extensions could further broaden the applicability of our approach. Second, the current feature reuse strategy relies on a unified criterion derived from the performance of the first component of the enhanced features. Developing more fine-grained reuse schemes that differentiate among various components of the generated features may further improve stability and performance.

## CRediT authorship contribution statement

**Jia-Le Xu:** Writing – original draft, Methodology, Formal analysis, Conceptualization. **Shen-Huan Lyu:** Writing – review & editing, Supervision, Project administration, Funding acquisition. **Yu-Nian Wang:** Writing – review & editing, Investigation. **Ning Chen:** Writing – review & editing, Validation. **Zhihao Qu:** Writing – review & editing, Validation. **Bin Tang:** Writing – review & editing, Validation. **Baoliu Ye:** Writing – review & editing, Validation.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. 62306104, 62441225 and 62572171), Basic Research Program of Jiangsu (No. BK20253011), Hong Kong Scholars Program (No. XJ2024010), Research Grants Council of the Hong Kong Special Administrative Region, China (GRF Project No. CityU11212524), Natural Science Foundation of Jiangsu Province, China (No. BK20230949), Jiangsu Association for Science and Technology (No. JSTJ2024285), China Postdoctoral Science Foundation (No. 2023TQ0104), and the High Performance Computing Platform of Hohai University.

## Data availability

Data will be made available on request.

