Investigating the Sample Weighting Mechanism Using an Interpretable Weighting Framework

Xiaoling Zhou, Ou Wu, and Mengyang Li

Abstract—Training deep learning models with unequal sample weights has been shown to enhance model performance in various typical learning scenarios, particularly for imbalanced and noisy-label learning scenarios. A deep understanding of the weighting mechanism facilitates the application of existing weighting strategies and illuminates the design of new weighting strategies for real learning tasks. Scholars have focused on exploring existing weighting methods. However, their studies mainly establish how the weights of samples influence the model training. Little headway is made on the weighting mechanism, i.e., which and how the characteristics of a sample influence its weight. In this study, we adopt a data-driven approach to investigate the weighting mechanism by utilizing an interpretable weighting framework. First, a wide range of sample characteristics is extracted from the classifier network during training. Second, the extracted characteristics are fed into a new neural regression tree (NRT), which is a tree model implemented by a neural network, and its output is the weight of the input sample. Third, the NRT is trained using meta-learning within the whole training process. Once the NRT is learned, the weighting mechanism, including the importance of weighting characteristics, prior modes, and specific weighting rules, can be obtained. We conduct extensive experiments on benchmark noisy and imbalanced data corpora. A package of weighting mechanisms is derived from the learned NRT. Furthermore, our proposed interpretable weighting framework exhibits superior performance in comparison to existing weighting strategies.

Index Terms—Sample weighting, interpretability, neural regression tree, meta-learning.

1 Introduction

In addition to the design of new deep neural networks, the design of unequal weights on the losses of training samples is also a common technique to improve model performance in various machine learning tasks. Different weighting strategies are proposed based on diverse empirical observations or theoretical inspirations. For example, the typical weighting strategy called Focal Loss [1] was motivated by the observation that background (easy) samples are overrepresented in object detection datasets relative to foreground (hard) samples. Thus, this weighting strategy assigns relatively high weights on hard samples and relatively low weights on easy ones. The experimental results show that detection performance was improved in terms of accuracy. The classical shallow classification method, called AdaBoost [2], also exerts high weights on hard samples. By contrast, the weighting strategies in Curriculum Learning (CL) [3] and Self-paced Learning (SPL) [4] set high weights on easily classified samples. Studies have verified that the easy-first paradigm CL mainly takes effect on noisy datasets [17]. In addition, given that noisy samples are generally hard ones, SuperLoss [56] also assigns higher weights to easy samples and lower weights to hard ones, which yields favorable performance in noisy-label learning scenarios. These two approaches, characterized by assigning high weights to either hard or easy samples, adopt fundamentally contrasting strategies, yet both demonstrate effectiveness in specific learning scenarios. As a result, the current understanding regarding the prioritization of samples for learning remains inconclusive. Besides, samples in small categories are generally hard ones and are assigned with high weights in long-tailed classification [5], [6]. In some cost-sensitive tasks, such as fraud detection [54] and medical diagnosis [55], samples having large gains or costs are allocated higher weights. Moreover, samples that play a greater role in enhancing the training and generalization abilities of the model are assigned higher weights. For example, in data augmentation, the weights assigned to original samples are generally larger than those assigned to augmented ones [47].

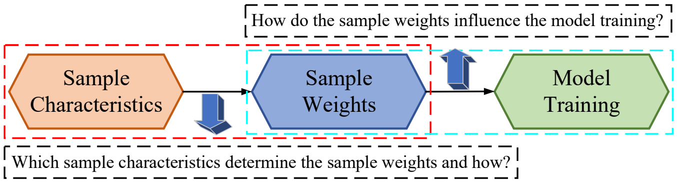

How do the sample weights influence the model training?
Sample Characteristics
Sample Weights
Model Training
Which sample characteristics determine the sample weights and how?

Fig. 1. Two issues of understanding weighting strategies.

In summarizing of existing studies, the mainstream approaches for sample weighting can be categorized into two categories. The first category encompasses heuristically defined weighting functions, such as Focal Loss [1], SuperLoss [56], SPL [4], etc. These methods explicitly model the sample weights as a function of the sample characteristics. For example, Class-Balanced Loss [5] formulates weights as a function of the category distribution, Focal Loss assigns weights based on sample predictions, while SPL and SuperLoss determine weights based on the sample loss. However, a consensus regarding which weighting characteristics more effectively determine sample weights and the underlying mechanisms governing their determination has yet to be

- *Xiaoling Zhou, Ou Wu, and Mengyang Li are with the Center for Applied Mathematics, Tianjin University, Tianjin, China, 300072.*
E-mail: xiaolingzhou/wuou/limengyang@tju.edu.cn

Manuscript received September 13, 2023.

reached. The second category contains meta-learning-based weighting strategies [7], [8], [9], which have demonstrated state-of-the-art performance compared to heuristically defined weighting functions. For instance, Meta-weight-net [7] leverages the loss as an input to the weighting network and generates weights based on the usage of meta-learning. Nevertheless, existing meta-learning-based weighting methods often rely on a single or very limited set of indicators as inputs to the weighting network. Furthermore, the weighting network is typically a black box model, lacking interpretability, thereby posing a substantial obstacle to the thorough investigation of sample weights.

The understanding of sample weights can be divided into two issues, as shown in Fig. 1. The first issue represents the mechanism of weighting, i.e., knowing which sample characteristics determine the weight of a sample and how. The second issue is concerned with the role of weighting, i.e., knowing how the sample weights influence the model training. Thus far, the previous studies pertaining to the understanding of sample weights in deep learning mainly focused on the second issue. For example, Byrd and Lipton [10] investigated importance weighting which is the likelihood of the densities of the target and source distributions and found that sample weights affect deep model training by influencing the implicit bias of gradient descent. On the basis of their finding, Xu et al. [11] revealed the role of sample weights on the optimization dynamics and generalization performance for deep learning under certain rigorous assumptions.

Few studies have attempted to address the first issue, which is investigating the weighting mechanism. Consequently, numerous significant problems remain unsolved. We summarize the unsolved but widely concerned problems as follows:

(i) Different weighting methods rely on distinct characteristics of samples, such as loss, prediction, margin, etc. Which types of characteristics are more effective than others?

(ii) Nearly all existing weighting strategies only utilize a single weighting characteristic. Whether and How does the combination of multiple weighting characteristics yield better performance?

(iii) In some schemes, hard samples[1] are assigned with high weights, which is called the hard-first mode [1], [5]. In some other schemes, easy samples have higher weights than hard ones, which is called the easy-first mode [3], [4]. Both modes are claimed to be effective in previous studies. Consequently, which samples should be learned first [12], easy or hard ones?

(iv) The prior mode[2] (easy-first or hard-first) is fixed during the training process in existing heuristic strategies. Are there any other effective prior modes?

