SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

November 2024, Vol. 67, Iss. 11, 212102:1–212102:14
https://doi.org/10.1007/s11432-022-3915-6

Residual diverse ensemble for long-tailed multi-label text classification

Jiangxin SHI[1,2†], Tong WEI[3,4†] & Yufeng LI[1,2*]

[1]*National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing* 210023*, China;*
[2]*School of Artificial Intelligence, Nanjing University, Nanjing* 210023*, China;*
[3]*School of Computer Science and Engineering, Southeast University, Nanjing* 210096*, China;*
[4]*Key Laboratory of Computer Network and Information Integration, Southeast University, Ministry of Education, Nanjing* 210096*, China*

Received 23 September 2022/Revised 14 May 2023/Accepted 15 August 2023/Published online 23 October 2024

Abstract Long-tailed multi-label text classification aims to identify a subset of relevant labels from a large candidate label set, where the training datasets usually follow long-tailed label distributions. Many of the previous studies have treated head and tail labels equally, resulting in unsatisfactory performance for identifying tail labels. To address this issue, this paper proposes a novel learning method that combines arbitrary models with two steps. The first step is the "diverse ensemble" that encourages diverse predictions among multiple shallow classifiers, particularly on tail labels, and can improve the generalization of tail labels. The second is the "error correction" that takes advantage of accurate predictions on head labels by the base model and approximates its residual errors for tail labels. Thus, it enables the "diverse ensemble" to focus on optimizing the tail label performance. This overall procedure is called residual diverse ensemble (RDE). RDE is implemented via a single-hidden-layer perceptron and can be used for scaling up to hundreds of thousands of labels. We empirically show that RDE consistently improves many existing models with considerable performance gains on benchmark datasets, especially with respect to the propensity-scored evaluation metrics. Moreover, RDE converges in less than 30 training epochs without increasing the computational overhead.

Keywords multi-label learning, extreme multi-label learning, long-tailed distribution, multi-label text classification, ensemble learning

1 Introduction

Recently, there has been a rapid growth in the data scale for various industrial applications. Machine learning problems comprising hundreds of thousands of labels are now quite common in various domains, such as recommendation system [1], annotating web-scale encyclopedia [2], and e-commerce [3]. Traditional multi-label learning algorithms [4–7] are not feasible in large-scale scenarios because of the heavy computational overhead. Therefore, to address this problem, the large-scale multi-label learning [8–11] has been proposed, wherein a classifier is learned to efficiently and effectively annotate given instances using the most relevant labels from an extremely large label space.

An important statistical characteristic of the large-scale multi-label text classification datasets is that the frequency of their different labels exhibits a long-tailed distribution [2, 12]. The labels observed in a few training instances are called tail labels, while others are called head labels. Figure 1 shows the distribution of label frequencies in commonly used long-tailed multi-label datasets. The figure shows that most labels are observed only in a few training instances, and only a small proportion of labels (≈15% in Amazon-670K) appear in more than five training instances.

Several methods have been proposed to learn from the large-scale multi-label datasets, including FastXML [13], DiSMEC [14], and LightXML [15]. However, most of these methods did not consider the long-tailed distribution. Interestingly, a study [9] claimed that it can still obtain good performance without considering tail labels with respect to commonly used performance metrics (P@k and nDCG@k).

* Corresponding author (email: liyf@nju.edu.cn)
† Shi J X and Wei T have the same contribution to this work.

© Science China Press 2024

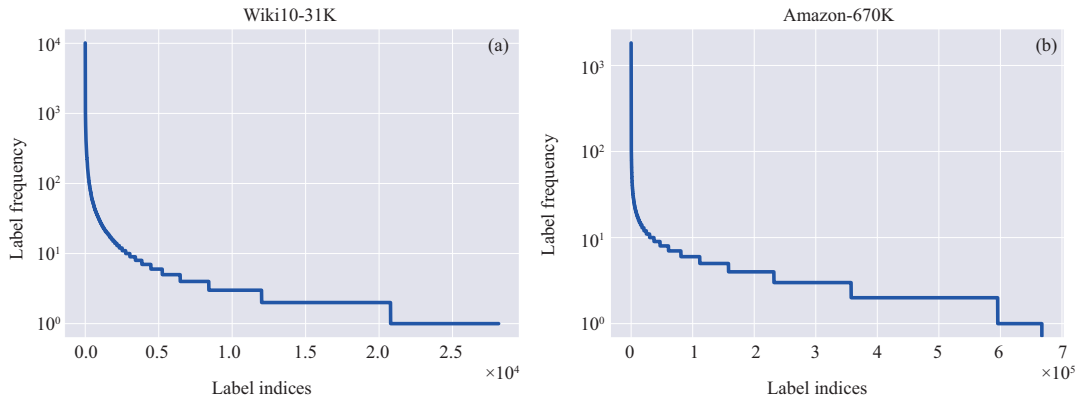

Figure 1 (Color online) Label frequency of (a) Wiki10-31K and (b) Amazon-670K datasets, which follows a long-tailed distribution.

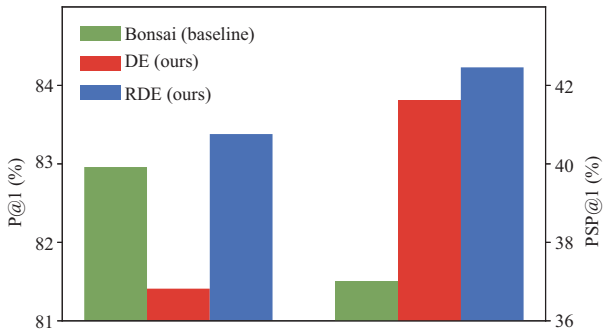

Figure 2 (Color online) Comparisons of our proposed methods (DE and RDE) with Bonsai [17] as the base model on EurLex-4K dataset.

However, in some real-world applications such as personalized recommendations [16], achieving the accurate prediction of tail labels is more desirable because they tend to carry more informative value. Unlike head labels, tail labels have very less training instances to train a satisfactory model, posing a great challenge for the advancement of long-tailed multi-label learning. To encourage more correct predictions of tail labels, some more evaluation metrics have been proposed, including propensity-scored precision@k (PSP@k) and propensity-scored nDCG@k (PSnDCG@k) [16].

Re-sampling and re-weighting are classical approaches for dealing with class imbalance in single-label classification problems. However, in multi-label learning, each instance can be associated with multiple labels. Therefore, the oversampling of tail label examples simultaneously increases the head label frequency. By contrast, the re-weighting method assigns individual weights to each label and imposes large penalties for incorrect tail label predictions, which is more suitable for multi-label settings. For example, PfastreXML [16] computes the propensity scores of labels and optimizes a propensity-scored objective function, considerably improving the tail label predictions. Nevertheless, the performance of PfastreXML with respect to vanilla metrics (P@k and nDCG@k) is far from competitive compared with other state-of-the-art methods, indicating the potential of PfastreXML to enhance the head label accuracy.

In light of the above considerations, we propose a novel method called residual diverse ensemble (RDE) to address the long-tailed multi-label classification problem. To combat the data scarcity of tail labels, RDE learns an ensemble of shallow classifiers and encourages diverse predictions. We also find that training classifiers from scratch usually lead to slow convergence. Therefore, we integrate our proposed method into existing multi-label learning models to approximate its residual errors. In other words, our method minimizes the error between the prediction of a base model and the ground truth by learning the residuals between them. In this way, it efficiently employs the preliminary predictions of the base model and improves the performance of tail labels. Figure 2 compares our proposed method RDE with the base method Bonsai [17]. The results show that the performance with respect to PSP@1 improves by a considerable gain, by applying the diverse ensemble (DE), whereas that of P@1 deteriorates. This is because DE encourages diverse predictions between classifiers, which may result in many incorrect predictions, especially on head labels. Moreover, the residual learning module performs better using the

predictions of the base model since RDE does not sacrifice P@1 than DE. The results indicate that the proposed method can effectively improve the performance of tail labels without affecting the performance of head labels. The main advantages of the proposed method RDE are summarized as follows.