## References

[1] M.-L. Zhang, Z.-H. Zhou, A review on multi-label learning algorithms, IEEE Trans. Knowl. Data Eng. 26 (8) (2013) 1819–1837.
[2] Q. Ma, C. Yuan, W. Zhou, S. Hu, Label-specific dual graph neural network for multi-label text classification, in: Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics, 2021, pp. 3855–3864.
[3] T. Wu, S. Yang, Contrastive enhanced learning for multi-label text classification, Appl. Sci. 14 (2024) 8650.
[4] G. Lyu, Z. Yang, X. Deng, S. Feng, L-VSM: Label-driven view-specific fusion for multiview multilabel classification, IEEE Trans. Neural Netw. Learn. Syst. 36 (2025) 6569–6583.
[5] Q. Zhong, G. Lyu, Z. Yang, Align While Fusion: A generalized nonaligned multiview multilabel classification method, IEEE Trans. Neural Netw. Learn. Syst. 36 (2025) 7627–7636.
[6] Z.-M. Chen, X.-S. Wei, P. Wang, Y. Guo, Learning graph convolutional networks for multi-label recognition and applications, IEEE Trans. Pattern Anal. Mach. Intell. 45 (2021) 6969–6983.
[7] Z.-M. Chen, X. Jin, Y. Ge, S. Chan, In pursuit of causal label correlations for multi-label image recognition, in: Advances in Neural Information Processing Systems 37, 2024, pp. 51634–51654.
[8] C. Zhang, C. Xu, Y. Xie, W. Mao, B. Yu, Dynamic multi-modal hypergraph learning for semi-supervised multi-label image recognition, Pattern Recognit. 169 (2026) 111959.
[9] X. Geng, Label distribution learning, IEEE Trans. Knowl. Data Eng. 28 (7) (2016) 1734–1748.
[10] N. Le, K. Nguyen, Q. Tran, E. Tjiputra, B. Le, A. Nguyen, Uncertainty-aware label distribution learning for facial expression recognition, in: Proceedings of the 40th IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 6088–6097.
[11] H. Shin, B. Lee, B. Ku, H. Ko, Noisy label facial expression recognition via face-specific label distribution learning, Image Vis. Comput. 143 (2024) 104901.
[12] A. Khelifa, H. Ghazouani, W. Barhoumi, Label distribution learning for compound facial expression recognition in-the-wild: A comparative study, Expert Syst. 42 (2) (2025) e13724.
[13] H. Xu, X. Liu, Q. Zhao, Y. Ma, C. Yan, F. Dai, Gaussian label distribution learning for spherical image object detection, in: Proceedings of the 40th IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 1033–1042.
[14] M. Nishio, H. Matsuo, Y. Kurata, O. Sugiyama, K. Fujimoto, Label distribution learning for automatic cancer grading of histopathological images of prostate cancer, Cancers 15 (5) (2023) 1535.
[15] A. Shrivastava, N. Rajput, P. Rajesh, S. Swarnalatha, IoT-based label distribution learning mechanism for autism spectrum disorder for healthcare application, in: Practical Artificial Intelligence for Internet of Medical Things, CRC Press, 2023, pp. 305–321.
[16] N. Chen, S.-H. Lyu, T.-S. Wu, Y. Wang, B. Tang, Improving multi-label contrastive learning by leveraging label distribution, Pattern Recognit. 174 (2026) 113011.
[17] X. Jia, W. Li, J. Liu, Y. Zhang, Label distribution learning by exploiting label correlations, in: Proceedings of the 32nd AAAI Conference on Artificial Intelligence, vol. 32, (1) 2018.
[18] T. Ren, X. Jia, W. Li, L. Chen, Z. Li, Label distribution learning with label-specific features, in: Proceedings of the 28th International Joint Conference on Artificial Intelligence, 2019, pp. 3318–3324.
[19] Y. Jin, R. Gao, Y. He, X. Zhu, GLDL: Graph label distribution learning, in: Proceedings of the AAAI Conference on Artificial Intelligence, 2024, pp. 12965–12974.
[20] Y. Lin, Y. Li, C. Wang, L. Guo, J. Chen, Label distribution learning based on horizontal and vertical mining of label correlations, IEEE Trans. Big Data 10 (3) (2024) 275–287.
[21] Y. Lin, G. Lyu, H. Cai, D.-B. Wang, H. Wang, Z. Yang, Simplified graph contrastive learning model without augmentation, IEEE Trans. Knowl. Data Eng. 37 (2025) 6159–6172.
[22] H. Wu, W. Li, X. Jia, Domain adaptation for label distribution learning, IEEE Trans. Big Data 11 (3) (2025) 1221–1234.
[23] X. Jia, X. Shen, W. Li, Y. Lu, J. Zhu, Label distribution learning by maintaining label ranking relation, IEEE Trans. Knowl. Data Eng. 35 (2) (2021) 1695–1707.
[24] S. Xu, L. Shang, F. Shen, X. Yang, W. Pedrycz, Incomplete label distribution learning via label correlation decomposition, Inf. Fusion (ISSN: 1566-2535) 113 (2025) 102600.
[25] T. Ren, X. Jia, W. Li, S. Zhao, Label distribution learning with label correlations via low-rank approximation, in: Proceedings of the 28th International Joint Conference on Artificial Intelligence, 2019, pp. 3325–3331.
[26] H. Tang, J. Zhu, Q. Zheng, J. Wang, S. Pang, Z. Li, Label enhancement with sample correlations via low-rank representation, Proc. the AAAI Conf. Artif. Intell. 34 (04) (2020) 5932–5939.
[27] Z.-H. Zhou, J. Feng, Deep forest: towards an alternative to deep neural networks, in: Proceedings of the 26th International Joint Conference on Artificial Intelligence, 2017, pp. 3553–3559.
[28] Y. Boualleg, M. Farah, I.R. Farah, Remote sensing scene classification using convolutional features and deep forest classifier, IEEE Geosci. Remote. Sens. Lett. 16 (12) (2019) 1944–1948.
[29] J.-M. Burmeister, J. Zabbarov, S. Reder, R. Richter, J.-P. Mund, J. Döllner, Fine-tuning DeepForest for forest tree detection in high-resolution uav imagery, Int. Arch. Photogramm. Remote. Sens. Spat. Inf. Sci. 48 (2025) 39–46.
[30] Y.-H. Chen, S.-H. Lyu, Y. Jiang, Improving deep forest by exploiting high-order interactions, in: Proceedings of the 21st IEEE International Conference on Data Mining, 2021, pp. 1030–1035.
[31] L.V. Utkin, An imprecise deep forest for classification, Expert Syst. Appl. 141 (2020) 112978.
[32] L.V. Utkin, M.A. Ryabinin, A siamese deep forest, Knowl.-Based Syst. 139 (2018) 13–22.
[33] S.-H. Lyu, Y.-H. Chen, Z.-H. Zhou, A region-based analysis for the feature concatenation in deep forests, Chin. J. Electron. 31 (6) (2022) 1072–1080.
[34] S.-H. Lyu, Y.-X. He, Z.-H. Zhou, Depth is more powerful than width with prediction concatenation in deep forests, in: Advances in Neural Information Processing Systems 35, 2022, pp. 29719–29732.

[35] L. Yang, X.-Z. Wu, Y. Jiang, Z.-H. Zhou, Multi-label learning with deep forest, in: Proceedings of the 24th European Conference on Artificial Intelligence, 2020, pp. 1634–1641.
[36] P. Ilidio, R. Cerri, C. Vens, F.K. Nakano, Deep forests with tree-embeddings and label imputation for weak-label learning, in: Proceedings of the 33th International Joint Conference on Artificial Intelligence, 2024, pp. 1–8.
[37] M. Chen, X. Wang, B. Feng, W. Liu, Structured random forest for label distribution learning, Neurocomputing 320 (2018) 171–182.
[38] X. Jia, Z. Li, X. Zheng, W. Li, S.-J. Huang, Label distribution learning with label correlations on local samples, IEEE Trans. Knowl. Data Eng. 33 (4) (2019) 1619–1631.
[39] C. Tan, S. Chen, J. Zhang, Z. Xu, X. Geng, G. Ji, RG4LDL: Renormalization group for label distribution learning, Knowl.-Based Syst. 320 (2025) 113666.