To solve the above questions, this study attempts to design a data-driven method to investigate the weighting mechanism of a good set of sample weights. For this purpose, an interpretable weighting framework is constructed to infer the sample weights with an interpretable neural regression tree (NRT). Aiming to ensure a “good” set of weights, meta-learning [7] is utilized based on an additional meta dataset. Specifically, a wide range of sample characteristics in the previous studies is first extracted from the classifier. These characteristics are then fused and fed into our proposed NRT. Finally, the NRT is trained with meta-learning to achieve a good set of sample weights. As the NRT can naturally yield concrete weighting rules from the input sample characteristics to the output weights, the weighting rules can also be obtained once the NRT is trained. Our approach possesses notable advantages in contrast to previous weighting strategies:

- In contrast to existing strategies that employ only a single or very limited weighting characteristic, our method employs a wider range of sample characteristics to infer sample weights.
- The weighting network in our framework is an NRT, which possesses inherent interpretability, whereas other meta-learning-based approaches employ black box models as the weighting networks. Consequently, the weighting mechanisms can be readily explained by our framework.
- Our framework obviates the necessity of designing an explicit weighting function, instead opting for a data-driven approach to ascertain sample weights. As a result, our framework can accommodate all feasible optimal prior modes. Furthermore, the prior mode exhibits the potential to dynamically change throughout the training process.

A package of weighting mechanisms has been revealed in our experiments. First, the importance of weighting characteristics in some typical occasions is analyzed. These important characteristics can be adopted preferentially in future weighting strategies. Second, the hybrid prior mode is observed from the learned weights, referring to both the easy-first and the hard-first modes utilized by the partial samples, while existing weighting strategies adopt a single prior pattern for all samples. Third, the shifting prior mode is discovered. Alternatively, the transition from easy (hard)-first to hard (easy)-first may occur during training, while existing methods maintain the same prior mode throughout the training process. In addition, concrete weighting rules are obtained from the learned NRT. Furthermore, the experimental results indicate that our proposed interpretable weighting framework can achieve superior performance compared with other advanced weighting strategies.

Our contributions can be summarized as follows:

- A novel interpretable weighting framework is initiated to investigate the weighting mechanism in deep learning. To the best of our knowledge, this work is the first one to apply interpretability techniques to facilitate the understanding of an important training component (i.e., sample weighting). Our framework can also be utilized to investigate the mechanism of other training components.
- A package of weighting mechanisms, including the importance of weighting characteristics, major prior modes, and specific weighting rules, is achieved.

[1] The learning difficulty of a training sample is usually approximated by the sample characteristics, such as loss, margin, etc.

[2] Prior mode means that a weighting strategy assigns higher weights to specific types of samples than other ones in training.

Heuristic priors used in previous studies are observed. Moreover, valuable findings undetected in previous studies are obtained.

- An NRT is proposed. Taking the interpretable tree as the weighting network, our framework can achieve superior performance with excellent interpretability compared with employing black box models as the weighting network, which is inexplicable.

2 Related Work

2.1 Learning with Sample Weights

Samples differ from each other because of their differences in aspects, including data quality [13], [14], sample neighbors [15], [16], and category distribution [5], [6]. Treating each training sample independently can improve learning performance in many machine learning tasks [1], [17]. Numerous weighting schemes have been proposed, which are based on various weighting characteristics related to the above-mentioned differences. However, these strategies only rely on a single or very limited weighting characteristic, thereby falling short in their ability to capture the diverse attributes of samples. The most frequently used characteristic is loss (or the predicted probability of ground truth). Three popular weighting methods, namely SPL [4], Focal Loss [1], and SuperLoss [56] are both based on the loss. Some weighting schemes are also based on other characteristics, such as margin [16], loss gradient [52], uncertainty [19], and category proportion [5], [6]. However, which characteristics are more effective than others remains unclear. In contrast to existing weighting methods that rely on one or very limited characteristics, our approach leverages multiple dimensions of sample characteristics, such as loss, margin, uncertainty, etc. Consequently, it enables more precise assignment of sample weights based on their unique characteristics.

According to the prior mode principle, existing weighting methods can be divided into two categories, namely easy-first and hard-first. Some typical easy-first weighting methods have achieved success, especially for noisy-label learning occasions [3], [56]. For example, CL [3] is inspired by human learning in which easy samples should be learned first. An empirical study conducted by Wu et al. [17] verified that CL mainly takes effect under noisy scenarios. SPL [4] also belongs to the easy-first mode, which sets the weights of hard samples to zero with a threshold. The threshold is gradually increased to ensure that more hard samples can participate in the next epochs of training. Meanwhile, there are also some weighting strategies belonging to the hard-first mode. These strategies behave well in imbalanced learning scenarios. Class-Balanced Loss [5] and Weighted Broad Learning System [20] exert high weights on samples in the minority categories, which are generally hard ones. Learning Optimal Weights [52] forces a model to focus on less represented or more challenging samples. However, current heuristic approaches are constrained by the adoption of either an easy-first or hard-first strategy exclusively during the entire training procedure, thus restricting their applicability to specific learning scenarios. In contrast, our proposed framework offers a dynamic and flexible solution that can accommodate both easy-first and hard-first modes. As a result, our method exhibits broad versatility across diverse learning scenarios, including imbalanced learning and noisy-label learning.

Nowadays, meta-learning-based weighting methods have achieved state-of-the-art performance. MentorNet [9] learns a data-driven curriculum of StudentNet by training a teacher network, while L2RW [8] learns to assign weights to samples based on their gradient directions. MetaReg [21] learns the weights of noisy samples using meta-learning, which is a powerful regulation algorithm. Shu et al. [7] adopted an online learning strategy to alternatively update parameters in a classifier network and a sample weighting network. The sample weighting network is a multilayer perception network (MLP) and its input is only loss. Notably, the weighting networks in previous meta-learning-based approaches are black box models, rendering them incapable of explaining the fundamental mechanisms underpinning sample weighting. In contrast, our proposed framework employs an interpretable NRT as the weighting network. Consequently, we are able to systematically investigate the intrinsic mechanisms that govern sample weighting.

2.2 Understanding the Weighting Strategies

Only a few studies have attempted to understand the weighting strategies. These studies mainly focused on exploring the influence of sample weights on model training. For example, Byrd and Lipton [10] empirically found that importance weighting influences deep learning models by affecting the implicit bias of gradient descent [22]. Xu et al. [11] investigated how the theoretical understanding of implicit bias of gradient descent adjusts to the weighted empirical risk minimization. However, their theoretical analysis was based on some rigorous assumptions, resulting in their conclusions being limited and inappropriate in some cases. Few studies have attempted to investigate the weighting mechanism or know which sample characteristics influence the sample weights and how. Our study fills this gap by thoroughly investigating the mechanism of sample weighting. Specifically, we delve into the critical aspects of the weighting characteristics that govern the determination of sample weights and shed light on the mechanisms.