- The proposed method is general and can be applied to any existing multi-label learning model.
- The proposed method can improve the tail label generalization without compromising the performance of head labels.
- The proposed method is lightweight and converges in less than 30 training epochs.

2 Related work

Large-scale multi-label learning. The large-scale multi-label learning aims to associate an instance with the most relevant labels from an extremely large candidate set. Existing studies mainly fall into one of the following four categories. (1) One-vs.-rest methods [14, 18–22]. As the name suggests, one-vs.-rest methods train a classifier for each label to compute its confidence against the remaining labels. Such methods suffer from high computational complexity, especially when the label space is extremely large. (2) Tree-based methods [13, 17, 23–28]. To reduce the computational cost, tree-based methods partition the large label space into several subspaces and conquer the classification problem in each subspace hierarchically. Parabel [27] and Bonsai [17] recursively partition the entire label set into two branches of label subsets and train a one-vs.-rest classifier for each leaf node within a sufficiently small label space. FastXML [13] learns to split instances using a hyperplane rather than a single feature. (3) Embedding-based methods [8, 29–35]. This method assumes that despite the high dimension of the label space, the label matrix is usually low ranking and can be projected to a low-dimensional embedding space. (4) Deep learning methods [36–41]. Deep learning methods use raw text as input instead of bag-of-word (BoW) features to extract more informative feature representations, such as using semantic word embeddings. Specifically, AttentionXML [38] employs a bidirectional long short-term memory (LSTM) [42] to learn context representations, and a multi-label attention module is developed based on the learned representations. Most of the abovementioned studies have focused on overall prediction performance and treated all labels identically; however, they are not specifically designed to consider tail labels.

Learning to predict tail labels. Apart from improving the overall performance, several extant methods aim to improve the performance of tail labels. For example, PfastreXML [16] incorporates the propensity-scored knowledge for each label and optimizes the propensity-scored nDCG during training. However, PfastreXML improves the performance of tail labels at the cost of sacrificing head label accuracy, which is not desirable. Moreover, label propensity scores are difficult to access in some real-world scenarios, hindering their applicability. ProXML [22] improves the performance of tail labels by designing a robust optimization problem. TAIL [43] proposes to construct label-specific features to facilitate the learning of data-scarce tail labels. However, ProXML and TAIL learn a one-vs.-rest style model by assuming that labels are independent, which may be suboptimal for multi-label learning. In this paper, we propose a new tail label prediction enhancement method by learning residual errors and optimizing a diverse loss function. Our proposed method is more efficient and scalable since it can easily converge and be incorporated with any existing methods.

3 RDE

3.1 Overview

Given the multi-label training dataset $\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^N$, where $\boldsymbol{x}_i \in \mathbb{R}^D$ represents the feature vector of the i-th training instance and $\boldsymbol{y}_i = [y_{i1}; y_{i2}; \ldots; y_{il}] \in \{0,1\}^L$ represents the associated label vector. Conventionally, a multi-label learning method aims to learn a classifier $h : \mathbb{R}^D \to \mathbb{R}^L$ which takes a feature vector $\boldsymbol{x}_i$ as input and predicts its relevance scores $\hat{\boldsymbol{y}}_i = [\hat{y}_{i1}; \hat{y}_{i2}; \ldots; \hat{y}_{il}] \in [0,1]^L$. Formally, the goal of multi-label learning can be achieved by optimizing the following objective:

$$\min_{h \in \mathcal{H}} \frac{1}{N} \sum_{i=1}^{N} \ell\left(\hat{\boldsymbol{y}}_i, \boldsymbol{y}_i\right) + \mathcal{R}(h), \tag{1}$$

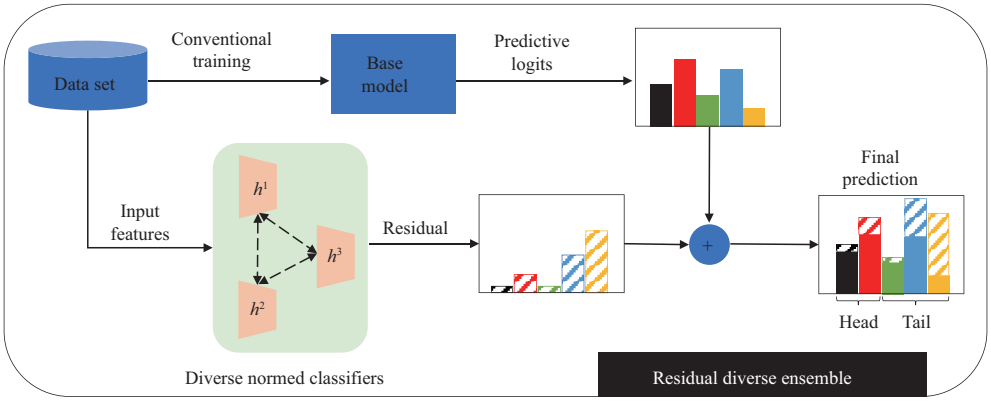

Figure 3 (Color online) Overview of the proposed method.

Table 1 Major mathematical notations

Notations	Meanings
$\mathcal{D}$	Multi-label training dataset $\{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^N$
$\boldsymbol{x}_i$	Feature vector of the i-th training instance, $\boldsymbol{x}_i \in \mathbb{R}^D$
$\boldsymbol{y}_i$	Ground-truth label of the i-th training instance, $\boldsymbol{y}_i = [y_{i1}; y_{i2}; \ldots; y_{iL}] \in \{0,1\}^L$
$h_0(\cdot)$	Base model (any existing model), $h_0 : \mathbb{R}^D \to \mathbb{R}^L$
$h^k(\cdot)$	The k-th classifier $h^k : \mathbb{R}^D \to \mathbb{R}^L$
$\sigma(\cdot)$	Activation function $\sigma : \mathbb{R}^L \to [0,1]^L$. In multi-label classification, we use the sigmoid activation function $\sigma(z) = \frac{1}{1+e^{-z}}$
$\tilde{\boldsymbol{z}}_i, \tilde{\boldsymbol{y}}_i$	Logit vector and final predictions output by h_0, i.e., $\tilde{\boldsymbol{z}}_i = h_0(\boldsymbol{x}_i)$, and $\tilde{\boldsymbol{y}}_i = \sigma(\tilde{\boldsymbol{z}}_i)$
$\hat{\boldsymbol{z}}_i^k, \hat{\boldsymbol{y}}_i^k$	Logit vector and final predictions output by h^k, i.e., $\hat{\boldsymbol{z}}_i^k = h^k(\boldsymbol{x}_i)$, and $\hat{\boldsymbol{y}}_i^k = \sigma(\hat{\boldsymbol{z}}_i^k)$

where $\mathcal{R}(h)$ is the regularizer imposed on h, and ℓ denotes the multi-label loss function, such as the frequently used binary cross-entropy (BCE) loss:

$$\ell_{\mathrm{BCE}}\left(\hat{\boldsymbol{y}}_i, \boldsymbol{y}_i\right) = -\frac{1}{L}\sum_{l=1}^{L} w_l \left(y_{il} \log \hat{y}_{il} + (1 - y_{il}) \log(1 - \hat{y}_{il})\right). \tag{2}$$

In long-tailed datasets, only a few training instances are available for tail labels, making it hard to learn a classifier with satisfactory generalization performance. To alleviate this issue, we propose to learn DE classifiers $\{h^1, h^2, \ldots, h^K\}$. Specifically, we train multiple classifiers by jointly minimizing the classification error and separately maximizing the diversity between each other. This can be achieved by designing a diversity loss function, which encourages classifiers to produce diverse predictions for better ensemble performance. In other words, the regularizer term $\mathcal{R}(h)$ is instantiated as the diversity between each classifier h and other classifiers.

However, we empirically find that directly optimizing DE classifiers may lead to performance deterioration w.r.t. vanilla performance metrics, i.e., P@k as demonstrated in Figure 2. By taking advantage of effective and efficient models that have been proposed, we attempt to learn the residual errors of an existing base model h_0 by optimizing the residual errors between its predictions and the ground truths. Figure 3 illustrates an overview of the proposed method. In the following, we present the proposed two components, i.e., the DE optimization and the residual error correction. We summarize frequently used mathematical notations and their meanings in Table 1 for ease of reference.

3.2 DE optimization

3.2.1 *Diversity loss*

To achieve better generalization performance for ensemble classifiers, we present a DE module with multiple classifiers by optimizing the diversity loss function. In this manner, each classifier is designed to minimize its classification error, as well as maximize its prediction diversity with other classifiers.

Formally, for the k-th classifier, the diversity loss is calculated as

$$\mathcal{L}_{\text{div}}^{k} = -\frac{1}{N}\sum_{i=1}^{N}\frac{1}{K-1}\sum_{k'\neq k}^{K}\text{KL}(\phi^{k'}(\boldsymbol{x}_i,\boldsymbol{T}),\phi^{k}(\boldsymbol{x}_i,\boldsymbol{T})), \tag{3}$$

where $\text{KL}(P,Q) = \sum_{x\in\mathcal{X}} P(x)\log\frac{P(x)}{Q(x)}$ is the KL-divergence and $\phi^k(\boldsymbol{x}_i,\boldsymbol{T}) = \text{Softmax}(h^k(\boldsymbol{x}_i)\cdot\boldsymbol{T}) = \text{Softmax}(\hat{\boldsymbol{z}}_i^k\cdot\boldsymbol{T})$. We use K to denote the number of classifiers. Moreover, $\boldsymbol{T}$ refers to the temperature of each sample. Since our main concern is the performance improvement on tail labels, we set a higher temperature for tail labels to encourage more diverse predictions. We use label frequencies to calculate temperature T_l for the l-th label as follows:

$$T_l = \frac{1}{\sqrt{N_l}}, \tag{4}$$

where N_l is the number of training instances associated with the l-th label.

3.2.2 *Normalized classifier*

Training models using long-tailed data tend to result in biases toward head labels. Previous studies [44–46] empirically shows that the norms of the last layers of the classifiers are likely to follow an imbalanced distribution, where the norms of head label classifiers are much larger than that of tail labels. This naturally introduces biases in inference. To overcome this problem, we use normalized classifiers such that all label classifiers have the same magnitude of prediction scores. Formally, each classifier h can be decomposed into an embedding projection $\boldsymbol{E}(\cdot):\mathbb{R}^D\to\mathbb{R}^d$ and a last-layer linear projection $\boldsymbol{W} = [\boldsymbol{w}_1;\boldsymbol{w}_2;\ldots;\boldsymbol{w}_L]^{\text{T}}\in\mathbb{R}^{L\times d}$. Then its decision function can be written as $h(\boldsymbol{x}) = \boldsymbol{W}\boldsymbol{E}(\boldsymbol{x})$. To yield a normalized classifier, we apply L_2 normalization on $\boldsymbol{W}$ such that

$$\overline{\boldsymbol{w}}_l = \frac{\boldsymbol{w}_l}{\|\boldsymbol{w}_l\|_2},\quad \forall l\in[L]. \tag{5}$$

After that, $\overline{\boldsymbol{W}} = [\overline{\boldsymbol{w}}_1;\overline{\boldsymbol{w}}_2;\ldots;\overline{\boldsymbol{w}}_L]^{\text{T}}$ is used for inference instead of $\boldsymbol{W}$. It is worth mentioning that the normalized classifiers essentially balance the output of the model across different labels, thus can also alleviate the tail label problem.

In total, by combining the diversity loss and the classification loss, the final loss function for the k-th classifier can be written as

$$\mathcal{L}_{\text{total}}^{k} = \mathcal{L}_{\text{clf}}^{k} + \lambda\cdot\mathcal{L}_{\text{div}}^{k}, \tag{6}$$

where λ is the balancing factor between diversity loss and classification loss. $\mathcal{L}_{\text{clf}}^{k}$ refers to the conventional BCE loss in (2), i.e., $\mathcal{L}_{\text{clf}}^{k} = \ell_{\text{BCE}}\left(\hat{\boldsymbol{y}}_i^k,\boldsymbol{y}_i^k\right)$.

Applying the DE module can improve the generalization of tail labels. However, as demonstrated in Figure 2, directly optimizing the diversity loss can lead to performance deterioration w.r.t. vanilla metrics, such as P@1. The reason is that encouraging the diversity between classifiers can result in inconsistent and incorrect predictions on head labels. To alleviate this problem, we propose the residual error correction module.

3.3 Residual error correction

To ensure more stable performances on head labels, we propose a residual error correction module by taking advantage of existing learning models. Formally, supposing that a base model h_0 takes instance $\boldsymbol{x}\in\mathbb{R}^D$ as the input and outputs the logit vector $\tilde{\boldsymbol{z}} = [\tilde{z}_1;\tilde{z}_2;\ldots;\tilde{z}_l]\in\mathbb{R}^L$. We learn the residual errors between base model predictions $\tilde{\boldsymbol{z}}$ and ground-truth labels $\boldsymbol{y}_i$. Formally, for instance $\boldsymbol{x}$, we obtain its predictions $\hat{\boldsymbol{y}}$ as

$$\hat{\boldsymbol{y}} = \sigma(h(\boldsymbol{x}) + \tilde{\boldsymbol{z}}), \tag{7}$$

where $\sigma(\cdot)$ is the sigmoid activation function, i.e., $\sigma(z) = \frac{1}{1+\text{e}^{-z}}$. If the base model outputs the probability vector $\tilde{\boldsymbol{y}}$ rather than $\tilde{\boldsymbol{z}}$, we calculate $\tilde{\boldsymbol{z}} = \sigma^{-1}(\tilde{\boldsymbol{y}})$ to get the logit vector. By this means, the classifier h predicts the residual errors instead of the ground-truth labels.

From (7), we can see that if the base model has an accurate prediction for the l-th label, the residual error is small and $h(\boldsymbol{x})_l$ is expected to be close to 0. Conversely, if the prediction result of the base model

Algorithm 1 Training procedure of RDE

Input: Training data $\mathcal{D} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1}^{N}$; base model h_0; Number of classifiers n;
Output: Ensemble of classifiers $\mathcal{H} = \{h^k\}_{k=1}^{n}$;

// Calculate base model results
Initialize base model output logits $\tilde{\boldsymbol{Z}} = \emptyset$;
Initialize base model predictions $\tilde{\boldsymbol{Y}} = \emptyset$;
for $(\boldsymbol{x}_i, \boldsymbol{y}_i) \in \mathcal{D}$ **do**
  Calculate $\tilde{\boldsymbol{z}}_i = h_0(\boldsymbol{x}_i)$ and $\tilde{\boldsymbol{y}}_i = \sigma(\tilde{\boldsymbol{z}}_i)$;
  $\tilde{\boldsymbol{Z}} = \tilde{\boldsymbol{Z}} \bigcup \tilde{\boldsymbol{z}}_i$, $\tilde{\boldsymbol{Y}} = \tilde{\boldsymbol{Y}} \bigcup \tilde{\boldsymbol{y}}_i$;
end for
// Train ensemble of classifiers
Initialize classifiers $\{h^1, h^2, \ldots, h^n\}$;
Warm-up classifiers for 5 epochs by optimizing BCE;
repeat
  for $(\boldsymbol{x}_i, \boldsymbol{y}_i) \in \mathcal{D}$, $\tilde{\boldsymbol{z}}_i \in \tilde{\boldsymbol{Z}}$ and $\tilde{\boldsymbol{y}}_i \in \tilde{\boldsymbol{Y}}$ **do**
    for $h^k \in \mathcal{H}$ **do**
      Calculate $\hat{\boldsymbol{z}}_i^k$ and $\hat{\boldsymbol{y}}_i^k$ according to Eq. (7);
      Calculate $\mathcal{L}_{\text{clf}}^k$ according to Eq. (2);
    end for
    for $h^k \in \mathcal{H}$ **do**
      Calculate $\mathcal{L}_{\text{div}}^k$ according to Eq. (3);
      Calculate $\mathcal{L}_{\text{total}}^k$ according to Eq. (6);
      Update classifier h^k with $\mathcal{L}_{\text{total}}^k$;
    end for
  end for
until Stop condition reached.

is far from correct on the j-th label, $|h(\boldsymbol{x})_l|$ is expected to be large. Since existing models tend to predict credible results on head labels, the residual errors on head labels are easy to fit. Therefore, the DE module can be trained to boost performance on tail labels, without the risk of performance deterioration on head labels.