2.3 Rule-Based Deep Learning Interpretability

In building trust in intelligent systems, "transparent" models must be built to explain why they predict what they predict. Compared with deep learning models, the prediction of a rule-based model is easier to explain because its prediction depends on a short sequence of rules, and each rule is directly based on the input data. The alternative rule model [23] explains the behavior of black box models by selecting the rule with a prediction that is closest to the black box model based on a pre-verifiable finite rule set. Some studies have also combined neural networks and decision trees. Frosst and Hinton [24] utilized the neural network to train decision trees that mimic the input-output function discovered by the neural network. Neural-Backed Decision Trees [25] replace the final layer of a neural network with a differentiable sequence of decisions and a surrogate loss. However, the aforementioned methods only have limited explainability, and they are not suitable for

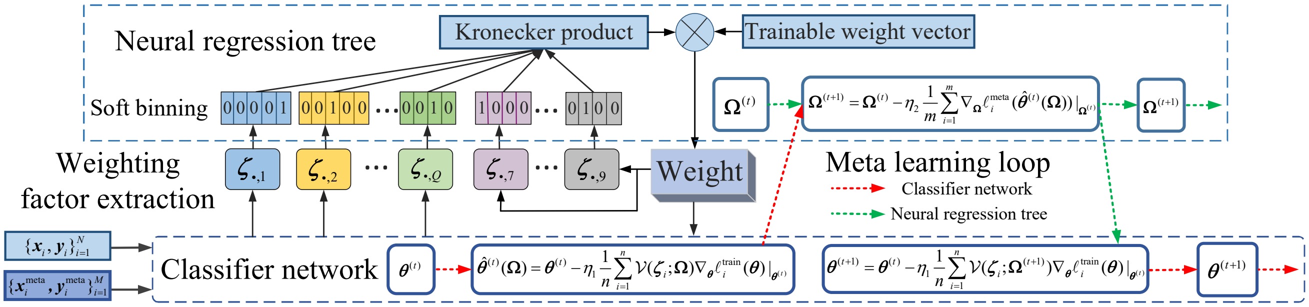

Neural regression tree
Kronecker product
Trainable weight vector
Soft binning
0 0 0 0 1
0 0 1 0 0
0 0 1 0
1 0 0 0
0 1 0 0
$\Omega^{(t)}$
$\Omega^{(t+1)} = \Omega^{(t)} - \eta_2 \frac{1}{m}\sum_{i=1}^{m} \nabla_{\Omega} \ell_i^{\text{meta}}(\hat{\theta}^{(t)}(\Omega))|_{\Omega^{(t)}}$
$\Omega^{(t+1)}$
Weighting factor extraction
$\zeta_{\cdot,1}$
$\zeta_{\cdot,2}$
$\zeta_{\cdot,Q}$
$\zeta_{\cdot,7}$
$\zeta_{\cdot,9}$
Weight
Meta learning loop
Classifier network
Neural regression tree
$\{x_i, y_i\}_{i=1}^N$
$\{x_i^{\text{meta}}, y_i^{\text{meta}}\}_{i=1}^M$
Classifier network
$\theta^{(t)}$
$\hat{\theta}^{(t)}(\Omega) = \theta^{(t)} - \eta_1 \frac{1}{n}\sum_{i=1}^{n} \mathcal{V}(\zeta_i;\Omega)\nabla_{\theta} \ell_i^{\text{train}}(\theta)|_{\theta^{(t)}}$
$\theta^{(t+1)} = \theta^{(t)} - \eta_1 \frac{1}{n}\sum_{i=1}^{n} \mathcal{V}(\zeta_i;\Omega^{(t+1)})\nabla_{\theta} \ell_i^{\text{train}}(\theta)|_{\theta^{(t)}}$
$\theta^{(t+1)}$

Fig. 2. The overall structure of our proposed interpretable weighting framework.

tabular data[3]. Deep Neural Decision Tree (DNDT) [26] is a tree classification model realized by deep neural networks, and it is intrinsically interpretable. In addition, DNDT can sufficiently fit tabular data. However, the structure of DNDT is fixed, which is inefficient and seriously damages its interpretability. In this study, an NRT with a variable structure that is trained with gradient descent is proposed to achieve an effective and interpretable weighting network, which is suitable for tabular data with inherent interpretability.

3 METHODOLOGIES

This section first introduces the overall configuration of our proposed interpretable weighting framework. Its three important modules are then detailed.

3.1 The Proposed Interpretable Weighting Framework

The core element of nearly all existing weighting methods is either a mathematical function heuristically defined with prior knowledge or a black box model such as the MLP network learned using meta data [1], [4], [7]. The input of both methods is one or no more than three limited weighting characteristics in existing studies. For a heuristically defined mathematical function, it is usually simple and has strong interpretability. However, its performance is inferior to the black box model because it heavily relies on prior knowledge or assumptions. The black box model is generally learned via meta-learning based on an additional meta dataset. However, its interpretability is poor.

Inspired by the strong interpretability of the decision tree and the competitive performance of the pure data-driven mode of meta-learning [7], an interpretable weighting framework is constructed as shown in Fig. 2. Our framework consists of four main parts, namely, the backbone classifier network, weighting characteristic extraction module, NRT-based weighting network, and meta-learning optimization, which optimizes the entire parameters of both the classifier network and the NRT.

Our interpretable weighting framework differs from existing weighting methods. First, the primary goal of our framework is to acquire the weighting mechanism in deep learning whereas other weighting methods are mainly concerned with model performance. Undoubtedly, a thorough understanding of the weighting mechanism can facilitate the improvement of model performance. Second, the core weighting network of our framework is an interpretable model, whereas those of existing meta-learning-based methods are nearly inexplicable as they adopt the black box model, such as MLP, as the weighting network. Third, no prior knowledge or inspiration is required in our framework, whereas implicit or explicit assumptions are needed by most mathematical weighting functions. In addition, more weighting characteristics are extracted and input into our weighting network which can contribute to a more effective weighting function, while existing methods only consider one or part of them.

The next three subsections will introduce the weighting characteristic extraction, the NRT-based weighting network, and the entire training with meta-learning.

3.2 Weighting Characteristic Extraction

This module aims to extract quantifiable characteristics of samples in each training iteration as a means of sufficiently determining the weight of each training sample. Then, the weighting network assigns weights to samples according to these extracted characteristics. First, the characteristics employed in the core weighting functions in previous studies are inherited in this study. Six characteristics ($\boldsymbol{\zeta}_{\cdot,1}, \boldsymbol{\zeta}_{\cdot,2}, \cdots, \boldsymbol{\zeta}_{\cdot,6}$) are considered and extracted first, which can be described as follows.

(1) Loss ($\boldsymbol{\zeta}_{\cdot,1}$) is the most widely used characteristic in existing weighting strategies [4], [7], [56]. For example, SPL [4] and SuperLoss [56] decrease the weights of samples with large losses, which behave well on noisy datasets, while Focal Loss [1] assigns high weights to these samples.

(2) Margin ($\boldsymbol{\zeta}_{\cdot,2}$) refers to the distance from the sample to the classification boundary, and it is usually used to measure the difficulty of samples [16]. It can be calculated by

$$\zeta_{i,2} = f(\boldsymbol{x}_i)_{y_i} - \max_{j \neq y_i}(f(\boldsymbol{x}_i)_j), \tag{1}$$

where $f(\boldsymbol{x}_i)$ and y_i are the output of the classifier after Softmax and the label of sample $\boldsymbol{x}_i$, respectively.