To further alleviate the influence of the error correction module on head labels, we use the re-weighted loss function by setting where w_l in (2) as $w_l = \frac{1}{\sqrt{N_l}}$. In this way, we encourage the classifiers to focus more on learning residual errors of tail labels.

Moreover, we empirically find that learning residual errors can yield faster convergence, which further demonstrates the superiority of our method. More details are given in the experiments. To sum up, we list the training procedure of RDE in Algorithm 1.

In the inference stage, we calculate the mean of the logits output by different classifiers and then add the base model results to get the final predictions. Formally, given an instance $\boldsymbol{x}$, we have

$$\hat{\boldsymbol{y}}_{\text{pred}} = \sigma\left(\frac{1}{K}\sum_{k=1}^{K} h^k(\boldsymbol{x}) + \tilde{\boldsymbol{z}}\right). \tag{8}$$

3.4 Complexity analysis

The time complexity of RDE is determined by the following factors: (1) the number of training instances N; (2) the dimension of the input features D and the dimension of the label space L; (3) the hidden size of each single-layer perceptron d; (4) the number of diverse classifiers k; (5) the training epochs T.

For each instance, the inference cost is $O((D + L)d)$, considering that the single-layer perceptron classifier first projects the input features into the embedding space R^d and into the label space. Noting that d is much smaller compared with D or L. When training the ensemble of classifiers, we use all training instances to update each classifier to learn the residual errors between the base model results and the ground truth labels. It has a total time cost of $O(TNk(D+L)d)$. Here $T = 30$ since our method can converge in a short stage. Number k is set to 3, and can be ignored if we use parallel procedures to train different classifiers.

In contrast to the baseline methods, RDE does not introduce too many additional parameters, as it learns a bottleneck perceptron with a size of $(D+L)d$ ($d \ll D, L$). The baseline methods such as Parabel and Bonsai require approximately $(DL \log L)$ parameters for learning multiple leaf node classifiers. The deep model X-Transformer contains more parameters considering its multi-layered design.

Table 2 Dataset statistics[a)]

Dataset	N	D	L	$\overline{L}$	$\widetilde{L}$
EurLex-4K	15539	5000	3993	5.31	25.73
Wiki10-31K	14146	101938	30938	18.64	8.52
Amazon-670K	490449	135909	670091	5.39	5.11

a) N refers to the number of training instances, D/L refers to feature/label dimension, $\overline{L}$ denotes the average number of labels per instance, $\widetilde{L}$ denotes the average number of instances per label.

3.5 Connection to previous work

Our method is partially connected to previous studies on learning from weakly supervised data. In real-world scenarios, obtaining a significant volume of elaborately annotated data is unfeasible, giving rise to the challenge of weakly supervised learning. This is explored in various research domains, such as few-shot learning [47–49] and semi-supervised learning [50, 51]. Furthermore, our study delves into a more realistic and challenging problem where the labels exhibit a long-tailed distribution.

To deal with the long-tailed distribution in single-label classification, several approaches have been proposed. For example, RIDE [52] proposes to enlarge the diversity between predictions generated by different classifiers. ResLT [53] optimizes two residual branches and aggregates their outputs to the main branch to enhance the predictions on tail labels. However, the main difference is that these approaches train the model starting from scratch and learn to predict the ground-truth label directly. Moreover, these works are proposed for single-label classification. This motivates us to design a more efficient and scalable method under multi-label settings.

The re-weighted loss functions are also widely adopted to deal with imbalanced classification problems with long-tailed label distribution, such as Class-Balanced loss [54] and LDAM loss [55]. The difference is that previous re-weighting methods amplify the weights for tail labels at the sacrifice of the performance of head labels. In contrast, our re-weighting strategy is applied to the residual errors, thereby the base results on head labels are less affected.

Another popular work in single-label classification is logit adjustment (LA) [56], which rectifies biased logits through a post hoc calibration and estimates optimal predictions under a balanced class distribution. The motivation of LA is very similar to ours; however, applying LA requires knowledge of the test-time class prior distribution, which is difficult to obtain in real-world scenarios [57, 58]. It remains an interesting problem to estimate the underlying class priors of the multi-label dataset and further improve the generalization, and we leave this for future work.

4 Experiments

In the experiments, we validate the proposed method from the following perspectives.

- Generality and effectiveness. We apply RDE to different types of methods, i.e., tree-based, one-vs.-all, and deep learning methods. Our method consistently improves their performance.
- Efficiency. We show the learning curve of RDE, which converges in less than 30 training epochs.
- Ablation studies on each component. We find that the "DE" improves the generalization for tail labels, and "error correction" makes the classifier perform stable on head labels.

4.1 Datasets and evaluation

Datasets. We conduct experiments on three benchmark datasets with different scales, including EurLex-4K, Wiki10-31K, and Amazon-670K. Table 2 lists the statistics of these datasets. All datasets we used are based on BoW features and can be directly downloaded from the extreme classification repository[1)].

Comparison methods. To validate that the proposed method can readily incorporate any existing large-scale multi-label learning methods, we combine our method with five baseline models including Parabel [27], Bonsai [17], FastXML [13], PfastreXML [16], and X-Transformer [39].

- Parabel recursively partitions label set into two equal-size disjoint subsets and trains one-vs.-rest classifiers for each leaf node of the tree.
- Bonsai attempts to improve Parabel by partitioning the label set into multiple subsets using k-means, resulting in more diverse predictions.

1) http://manikvarma.org/downloads/XC/XMLRepository.html.

• FastXML learns a hyperplane to split similar examples into the same child nodes and makes predictions according to label distributions in leaf nodes.

• PfastreXML is an extension of FastXML, which optimizes propensity-scored objective function to encourage more accurate predictions for tail labels.

• X-Transformer fine-tunes Transformer models to obtain better instance and label representations, rather than using conventional BoWs features.

Evaluation metrics. We evaluate models on the test set and report results with respect to the commonly used evaluation metrics, i.e., P@k, nDCG@k (N@k), PSP@k, and PSnDCG@k (PSN@k), where $k \in \{1, 3, 5\}$. Definitions of used metrics are elaborated in the following. P@k. Top-k precision is a commonly used ranking-based performance measure in long-tailed multi-label learning and has been widely adopted for ranking tasks. In Top-k precision, only a few top predictions of an instance will be considered. For each instance $\boldsymbol{x}$, the Top-k precision is defined for a predicted score vector $\hat{\boldsymbol{y}} \in \mathbb{R}^L$ and ground truth label vector $\boldsymbol{y} \in \{0, 1\}^L$ as

$$\text{P@}k := \frac{1}{k} \sum_{l \in \text{rank}_k(\hat{\boldsymbol{y}})} \boldsymbol{y}_l,$$

where $\text{rank}_k(\hat{\boldsymbol{y}})$ returns the indices of k largest value in $\hat{\boldsymbol{y}}$ ranked in descending order.

nDCG@k. nDCG@k is another commonly used ranking-based performance measure:

$$\text{nDCG@}k := \frac{\text{DCG@}k}{\sum_{l=1}^{\min(k, \|\boldsymbol{y}\|_0)} \frac{1}{\log(l+1)}},$$

where $\text{DCG@}k := \sum_{l \in \text{rank}_k(\hat{\boldsymbol{y}})} \frac{\boldsymbol{y}_l}{\log(l+1)}$ and $||\boldsymbol{y}||_0$ returns the 0-norm of the true-label vector.

PSP@k. Propensity-scored variants of such losses, including precision@k and nDCG@k, are developed and proved to give unbiased estimates of the true loss function even when ground-truth labels go missing under the arbitrary probabilistic label noise models.

$$\text{PSP@}k := \frac{1}{k} \sum_{l \in \text{rank}_k(\hat{\boldsymbol{y}})} \frac{\boldsymbol{y}_l}{p_l}.$$

Here, p_l is the propensity score for label l which helps in making metrics unbiased in the presence of missing labels. Following previous work [16], the propensity score is calculated by

$$p_l = \frac{1}{1 + C\left(N_l + B\right)^{-A}}, \tag{9}$$

where A, B, C are constants.

PSnDCG@k. Similar to nDCG@k, its propensity-scored variant is defined as

$$\text{PSnDCG@}k := \frac{\text{PSDCG@}k}{\sum_{l=1}^{k} \frac{1}{\log(l+1)}},$$

where

$$\text{PSDCG@}k := \sum_{l \in \text{rank}_k(\hat{\boldsymbol{y}})} \frac{\boldsymbol{y}_l}{p_l \log(l+1)}.$$

Implementation details. We use a single-hidden-layer fully connected network for each classifier in the DE module and the hidden layer has 512 neurons. We set the trade-off hyper-parameter in (6) simply as $\lambda = 1$. All models are trained for 30 epochs by the Adam optimizer with a learning rate of 10^{-2} on Nvidia Titan V100 GPUs. Initially, we warm up the networks for 4 epochs by optimizing the BCE loss. We apply dropout [59] with a probability of 0.7. To prevent $\mathcal{L}_{\text{div}}$ from diverging, we apply flooding [60] to $\mathcal{L}_{\text{div}}$ in (3), imposing it to be larger than a constant $b = -10^{-4}$. With respect to the base models, we follow the default settings in their papers. For X-Transformer, we use the pre-trained models released by the authors[2)].

2) https://github.com/OctoberChang/X-Transformer.

Table 3 Comparison results of models with and without applying the proposed RDE method on three datasets

Model	P@1	P@3	P@5	N@1	N@3	N@5	PSP@1	PSP@3	PSP@5	PSN@1	PSN@3	PSN@5
EurLex-4K												
Parabel	82.20	69.28	57.98	82.20	72.66	66.98	36.32	44.41	48.81	36.32	42.22	45.16
w/ RDE	82.83	70.17	58.40	82.83	73.45	67.51	41.68	47.63	49.91	41.68	45.99	47.59
Δ	+0.63	+0.89	+0.42	+0.63	+0.79	+0.53	+5.36	+3.22	+1.10	+5.36	+3.77	+2.43
Bonsai	82.96	69.57	58.18	82.96	73.01	67.29	37.01	45.05	49.45	37.01	42.82	45.78
w/ RDE	83.38	69.89	58.59	83.38	73.32	67.66	42.46	47.47	50.19	42.46	46.06	47.88
Δ	+0.42	+0.32	+0.41	+0.42	+0.31	+0.37	+5.45	+2.42	+0.74	+5.45	+3.24	+2.10
FastXML	70.88	59.25	49.60	70.88	62.30	57.35	26.44	33.59	37.95	26.44	31.64	34.52
w/ RDE	81.31	68.35	57.11	81.31	71.70	66.01	40.63	45.73	48.36	40.63	44.32	46.08
Δ	+10.43	+9.10	+7.51	+10.43	+9.40	+8.66	+14.19	+12.14	+10.41	+14.19	+12.68	+11.56
PfastreXML	70.60	59.16	50.65	70.60	62.10	58.04	43.48	45.38	47.12	43.48	44.88	45.98
w/ RDE	81.41	68.84	57.75	81.41	72.09	66.54	43.75	47.83	49.94	43.75	46.72	48.12
Δ	+10.81	+9.68	+7.10	+10.81	+9.99	+8.50	+0.27	+2.45	+2.82	+0.27	+1.84	+2.14
X-Transformer	86.96	75.20	62.89	86.96	78.41	72.48	38.50	49.09	54.26	38.50	46.17	49.68
w/ RDE	84.60	72.61	61.35	84.60	75.80	70.43	47.50	53.18	55.80	47.50	50.64	52.43
Δ	−2.36	−2.59	−1.54	−2.36	−2.61	−2.05	+9.00	+4.09	+1.54	+9.00	+4.47	+2.75
Wiki10-31K												
Parabel	84.42	72.84	63.81	84.42	75.56	68.61	11.63	12.80	13.77	11.63	12.52	13.19
w/ RDE	85.34	74.17	65.16	85.34	76.77	69.88	16.75	16.63	16.83	16.75	16.67	16.78
Δ	+0.92	+1.33	+1.35	+0.92	+1.21	+1.27	+5.12	+3.83	+3.06	+5.12	+4.15	+3.59
Bonsai	84.63	73.64	64.82	84.63	76.23	69.48	11.89	13.45	14.75	11.89	13.06	13.97
w/ RDE	84.63	74.65	65.70	84.63	76.95	70.19	18.00	17.92	18.19	18.00	17.93	18.09
Δ	0.00	+1.01	+0.88	0.00	+0.72	+0.71	+6.11	+4.47	+3.44	+6.11	+4.87	+4.12
FastXML	82.95	67.86	57.86	82.95	71.29	63.48	9.77	10.28	10.62	9.77	10.16	10.39
w/ RDE	85.05	73.41	64.04	85.05	76.15	68.97	15.04	15.61	16.02	15.04	15.49	15.78
Δ	+2.10	+5.55	+6.18	+2.10	+4.86	+5.49	+5.27	+5.33	+5.40	+5.27	+5.33	+5.39
PfastreXML	75.63	64.78	57.21	75.63	67.29	61.40	18.93	18.41	18.48	18.93	18.55	18.56
w/ RDE	80.96	73.19	64.58	80.96	75.11	68.70	17.22	18.46	18.96	17.22	18.18	18.56
Δ	+5.33	+8.41	+7.37	+5.33	+7.82	+7.30	−1.71	+0.05	+0.48	−1.71	−0.37	0.00
X-Transformer	88.48	78.53	69.05	88.48	80.90	73.75	12.38	13.49	14.56	12.38	13.20	13.94
w/ RDE	86.15	76.99	68.75	86.15	79.12	72.90	18.44	18.15	18.42	18.44	18.22	18.37
Δ	−2.33	−1.54	−0.30	−2.33	−1.78	−0.85	+6.06	+4.66	+3.86	+6.06	+5.02	+4.43
Amazon-670K												
Parabel	46.02	41.05	37.33	46.02	43.41	41.74	26.91	30.91	34.52	26.91	29.87	32.32
w/ RDE	46.03	41.30	37.84	46.03	43.67	42.22	31.06	33.46	35.81	31.06	32.83	34.42
Δ	+0.01	+0.25	+0.51	+0.01	+0.26	+0.48	+4.15	+2.55	+1.29	+4.15	+2.96	+2.10
Bonsai	45.58	40.40	36.62	45.58	42.80	41.08	27.09	30.80	34.14	27.09	29.84	32.10
w/ RDE	45.93	41.13	37.70	45.93	43.51	42.08	31.19	33.45	35.78	31.19	32.86	34.43
Δ	+0.35	+0.73	+1.08	+0.35	+0.71	+1.00	+4.10	+2.65	+1.64	+4.10	+3.02	+2.33
FastXML	35.89	31.79	28.59	35.89	33.63	31.97	18.65	21.83	24.43	18.65	21.01	22.80
w/ RDE	44.23	39.86	36.64	44.23	42.11	40.80	29.83	32.19	34.58	29.83	31.58	33.19
Δ	+8.34	+8.07	+8.05	+8.34	+8.48	+8.83	+11.18	+10.36	+10.15	+11.18	+10.57	+10.39
PfastreXML	36.84	34.23	32.09	36.84	36.01	35.43	29.27	30.80	32.40	29.27	30.40	31.48
w/ RDE	41.32	37.56	34.62	41.32	39.57	38.39	29.31	31.26	33.24	29.31	30.75	32.08
Δ	+4.48	+3.33	+2.53	+4.48	+3.56	+2.96	+0.04	+0.46	+0.84	+0.04	+0.35	+0.60