(3) Gradient norm ($\boldsymbol{\zeta}_{\cdot,3}$) of $f(\boldsymbol{x}_i)$ is another commonly used weighting characteristic [52]. As the Cross-Entropy (CE) loss is adopted in our framework, it is calculated by

$$\zeta_{i,3} = ||\boldsymbol{y}_i - f(\boldsymbol{x}_i)||_2, \tag{2}$$

where $\boldsymbol{y}_i$ is the one-hot label vector of sample $\boldsymbol{x}_i$.

(4) Information entropy ($\boldsymbol{\zeta}_{\cdot,4}$) of $f(\boldsymbol{x}_i)$ is used to measure the uncertainty of training samples [27]. Its calculation is

$$\zeta_{i,4} = -\sum_{j=1}^{C} f(\boldsymbol{x}_i)_j \log(f(\boldsymbol{x}_i)_j), \tag{3}$$

[3] The input of the weighting network in our proposed interpretable weighting framework is the weighting characteristics of samples which are tabular data.

where C is the number of categories.

(5) Category proportion ($\boldsymbol{\zeta}_{\cdot,7}$) is commonly used to handle imbalanced category distribution [5], [6]. It can be calculated by

$$\boldsymbol{\zeta}_{i,7} = N_{y_i}/N, \tag{4}$$

where N_{y_i} and N are the numbers of samples in category y_i and in the entire training set, respectively.

(6) Average loss of each category ($\boldsymbol{\zeta}_{\cdot,6}$) is another category-level weighting characteristic used to indicate the average learning difficulty of a category. Its calculation is

$$\boldsymbol{\zeta}_{i,6} = \bar{\ell}_{y_i}, \tag{5}$$

where $\bar{\ell}_{y_i}$ is the average loss of samples in category y_i.

Besides the abovementioned six characteristics, three characteristics related to the weights of samples in previous rounds are also employed, including the weight of the last epoch ($\boldsymbol{\zeta}_{\cdot,7}$), the weight of the last but one epoch ($\boldsymbol{\zeta}_{\cdot,8}$), and the difference between the weight of the last epoch and the last but one epoch ($\boldsymbol{\zeta}_{\cdot,9}$).

To further enhance the performance of our framework, the sequence extensions of the first six characteristics can also be considered[4]. Specifically, we can integrate the discrepancy between the values of each characteristic from the previous iteration to the current iteration. Our employed NRT to be introduced in the next subsection can automatically perform the characteristic selection. Therefore, irrelevant or redundant characteristics in the input can be discarded during the tree training.

3.3 NRT-Based Weighting Network

Due to the superiority in interpretability, the decision tree is the primary choice for our weighting network. DNDT [26] offers a way to train a tree model with stochastic gradient descent (SGD) for tabular input data. However, three defects arise when DNDT is directly utilized in our framework. First, DNDT is a classification tree, but our weighting module requires a regression model. Second, the structure of DNDT is fixed during training, which equals to a perfect k-ary tree. The fixed structure creates numerous redundant nodes within DNDT and seriously impairs its interpretability. Third, the number of cut points for each characteristic is manually determined, which is obviously unreasonable and inefficient. Aiming to construct a more effective tree model for our weighting module, an NRT with a variable structure is proposed, in which the number of cut points of each characteristic is learned by the model.

Following DNDT, NRT still replaces the hard binning employed in conventional decision trees with a soft binning function $\pi(\cdot)$, which is a one-layer neural network with Softmax as its activation function.

$$\pi\left(\boldsymbol{\zeta}_{i,j}\right) = \text{Softmax}\left[\left(\mathbf{w}\boldsymbol{\zeta}_{i,j} + \boldsymbol{b}\right)/\tau\right], \tag{6}$$

where $\boldsymbol{\zeta}_{i,j}$ refers to the j-th weighting characteristic of sample $\boldsymbol{x}_i$ which is a scalar; $\mathbf{w} = [1, 2, \cdots, a_j + 1]$, where a_j is the number of cut points of $\boldsymbol{\zeta}_{\cdot,j}$; $\boldsymbol{b}$ is a trainable vector constructed as $\boldsymbol{b} = [0, -\beta_{j,1}, -\beta_{j,1} - \beta_{j,2}, \cdots, -\beta_{j,1} - \beta_{j,2} - \cdots - \beta_{j,a_j}]$, where $\beta_{j,1}$ to β_{j,a_j} are a_j cut points of $\boldsymbol{\zeta}_{\cdot,j}$ with the constraint that $\beta_{j,1} < \beta_{j,2} < \cdots < \beta_{j,a_j}$; and τ is a temperature factor. As $\tau \to 0$, then $\pi(\boldsymbol{\zeta}_{i,j})$ tends to be a one-hot vector. For example, if the loss characteristic $\boldsymbol{\zeta}_{\cdot,1}$ is divided into three intervals and the two cut points are denoted as $\beta_{1,1}$ and $\beta_{1,2}$, then the one-hot vector $\pi(\boldsymbol{\zeta}_{i,1}) = [1, 0, 0]$ implies that $\boldsymbol{\zeta}_{i,1} < \beta_{1,1}$.

Algorithm 1: Learning of NRT

```
Input: weighting characteristics $\boldsymbol{\zeta}_{\cdot,j}|_{j=1}^{Q}$ for training
       samples, maximum number of cut points $c$,
       thresholds $\varepsilon_+$ and $\varepsilon_-$, maximum iterations $T$,
       record iterations $T_r$, temperature $\tau$.
Output: Learned parameters of NRT $\boldsymbol{\Omega}$.
Initialize $\boldsymbol{\Omega}^{(1)}$, $\beta_{j,1,1}|_{j=1}^{Q}$, and $a_{j,1}|_{j=1}^{Q} = 1$;
for $t = 1$ to $T$ do
    Bin each weighting characteristic $\boldsymbol{\zeta}_{\cdot,j}$ by Eq. (6);
    Calculate all possible leaf nodes by Eq. (7);
    Calculate $\mathcal{V}(\boldsymbol{\zeta}^{(t)}; \boldsymbol{\Omega}^{(t)})$ of the input samples;
    Record the values of $\beta_{j,k,t}|_{k=1}^{a_{j,t}}, j = 1, \cdots, Q$;
    if $t\%T_r == 0$ then
        Calculate $s_{j,k,t}|_{k=1}^{a_{j,t}}$ & $d_{j,k,t}|_{k=1}^{a_{j,t}-1}, j = 1,\cdots,Q$
          by Eqs. (8) and (10);
        Calculate the numbers of samples between all
          adjacent activated cut points
          $\hat{n}_{j,k,t}|_{k=1}^{a_{j,t}-1}, j = 1,\cdots,Q$;
        Remove the recorded values of
          $\beta_{j,k,p}|_{p=t-T_r,k=1,j=1}^{p=t,k=a_{j,p},j=Q}$;
        for $j = 1$ to $Q$ do
            if $max_{k=1}^{a_{j,t}} s_{j,k,t} < \varepsilon_+$ & $a_{j,t} < c$ then
                activate and initialize
                  $\beta_{j,a_{j,t}+1}$ & $a_{j,t} = a_{j,t} + 1$;
            end
            for $k = 1$ to $a_{j,t} - 1$ do
                if $d_{j,k,t} < \varepsilon_-$ & $a_{j,t} > 0$ then
                    deactivate $\beta_{j,k,t}$ & $a_{j,t} = a_{j,t} - 1$;
                end
                if $\hat{n}_{j,k,t} == 0$ & $a_{j,t} > 0$ then
                    deactivate $\beta_{j,k,t}$ & $a_{j,t} = a_{j,t} - 1$;
                end
            end
        end
    end
    Update $\boldsymbol{\Omega}^{(t+1)}$ & $\beta_{j,k,t+1}|_{k=1}^{a_{j,t+1}=a_{j,t}}, j = 1, \cdots, Q$;
end
```

After binning each characteristic, the Kronecker product is calculated to determine all final nodes of the tree:

$$\boldsymbol{z} = \pi\left(\boldsymbol{\zeta}_{\cdot,1}\right) \otimes \pi\left(\boldsymbol{\zeta}_{\cdot,2}\right) \otimes \cdots \otimes \pi\left(\boldsymbol{\zeta}_{\cdot,Q}\right), \tag{7}$$

where Q refers to the number of weighting characteristics. $\boldsymbol{z}_i \in \mathrm{R}^{\mathbb{d}}$ in $\boldsymbol{z}$ is an approximated one-hot vector, which is the index of the leaf node where $\boldsymbol{\zeta}_i$ arrives. $\mathbb{d}$ is the number of all possible leaf nodes. The weights corresponding to all leaf nodes are a trainable vector with a size of $\mathbb{d} \times 1$. Finally, the weights of the input samples $\mathcal{V}(\boldsymbol{\zeta}; \boldsymbol{\Omega})$ are obtained by multiplying $\boldsymbol{z}$ and the trainable weight vector, where $\boldsymbol{\Omega}$ refers to the parameters of the weighting network.

As opposed to the fixed structure of DNDT, the structure of NRT is variable as it adopts both the growth and the

[4] The sequence extension is not applied for category proportion, as it is fixed for each sample during the training procedure.

pruning operations during training, which will be stated in the next two subsections. Thus, the number of activated cut points for each characteristic is dynamically determined by the model. Moreover, as the weights have no gold-standard values, the activating and deactivating of cut points for each characteristic is determined by the behavior of the activated cut points of this characteristic.

3.3.1 Growth of the Tree

The growth of the tree refers to the activation of new cut points for the weighting characteristics. Given that the values of all activated cut points vary during the training process in NRT, we believe that the model has learned the reasonable values of the existing activated cut points when their values are stable and then new cut points will be activated to achieve the growth of the tree. Let $a_{j,t}$ be the number of activated cut points of $\boldsymbol{\zeta}_{\cdot,j}$ at the t-th iteration. $s_{j,k,t}$ is the moving scope of the k-th cut point of $\boldsymbol{\zeta}_{\cdot,j}$ during the $(t-T_r)$-th iteration to the t-th iteration, which is

$$s_{j,k,t} = \max_{p=t-T_r}^{t}(\beta_{j,k,p}) - \min_{p=t-T_r}^{t}(\beta_{j,k,p}), \quad (8)$$

where T_r is the number of the record iterations. For $\zeta_{\cdot,j}$, if

$$\max_{k=1}^{a_{j,t}}(s_{j,k,t}) < \varepsilon_+, \quad (9)$$

indicating that the variation magnitudes for all activated cut points of characteristic $\zeta_{\cdot,j}$ during T_r iterations are smaller than ε_+, then $\beta_{j,a_{j,t}+1}$ will be activated, and $a_{j,t}$ increases by one. In the above formula, ε_+ is the activating threshold.

3.3.2 Pruning of the Tree

The pruning of the tree refers to the deactivation of the activated cut points for the weighting characteristics. There are two occasions that the pruning occurs. The first is that when two adjacent cut points are too close, that is,

$$d_{j,k,t} = \beta_{j,k+1,t} - \beta_{j,k,t} < \varepsilon_-, \quad (10)$$

then $\beta_{j,k,t}$ will be deactivated to alleviate overfitting, and $a_{j,t}$ decreases by one. ε_- is the deactivating threshold.

The second occasion is when no samples exist between two adjacent cut points, that is,

$$\boldsymbol{\zeta}_{\cdot,j} \notin (\beta_{j,k,t}, \beta_{j,k+1,t}). \quad (11)$$

The two cut points will then be merged to reduce the invalid nodes of the tree. Specifically, $\beta_{j,k,t}$ will be deactivated. The reason why we choose to deactivate the smaller cut point is that the subsequently activated cut points tend to be larger. As a result, the preceding activated cut points primarily accomplish the segmentation of characteristic ranges with relatively smaller values. By deactivating the smaller cut point, the newly activated cut point can more effectively partition the characteristic within its larger value range. Furthermore, our empirical findings demonstrate that deactivating $\beta_{j,k,t}$ leads to better performance compared to deactivating $\beta_{j,k+1,t}$. When $a_{j,t} = 1$, the two endpoints of $\boldsymbol{\zeta}_{\cdot,j}$ will be treated as cut points to decide whether to deactivate the remaining cut point.

As the number of cut points for each characteristic can be decreased to zero, NRT naturally has the ability to filter irrelevant or redundant characteristics. The learning algorithm of NRT is shown in Algorithm 1. Although NRT is not concerned with the order of layers in which the weighting characteristics are located during training, the information gain can be used to determine the hierarchical order of the weighting characteristics like conventional decision trees. In calculating the information gain, the weights are clustered using unsupervised clustering algorithms, such as k-means [28], and the category labels after clustering are adopted as the labels of the weights. Notably, the clustering algorithm is only used when visualizing the decision trees and does not affect the performance of the model. Fig. 3 shows an example of a regression tree obtained from our framework, in which information gain is utilized to determine the hierarchical order of weighting characteristics.