4.2 Performance comparison

To validate the effectiveness of RDE, we compare the performance of RDE and different base models on three datasets in Table 3. First, performance for all base models w.r.t. vanilla metrics, i.e., P@k and N@k, are significantly improved, indicating the superiority of RDE on head labels. Second, concerning PSP@k and PSN@k, RDE improves the performance of base models in almost all cases. In particular, it obtains performance gains of 5.36%, 5.12%, and 4.15% for Parabel on three datasets w.r.t. PSP@1. For Bonsai, the performance gains are 5.45%, 6.11%, and 4.10%. For FastXML, as much as 14.19%, 5.27%,

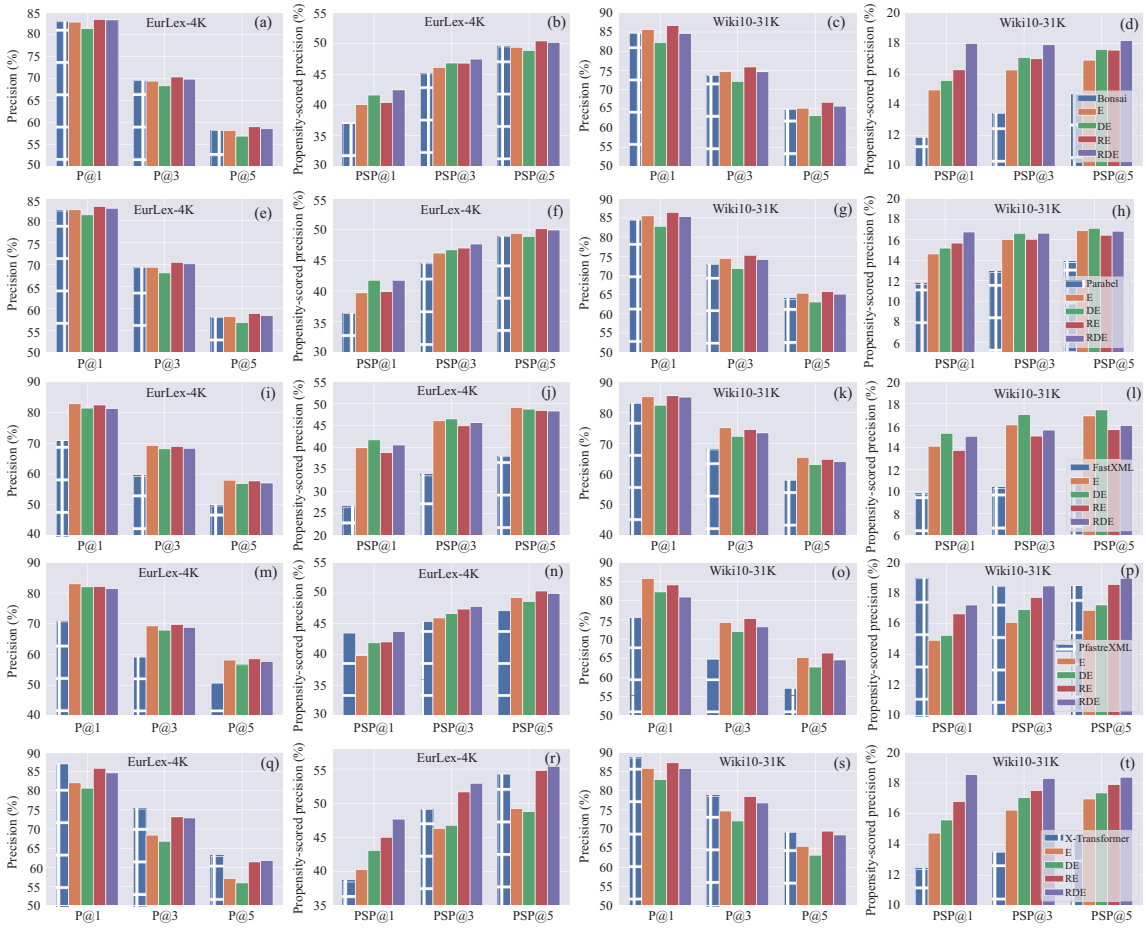

Figure 4 (Color online) Comparison of the baseline models and RDE as well as its three variants. EurLex-4K: (a) Bonsai P@k; (b) Bonsai PSP@k; (e) Parabel P@k; (f) Parabel PSP@k; (i) FastXML P@k; (j) FastXML PSP@k; (m) PfastreXML P@k; (n) PfastreXML PSP@k; (q) X-Transformer P@k; (r) X-Transformer PSP@k. Wiki10-31K: (c) Bonsai P@k; (d) Bonsai PSP@k; (g) Parabel P@k; (h) Parabel PSP@k; (k) FastXML P@k; (l) FastXML PSP@k; (o) PfastreXML P@k; (p) PfastreXML PSP@k; (s) X-Transformer P@k; (t) X-Transformer PSP@k.

and 11.18% improvements are achieved. The improvements w.r.t. PSP@k and PSN@k indicate that RDE can achieve superior performance on tail labels. Notably, PfastreXML achieves high-performance w.r.t. PSP@k and PSN@k at the cost of predictive accuracy because it employs re-ranking to predict tail labels with high probabilities. In most cases, RDE can still improve PSP@k and PSN@k. When integrating X-Transformer with RDE, PSP@k, and PSN@k also have significant improvements. Although the overall evaluation metrics P@k and N@k decline, they remain at high levels. In summary, empirical results show that RDE can effectively improve the performance w.r.t. four evaluation metrics by taking advantage of base models and learning to eliminate residual errors.

4.3 Ablation studies

We conduct ablation studies to gain a better understanding of RDE. We first analyze the contribution of each component proposed in RDE and subsequently show that the proposed residual error correction can lead to fast convergence and better precision. Finally, we investigate the effect of the number of diverse classifiers, the temperature, and the re-weighting variants on model performance.

Contribution of each component. We analyze the effect of each component by comparing RDE with the following four methods: (i) RE removes the DE module and applies random ensemble; (ii) DE removes the residual error correction module and learns from scratch; (iii) E removes the DE and residual error correction modules; and (iv) base models. The comparison results are shown in Figure 4.

The results can be summarized as follows: (i) The DE module improves performance on tail labels because PSP@k of RDE and DE are higher than RE and E, respectively. (ii) The residual error correction module results in more stable performance on head labels because P@k of RDE and RE are higher than

Table 4 Ablation study on the normalized classifiers and the re-weighted classification loss. This table reports the performance changes when removing these two modules from RDE. Experiments are conducted on EurLex-4K with Bonsai as the base model

	P@1	P@3	PSP@1	PSP@3
w/o normalized classifier	−0.26	−0.10	−0.59	−0.05
w/o re-weighting	+0.11	+0.67	−1.88	−0.28

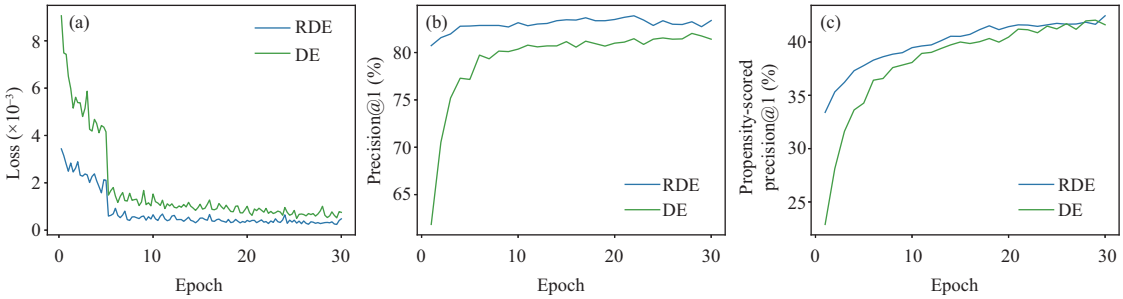

Figure 5 (Color online) Convergence of RDE and its variant DE. (a) Loss; (b) P@1; (c) PSP@1.