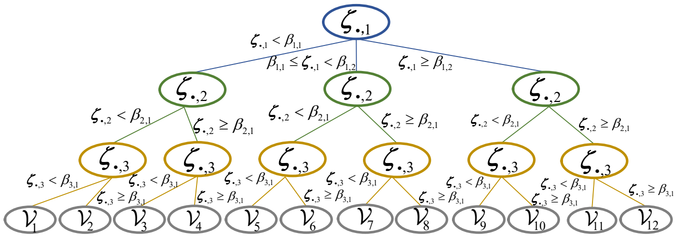

$\zeta_{\cdot,1}$
$\zeta_{\cdot,1} < \beta_{1,1}$
$\beta_{1,1} \leq \zeta_{\cdot,1} < \beta_{1,2}$
$\zeta_{\cdot,1} \geq \beta_{1,2}$
$\zeta_{\cdot,2}$
$\zeta_{\cdot,2}$
$\zeta_{\cdot,2}$
$\zeta_{\cdot,2} < \beta_{2,1}$
$\zeta_{\cdot,2} \geq \beta_{2,1}$
$\zeta_{\cdot,2} < \beta_{2,1}$
$\zeta_{\cdot,2} \geq \beta_{2,1}$
$\zeta_{\cdot,2} < \beta_{2,1}$
$\zeta_{\cdot,2} \geq \beta_{2,1}$
$\zeta_{\cdot,3}$
$\zeta_{\cdot,3}$
$\zeta_{\cdot,3}$
$\zeta_{\cdot,3}$
$\zeta_{\cdot,3}$
$\zeta_{\cdot,3}$
$\zeta_{\cdot,3} < \beta_{3,1}$
$\zeta_{\cdot,3} \geq \beta_{3,1}$
$\zeta_{\cdot,3} < \beta_{3,1}$
$\zeta_{\cdot,3} \geq \beta_{3,1}$
$\zeta_{\cdot,3} < \beta_{3,1}$
$\zeta_{\cdot,3} \geq \beta_{3,1}$
$\zeta_{\cdot,3} < \beta_{3,1}$
$\zeta_{\cdot,3} \geq \beta_{3,1}$
$\zeta_{\cdot,3} < \beta_{3,1}$
$\zeta_{\cdot,3} \geq \beta_{3,1}$
$\zeta_{\cdot,3} < \beta_{3,1}$
$\zeta_{\cdot,3} \geq \beta_{3,1}$
$\mathcal{V}_1$
$\mathcal{V}_2$
$\mathcal{V}_3$
$\mathcal{V}_4$
$\mathcal{V}_5$
$\mathcal{V}_6$
$\mathcal{V}_7$
$\mathcal{V}_8$
$\mathcal{V}_9$
$\mathcal{V}_{10}$
$\mathcal{V}_{11}$
$\mathcal{V}_{12}$

Fig. 3. An example of a regression tree obtained from our framework.

3.4 Training with Meta-Learning

The training dataset is denoted as $D^{\text{train}} = \{\boldsymbol{x}_i, \boldsymbol{y}_i\}_{i=1}^{N}$, where $\boldsymbol{x}_i$ denotes the i-th sample's feature, and $\boldsymbol{y}_i$ is the label vector of $\boldsymbol{x}_i$ over C categories. N refers to the number of samples in the training set. $f(\boldsymbol{x};\boldsymbol{\theta})$ denotes the classifier network, which is parameterized by $\boldsymbol{\theta}$. The sample weighting network is denoted as $\mathcal{V}(\boldsymbol{\zeta};\boldsymbol{\Omega})$, which is an NRT in our framework. Assume that we have a small amount of unbiased meta data $D^{\text{meta}} = \{\boldsymbol{x}_i^{\text{meta}}, \boldsymbol{y}_i^{\text{meta}}\}_{i=1}^{M}$. Our learning problem can be formulated as the following bi-level optimization problem:

$$\begin{aligned} &\min_{\boldsymbol{\Omega}} \sum_{j=1}^{M} \ell\left(\boldsymbol{y}_j^{\text{meta}}, f\left(\boldsymbol{x}_j^{\text{meta}}; \boldsymbol{\theta}^*\left(\boldsymbol{\Omega}\right)\right)\right), \\ &\text{s.t. } \boldsymbol{\theta}^*(\boldsymbol{\Omega}) = \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{N} \mathcal{V}\left(\boldsymbol{\zeta}_i; \boldsymbol{\Omega}\right) \ell\left(\boldsymbol{y}_i, f\left(\boldsymbol{x}_i; \boldsymbol{\theta}\right)\right). \end{aligned} \quad (12)$$

Notably, even if meta data are lacking, they can be compiled by meaningful samples from training data [29]. The problem in Eq. (12) is generally difficult to solve, and an online learning strategy inspired by Model-agnostic meta-learning [30] is adopted to alternatively update $\boldsymbol{\theta}$ and $\boldsymbol{\Omega}$ during training. Specifically, $\boldsymbol{\theta}$ and $\boldsymbol{\Omega}$ are updated using a single optimization loop, as shown in Fig. 2.

First, $\boldsymbol{\Omega}$ is treated as the to-be-updated parameters, and the parameters $\boldsymbol{\theta}$ of the updated classifier are formulated. SGD is used to optimize the training loss. Specifically, in each training iteration, a mini-batch of training samples $\{\boldsymbol{x}_i, \boldsymbol{y}_i\}_{i=1}^{n}$ is sampled, where n is the size of the mini-batch. Then, the updating equation of $\boldsymbol{\theta}$ can be formulated on a mini-batch of training data as follows:

$$\hat{\boldsymbol{\theta}}^{(t)}(\boldsymbol{\Omega}) = \boldsymbol{\theta}^{(t)} - \eta_1 \frac{1}{n} \sum_{i=1}^{n} \mathcal{V}\left(\boldsymbol{\zeta}_i; \boldsymbol{\Omega}\right) \nabla_{\boldsymbol{\theta}} \ell_i^{\text{train}}(\boldsymbol{\theta}) \Big|_{\boldsymbol{\theta}^{(t)}}, \quad (13)$$

Algorithm 2: Meta Training of Our Interpretable Weighting Framework

```
Input: Training data $D^{\text{train}}$, meta data $D^{\text{meta}}$, batch size $n$, meta batch size $m$, maximum iterations $T$, step sizes $\eta_1$ and $\eta_2$.
Output: Learned parameters $\boldsymbol{\theta}$ and $\boldsymbol{\Omega}$.
Initialize $\boldsymbol{\theta}^{(1)}$ and $\boldsymbol{\Omega}^{(1)}$;
for $t = 1$ to $T$ do
    Sample $\{\boldsymbol{x}_i, \boldsymbol{y}_i\}_{i=1}^{n}$ from $D^{\text{train}}$;
    Sample $\{\boldsymbol{x}_i^{\text{meta}}, \boldsymbol{y}_i^{\text{meta}}\}_{i=1}^{m}$ form $D^{\text{meta}}$;
    Formulate $\hat{\boldsymbol{\theta}}^{(t)}(\boldsymbol{\Omega})$ by Eq. (13);
    Execute steps 3 to 25 in Algorithm 1, in which $\boldsymbol{\Omega}^{(t+1)}$ is updated by Eq. (14);
    Update $\boldsymbol{\theta}^{(t+1)}$ by Eq. (15);
end
```

where η_1 is the step size, and ℓ_i^{train} refers to the loss of $\boldsymbol{x}_i$ in the mini-batch of training data.

After receiving the feedback from the classifier network, $\boldsymbol{\Omega}$ can be updated on a mini-batch of meta data as follows:

$$\boldsymbol{\Omega}^{(t+1)} = \boldsymbol{\Omega}^{(t)} - \eta_2 \frac{1}{m} \sum_{i=1}^{m} \nabla_{\boldsymbol{\Omega}} \ell_i^{\text{meta}} \left(\hat{\boldsymbol{\theta}}^{(t)}(\boldsymbol{\Omega}) \right) \Bigg|_{\boldsymbol{\Omega}^{(t)}}, \quad (14)$$

where m and η_2 are the mini-batch size of meta data and the step size, respectively. ℓ_i^{meta} refers to the loss of $\boldsymbol{x}_i$ in the mini-batch of meta data. Then, by fixing the parameters of NRT as $\boldsymbol{\Omega}^{(t+1)}$, the parameters of the classifier network are finally updated with the obtained sample weights.