Table 5 Impact of the number of classifiers in DE on EurLex-4K with Bonsai as the base model

# Classifier	P@1	P@3	P@5	PSP@1	PSP@3	PSP@5
1	82.23	69.65	58.05	40.69	47.03	49.61
2	83.36	69.97	58.44	41.74	47.49	50.12
3	83.38	69.89	58.59	42.46	47.47	50.19
4	83.43	70.16	58.65	42.07	47.60	50.28

DE and E, respectively. (iii) The benefit of RDE is related to the precision of the base model because when P@k of the base model is inferior to that of vanilla neural networks (E), and the performance gains of P@k and PSP@k of RDE are less significant.

We also investigate the effects of other components in RDE, including the normalized classifiers, and the re-weighted classification loss applied to residual errors. We analyze the influence of these two components by removing them from RDE. The experimental results are presented in Table 4. First, when we remove the normalized classifier (i.e., do not apply L_2 normalization on the classifier's weights $\boldsymbol{W}$), P@k and PSP@k decline, indicating the necessity of using normalized classifiers for long-tailed classification. Second, when the re-weighted classification loss is removed, it is natural to yield slightly higher P@k, but PSP@k decreases more significantly. This indicates that the re-weighted classification loss can improve performance for tail labels without affecting the performance of head labels.

Convergence of learning residual errors. We claim that learning residual errors of base models can lead to faster convergence than fitting ground-truth labels from scratch. Figure 5 shows the curves of the training loss and testing performance as training epochs increase. The results show that RDE trains much faster than DE and converges to good performance in less than 30 epochs.

Impact of the number of diverse classifiers. Table 5 shows the impact of the number of classifiers used in the DE module. When there is only one classifier, the diversity loss does not take effect; therefore, the performance is not superior to the base model. Meanwhile, when more classifiers are used, the DE module obviously makes sense as all metrics, especially PSP@1, have significant improvements. Moreover, for more than three classifiers, the performance gains become limited, and excessive diversity even leads to a performance deterioration on PSP@1. Considering generalization performance and computational cost, we use three classifiers in our proposed method.

Influence of temperature and re-weighting variants. In (3), we set the temperature for the l-th label as the reciprocal of the square root of N_l, i.e., $T_l = \frac{1}{\sqrt{N_l}}$. Here, N_l is the number of the associated training instances. In (2), we set the weight for the l-th label as $w_l = \frac{1}{\sqrt{N_l}}$. One may be concerned regarding the results of other variants than the square root number. We compare the performance with different variants such as $\frac{1}{N_l}$, $\frac{1}{\log(N_l)}$, and $\frac{1}{\log(\log(N_l))}$. Figure 6 shows the function curves for different variants. The results are presented in Table 6. In all cases, the proposed method clearly performs better than the baseline, indicating that the proposed method is not sensitive to different variants. Moreover, the performance gain increases when T_l and w_l are set to $\frac{1}{\sqrt{N_l}}$.

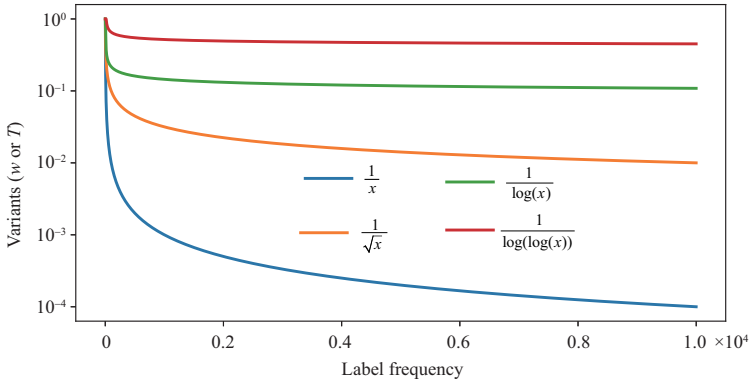

Figure 6 (Color online) Function curves of different variants.

Table 6 Influence of different temperature T_l and re-weighting variants w_l on EurLex-4K with Bonsai as the base model

		P@1	P@3	P@5	PSP@1	PSP@3	PSP@5
Baseline		82.96	69.57	58.18	37.01	45.05	49.45
w_l	$\frac{1}{N_l}$	83.12	69.27	58.11	42.08	46.95	50.08
	$\frac{1}{\sqrt{N_l}}$	83.38	69.89	58.59	42.46	47.47	50.19
	$\frac{1}{\log(N_l)}$	83.41	70.18	58.94	41.67	47.45	50.47
	$\frac{1}{\log(\log(N_l))}$	83.43	70.74	58.96	41.51	47.68	50.48
T_l	$\frac{1}{N_l}$	83.51	70.07	58.61	41.62	47.35	50.12
	$\frac{1}{\sqrt{N_l}}$	83.38	69.89	58.59	42.46	47.47	50.19
	$\frac{1}{\log(N_l)}$	83.41	69.94	58.69	42.30	47.54	50.43
	$\frac{1}{\log(\log(N_l))}$	82.99	69.93	58.58	41.62	47.09	50.11

5 Conclusion

This paper proposes RDE, a novel method designed to improve the performance of long-tailed multi-label text classification, particularly for tail labels. An ensemble of multiple classifiers is learned by jointly optimizing the diversity and classification losses. The proposed RDE considerably improves the generalization performance of tail labels by encouraging divergence among tail label predictions of multiple classifiers. Additionally, RDE approximates the residual errors to prevent performance deterioration on head label predictions, yielding faster and more stable convergence than training from scratch. Moreover, RDE is a general and independent method that can be directly integrated with other arbitrary models as an add-on enhancer module. Finally, extensive experimental results show the effectiveness of RDE, revealing that it can obtain significant performance gains for five base models on three benchmark datasets.

Data availability statement The source code of our method is available at https://www.lamda.nju.edu.cn/code_RDE.ashx and https://github.com/shijxcs/RDE.

Acknowledgements This work was supported by National Key R&D Program of China (Grant No. 2022YFC3340901) and National Natural Science Foundation of China (Grant No. 62176118).

References

1 McAuley J, Pandey R, Leskovec J. Inferring networks of substitutable and complementary products. In: Proceedings of the 21st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Sydney, 2015. 785–794
2 Zubiaga A. Enhancing navigation on Wikipedia with social tags. 2012. ArXiv:1202.5469
3 Medini T K R, Huang Q, Wang Y, et al. Extreme classification in log memory using count-min sketch: a case study of Amazon search with 50m products. In: Proceedings of the Advances in Neural Information Processing Systems, 2019. 13265–13275
4 Zhang M L, Zhou Z H. A review on multi-label learning algorithms. IEEE Trans Knowl Data Eng, 2014, 26: 1819–1837
5 Zhang M L, Zhou Z H. ML-KNN: a lazy learning approach to multi-label learning. Pattern Recogn, 2007, 40: 2038–2048
6 Hsu D J, Kakade S M, Langford J, et al. Multi-label prediction via compressed sensing. In: Proceedings of the Advances in Neural Information Processing Systems, Vancouver, 2009. 772–780
7 Wei T, Guo L Z, Li Y F, et al. Learning safe multi-label prediction for weakly labeled data. Mach Learn, 2018, 107: 703–725
8 Yu H F, Jain P, Kar P, et al. Large-scale multi-label learning with missing labels. In: Proceedings of the 31st International Conference on Machine Learning, Beijing, 2014. 593–601

9 Wei T, Li Y F. Does tail label help for large-scale multi-label learning? IEEE Trans Neural Netw Learn Syst, 2019. doi: 10.1109/TNNLS.2019.2935143
10 Wei T, Shi J X, Li Y F. Probabilistic label tree for streaming multi-label learning. In: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, 2021. 1801–1811
11 Wei T, Mao Z, Shi J X, et al. A survey on extreme multi-label learning. 2022. ArXiv:2210.03968
12 McAuley J, Leskovec J. Hidden factors and hidden topics: understanding rating dimensions with review text. In: Proceedings of the 7th ACM Conference on Recommender Systems, 2013. 165–172
13 Prabhu Y, Varma M. FastXML: a fast, accurate and stable tree-classifier for extreme multi-label learning. In: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York City, 2014. 263–272
14 Babbar R, Schölkopf B. DiSMEC: distributed sparse machines for extreme multi-label classification. In: Proceedings of the 10th ACM International Conference on Web Search and Data Mining, Cambridge, 2017. 721–729
15 Jiang T, Wang D, Sun L, et al. LightXML: transformer with dynamic negative sampling for high-performance extreme multi-label text classification. In: Proceedings of the AAAI Conference on Artificial Intelligence, 2021. 7987–7994
16 Jain H, Prabhu Y, Varma M. Extreme multi-label loss functions for recommendation, tagging, ranking & other missing label applications. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, 2016. 935–944
17 Khandagale S, Xiao H, Babbar R. Bonsai: diverse and shallow trees for extreme multi-label classification. Mach Learn, 2020, 109: 2099–2119
18 Bi W, Kwok J T. Efficient multi-label classification with many labels. In: Proceedings of International Conference on Machine Learning, Atlanta, 2013. 405–413
19 Niculescu-Mizil A , Abbasnejad M E. Label filters for large scale multi-label classification. In: Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, Fort Lauderdale, 2017. 1448–1457
20 Yen I E, Huang X, Dai W, et al. PPDsparse: a parallel primal-dual sparse method for extreme classification. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, 2017. 545–553
21 Fang H, Cheng M, Hsieh C J, et al. Fast training for large-scale one-versus-all linear classifiers using tree-structured initialization. In: Proceedings of the 2019 SIAM International Conference on Data Mining, 2019. 280–288
22 Babbar R, Schölkopf B. Data scarcity, robustness and extreme multi-label classification. Mach Learn, 2019, 108: 1329–1351
23 Jasinska K, Karampatziakis N. Log-time and log-space extreme classification. 2016. ArXiv:1611.01964
24 Daume III H, Karampatziakis N, Langford J, et al. Logarithmic time one-against-some. 2016. ArXiv:1606.04988
25 Jasinska K, Dembczynski K, Busa-Fekete R, et al. Extreme f-measure maximization using sparse probability estimates. In: Proceedings of the 33rd International Conference on Machine Learning, New York City, 2016. 1435–1444
26 Si S, Zhang H, Keerthi S S, et al. Gradient boosted decision trees for high dimensional sparse output. In: Proceedings of International Conference on Machine Learning, 2017. 3182–3190
27 Prabhu Y, Kag A, Harsola S, et al. Parabel: partitioned label trees for extreme classification with application to dynamic search advertising. In: Proceedings of the World Wide Web Conference, 2018. 993–1002
28 Siblini W, Kuntz P, Meyer F. CRAFTML, an efficient clustering-based random forest for extreme multi-label learning. In: Proceedings of the 35th International Conference on Machine Learning, 2018. 4664–4673
29 Kapoor A, Viswanathan R, Jain P. Multilabel classification using bayesian compressed sensing. In: Proceedings of the Advances in Neural Information Processing Systems, Lake Tahoe, 2012. 2645–2653
30 Bhatia K, Jain H, Kar P, et al. Sparse local embeddings for extreme multi-label classification. In: Proceedings of the Advances in Neural Information Processing Systems, Montreal, 2015. 730–738
31 Xu C, Tao D C, Xu C. Robust extreme multi-label learning. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, 2016. 1275–1284
32 Yeh C K, Wu W C, Ko W J, et al. Learning deep latent space for multi-label classification. In: Proceedings of the 31st AAAI Conference on Artificial Intelligence, San Francisco, 2017. 2838–2844
33 Tagami Y. AnnexML: approximate nearest neighbor search for extreme multi-label classification. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, 2017. 455–464
34 Evron I, Moroshko E, Crammer K. Efficient loss-based decoding on graphs for extreme classification. In: Proceedings of the Advances in Neural Information Processing Systems, 2018. 7233–7244
35 Liu W, Shen X. Sparse extreme multi-label learning with oracle property. In: Proceedings of the 36th International Conference on Machine Learning, 2019. 4032–4041
36 Liu J, Chang W C, Wu Y, et al. Deep learning for extreme multi-label text classification. In: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2017. 115–124
37 Zhang W, Yan J, Wang X, et al. Deep extreme multi-label learning. In: Proceedings of the 2018 ACM on International Conference on Multimedia Retrieval, Yokohama, 2018. 100–107
38 You R, Zhang Z, Wang Z, et al. AttentionXML: label tree-based attention-aware deep model for high-performance extreme multi-label text classification. 2019. ArXiv:1811.01727
39 Chang W C, Yu H F, Zhong K, et al. Taming pretrained transformers for extreme multi-label text classification. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2020. 3163–3171
40 Guo C, Mousavi A, Wu X, et al. Breaking the glass ceiling for embedding-based classifiers for large output spaces. In: Proceedings of Advances in Neural Information Processing Systems, 2019. 4944–4954
41 Xun G, Jha K, Sun J, et al. Correlation networks for extreme multi-label text classification. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2020. 1074–1082
42 Hochreiter S, Schmidhuber J. Long short-term memory. Neural Comput, 1997, 9: 1735–1780
43 Wei T, Tu W W, Li Y F. Learning for tail label data: a label-specific feature approach. In: Proceedings of the 28th International Joint Conference on Artificial Intelligence, Macau, 2019. 3842–3848
44 Kang B, Xie S, Rohrbach M, et al. Decoupling representation and classifier for long-tailed recognition. In: Proceedings of the International Conference on Learning Representations, 2020
45 Wei T, Shi J X, Tu W W, et al. Robust long-tailed learning under label noise. 2021. ArXiv:2108.11569
46 Wei T, Tu W W, Li Y F, et al. Towards robust prediction on tail labels. In: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, 2021. 1812–1820
47 Zhu Y, Zhuang F, Zhang X, et al. Combat data shift in few-shot learning with knowledge graph. Front Comput Sci, 2023, 17: 171305
48 Ji Z, Ni J, Liu X, et al. Teachers cooperation: team-knowledge distillation for multiple cross-domain few-shot learning. Front

Comput Sci, 2023, 17: 172312
49 Xue Z, Du J, Xu X, et al. Few-shot node classification via local adaptive discriminant structure learning. Front Comput Sci, 2023, 17: 172316
50 Guo L Z, Li Y F. Robust pseudo-label selection for holistic semi-supervised learning (in Chinese). Sci Sin Inform, 2014, 54: 623–637
51 Jia L H, Guo L Z, Zhou Z, et al. LAMDA-SSL: a comprehensive semi-supervised learning toolkit. Sci China Inf Sci, 2024, 67: 117101
52 Wang X, Lian L, Miao Z, et al. Long-tailed recognition by routing diverse distribution-aware experts. In: Proceedings of the International Conference on Learning Representations, 2021
53 Cui J, Liu S, Tian Z, et al. ResLT: residual learning for long-tailed recognition. IEEE Trans Pattern Anal Mach Intell, 2023, 45: 3695–3706
54 Cui Y, Jia M, Lin T Y, et al. Class-balanced loss based on effective number of samples. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019. 9268–9277
55 Cao K, Wei C, Gaidon A, et al. Learning imbalanced datasets with label-distribution-aware margin loss. In: Proceedings of the Advances in Neural Information Processing Systems, 2019. 1565–1576
56 Menon A K, Jayasumana S, Rawat A S, et al. Long-tail learning via logit adjustment. In: Proceedings of the International Conference on Learning Representations, 2021
57 Zhang Y, Hooi B, Hong L, et al. Self-supervised aggregation of diverse experts for test-agnostic long-tailed recognition. In: Proceedings of the Advances in Neural Information Processing Systems, 2022. 34077–34090
58 Wei T, Wang H, Tu W W, et al. Robust model selection for positive and unlabeled learning with constraints. Sci China Inf Sci, 2022, 65: 212101
59 Srivastava N, Hinton G, Krizhevsky A, et al. Dropout: a simple way to prevent neural networks from overfitting. J Mach Learn Res, 2014, 15: 1929–1958
60 Ishida T, Yamane I, Sakai T, et al. Do we need zero training loss after achieving zero training error? 2020. ArXiv:2002.08709