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \eta_1 \frac{1}{n} \sum_{i=1}^{n} \mathcal{V}\left(\zeta_i; \boldsymbol{\Omega}^{(t+1)}\right) \nabla_{\boldsymbol{\theta}} \ell_i^{\text{train}}(\boldsymbol{\theta}) \Bigg|_{\boldsymbol{\theta}^{(t)}}. \quad (15)$$

The meta-learning algorithm of our interpretable weighting framework is displayed in Algorithm 2.

In our experiments, MLP is also adopted as the weighting network in our framework. An analysis of the results of MLP and NRT, presented in Section 4.1, indicates that using NRT as the weighting network can achieve comparable or even better performance compared with using MLP.

4 Experiments

Our experiments consist of three parts. In the first part (Section 4.1), the weights generated by our interpretable weighting framework are verified to be good. The second part (Sections 4.2, 4.3, and 4.4) reveals the obtained weighting mechanisms, including the importance of the weighting characteristics, undiscovered prior modes, and specific weighting rules obtained from NRT. The third part (Sections 4.5 and 4.6) conducts ablation studies and calculates the time complexity for our proposed interpretable weighting framework. All experimental runs are repeated five times with different seeds. Our code is available at https://github.com/AI-Mathematical/Interpretable-weighting-framework.

4.1 Evaluation for Generated Weights

Three typical learning scenarios, namely noisy label, imbalance, and large corpus, are considered.

4.1.1 Noisy Label Learning

Two settings of corrupted labels following Shu et al. [7] are adopted, namely uniform and pair-flip noise labels. CIFAR-10 and CIFAR-100 [31] are employed as they are popularly used for the evaluation of noisy labels [32], [33]. The construction of meta data also follows Shu et al. [7], in which 1,000 images with clean labels in the validation set are selected as the meta data. Wide ResNet-28-10 (WRN-28-10) [34] and ResNet-32 [35] are adopted as the classifiers for the uniform and pair-flip noise, respectively. The comparison methods include the following: Baseline, which trains the backbone network with CE loss; the robust learning methods, including SPL [4], Focal Loss [1], Co-teaching [32], Dimensionality-Driven Learning (D2L) [33], and Active Passive Loss (APL) [36], Re-weighted CE Loss (R-CE) [53], Learning Optimal sample Weights (LOW) [52], and SuperLoss [56]; and meta-learning-based methods, including MentorNet [9], Learning to Re-Weight (L2RW) [8], Gold Loss Correction (GLC) [37], Meta-weight-net [7], and Warped Probabilistic Inference (WarPI) [51]. To demonstrate the effectiveness of NRT, we also compare the performance of our framework with MLP as the weighting network. In

TABLE 1
Test accuracy (%) of ResNet-32 on CIFAR-10 and CIFAR-100 with varying noise rates under flip noise. The best and the second best results are highlighted in bold and underlined. Mean accuracy (±std) over 5 repetitions is reported.

Dataset	Noisy CIFAR-10			Noisy CIFAR-100		
Noise rate	0	0.2	0.4	0	0.2	0.4
Baseline	92.89±0.32	76.83±2.30	70.77±2.31	70.50±0.12	50.86±0.27	43.01±1.16
SPL [4]	88.52±0.21	87.03±0.34	81.63±0.52	67.55±0.27	63.63±0.30	53.51±0.53
Focal Loss [1]	93.03±0.16	86.45±0.19	80.45±0.97	70.02±0.53	61.87±0.30	54.13±0.40
Co-teaching [32]	89.87±0.10	82.83±0.85	75.41±0.21	63.31±0.05	54.13±0.55	44.85±0.81
D2L [33]	92.02±0.14	87.66±0.40	83.89±0.46	68.11±0.26	63.48±0.53	51.83±0.33
APL [36]	92.36±0.21	87.23±0.27	80.08±0.12	69.13±0.11	59.37±0.43	52.98±0.70
MentorNet [9]	92.13±0.30	86.36±0.31	81.76±0.28	70.24±0.21	61.97±0.47	52.66±0.56
L2RW [8]	89.25±0.37	87.86±0.36	85.66±0.51	64.11±1.09	57.47±1.16	50.98±1.55
GLC [37]	91.02±0.20	89.58±0.33	88.92±0.24	65.42±0.23	63.07±0.53	62.22±0.62
Meta-weight-net [7]	92.07±0.15	90.33±0.61	87.54±0.23	70.11±0.33	64.22±0.28	58.64±0.47
R-CE [53]	90.44±0.25	88.25±0.52	83.86±0.34	65.78±0.27	63.48±0.46	54.65±0.33
LOW [52]	92.13±0.37	72.77±0.71	68.35±0.35	70.09±0.42	50.45±0.47	42.19±0.28
SuperLoss [56]	92.37±0.35	89.19±0.71	84.23±0.46	70.12±0.53	63.35±0.28	54.87±0.66
WarPI [51]	92.23±0.67	90.93±0.34	89.87±0.29	70.15±0.26	65.52±0.31	62.37±0.43
Ours (MLP)	93.56±0.32	91.47±0.24	90.36±0.23	**71.64±0.35**	66.36±0.33	62.73±0.20
Ours (NRT w/o pruning)	$\underline{93.79\pm0.25}$	**92.14±0.18**	$\underline{90.64\pm0.20}$	$\underline{71.38\pm0.21}$	$\underline{66.78\pm0.17}$	$\underline{63.14\pm0.34}$
Ours (NRT)	**93.91±0.23**	$\underline{91.91\pm0.19}$	**90.88±0.14**	71.34±0.25	**67.43±0.26**	**63.49±0.27**

TABLE 2
Test accuracy (%) of WRN-28-10 on CIFAR-10 and CIFAR-100 with varying noise rates under uniform noise.

Dataset	Noisy CIFAR-10			Noisy CIFAR-100		
Noise rate	0	0.4	0.6	0	0.4	0.6
Baseline	95.60±0.22	68.07±1.23	53.12±3.03	79.95±1.26	51.11±0.42	30.92±0.33
SPL [4]	90.81±0.34	86.41±0.29	53.10±1.78	59.79±0.46	46.31±2.45	19.08±0.57
Focal Loss [1]	95.70±0.15	75.96±1.31	51.87±1.19	81.04±0.24	51.19±0.46	27.70±3.77
Co-teaching [32]	88.67±0.25	74.81±0.34	73.06±0.25	61.80±0.25	46.20±0.15	35.67±1.25
D2L [33]	94.64±0.33	85.60±0.13	68.02±0.41	66.17±1.42	52.10±0.97	41.11±0.30
APL [36]	94.12±0.23	86.49±0.14	79.22±0.67	77.25±0.41	57.84±0.34	49.13±0.26
MentorNet [9]	94.35±0.42	87.33±0.22	82.80±1.35	73.26±1.23	61.39±3.99	36.87±1.47
L2RW [8]	92.38±0.10	86.92±0.19	82.24±0.36	72.99±0.58	60.79±0.91	48.15±0.34
GLC [37]	94.30±0.19	88.28±0.03	83.49±0.24	73.75±0.51	61.31±0.22	50.81±1.00
Meta-weight-net [7]	94.52±0.25	89.27±0.28	84.07±0.33	78.76±0.24	67.73±0.26	58.75±0.11
R-CE [53]	91.04±0.61	85.74±0.24	75.56±0.29	72.87±0.71	56.23±0.75	41.05±0.61
LOW [52]	94.24±0.27	67.25±0.48	51.46±0.53	74.87±0.25	50.25±0.66	36.32±0.37
SuperLoss [56]	94.55±0.55	86.43±0.17	79.42±0.23	78.89±0.44	55.64±0.71	41.34±0.46
WarPI [51]	94.67±0.44	89.73±0.61	84.44±0.69	78.68±0.52	67.90±0.43	59.04±0.19
Ours (MLP)	**96.49±0.20**	90.68±0.18	85.25±0.28	81.45±0.24	69.02±0.13	60.87±0.23
Ours (NRT w/o pruning)	96.31±0.25	90.65±0.19	85.73±0.24	81.72±0.18	69.56±0.21	**61.51±0.18**
Ours (NRT)	96.33±0.19	**91.13±0.22**	**86.06±0.41**	**81.95±0.23**	**69.99±0.11**	61.32±0.19

TABLE 3
Test accuracy (%) of ResNet-32 on long-tailed CIFAR-10 and CIFAR-100.

Dataset	Long-tailed CIFAR-10					Long-tailed CIFAR-100				
Imbalance factor	10	20	50	100	200	10	20	50	100	200
Baseline	86.39	82.23	74.81	70.36	65.68	55.71	51.14	43.85	38.32	34.84
Focal Loss [1]	86.66	82.76	76.71	70.38	65.29	55.78	51.95	44.32	38.41	35.62
Class-Balanced [5]	87.49	84.36	79.27	74.57	68.89	57.99	52.59	45.32	39.60	36.23
Class-Balanced Fine-tuning [38]	83.17	83.22	77.44	71.34	66.24	57.57	52.30	46.22	41.50	38.66
Class-Balanced Focal [1]	87.48	83.78	79.22	74.57	68.15	57.89	52.69	45.21	39.71	36.25
LDAM [39]	87.32	73.89	78.83	73.55	66.75	57.29	51.59	46.16	40.60	38.45
L2RW [8]	85.19	82.12	78.93	74.16	66.51	53.73	51.64	44.44	40.23	33.38
Meta-weight-net [7]	87.84	84.94	80.06	75.21	68.91	58.46	54.37	46.74	42.09	37.91
R-CE [53]	86.55	82.23	75.66	70.13	63.98	55.35	49.88	43.54	38.05	34.21
LOW [52]	87.55	83.46	76.42	72.78	67.23	55.42	51.93	44.64	38.88	35.21
SuperLoss [56]	85.98	82.45	75.23	70.62	64.75	55.28	50.99	42.74	38.46	34.56
Probe-and-allocate [50]	85.12	82.65	78.71	73.52	65.91	55.37	51.89	43.72	40.76	37.17
Ours (MLP)	89.11	86.37	81.91	77.33	70.99	59.35	55.12	48.06	42.97	39.25
Ours (NRT w/o pruning)	89.02	86.52	81.35	76.95	71.93	59.71	55.43	47.89	43.17	39.44
Ours (NRT)	**89.53**	**86.87**	**82.83**	**77.71**	**72.06**	**60.14**	**55.71**	**48.54**	**43.58**	**39.71**

addition, we compare the performance of NRT with and without pruning. The training and testing configurations used in Meta-weight-net [7] are followed. Regarding the parameters in NRT, ε_+ and T_r are fixed as 0.1 and 20 in all scenarios. ε_- and c are searched in $\{0.1, 0.2, 0.3\}$ and $\{2, 3, 4\}$, respectively. The temperature τ is fixed as 0.1. The comparative results are shown in Tables 1 and 2. Considering our configuration is consistent with Shu et al. [7], the results of the competing methods reported in the Meta-weight-net paper are directly presented (some are from their original papers).

Our interpretable weighting framework with either MLP or NRT as the weighting network achieves the best performance among the compared methods under all noisy settings. The superiority of our method over Meta-weight-net [7] indicates its effectiveness in weighting the samples when considering multiple weighting characteristics. In addition, taking NRT as the weighting network achieves comparable or even better performance than MLP. Thus, although our framework is mainly designed for interpretability, it also achieves commendable performance outcomes.

4.1.2 Imbalance Learning

The long-tailed CIFAR dataset compiled by Cui et al. [5] is employed. ResNet-32 [35] is utilized as the backbone network. Several robust weighting methods including meta-learning-based methods are compared: Focal Loss [1], Class-Balanced Loss [5], Label-Distribution-Aware Margin (LDAM) Loss [39], LDAM-Standard Re-Weighting (LDAM-DRW) [39], L2RW [8], R-CE [53], LOW [52], SuperLoss [56], Probe and allocate [50], and Meta-weight-net [7]. The experimental configurations including the settings of NRT are the same as those in the previous subsection. Ten images per category in the validation set are selected as the meta data [7]. The results are shown in Table 3. Our framework still obtains the best performance among the compared weighting methods. In addition, NRT achieves comparable or even better performance than MLP.

4.1.3 Learning for Large Corpus

For fair comparisons, we adopt ResNet-50 [35] as the backbone network for iNaturalist 2018 (iNat 2018) [40] and pretrain it on the ImageNet [41] and iNat 2017 [42] datasets. The experimental configuration and compared methods follow Li et al. [60]. The hyperparameter settings for NRT are the same as those in the previous subsection. Table 4 presents the experimental results on iNat 2018. Our interpretable weighting framework yields the best performance among the competing methods, verifying that our framework can also achieve valuable weights in this scenario.

4.2 Importance of Weighting Characteristics

The interpretability of our framework helps us to analyze the importance of each weighting characteristic during the

TABLE 4
Test top-1 and Top-5 accuracy (%) on iNaturalist (iNat) 2018.

Dataset	iNat 2018	
Method	Top-1	Top-5
CE	65.76	84.15
Class-Balanced [5]	66.43	84.17
Class-Balanced Focal [1]	61.12	81.03
BBN [44]	66.29	-
LDAM [39]	64.58	83.52
LDAM-DRW [39]	68.00	85.18
Meta-weight-net [7]	67.95	85.32
Meta-class-weight [45]	67.55	86.17
MetaSAug [60]	68.75	-
Ours (MLP)	70.15	86.94
Ours (NRT)	**70.33**	**87.45**

training process. Following the evaluation approach used for conventional decision trees, the information gain is adopted to measure the importance of the nine weighting characteristics every five epochs. The results are shown in Fig. 4. Under the noisy and standard scenarios, the characteristic of category proportion ($\boldsymbol{\zeta}_{\cdot,5}$) is filtered out as all categories have equal proportions. Three weighting characteristics, including loss ($\boldsymbol{\zeta}_{\cdot,1}$), margin ($\boldsymbol{\zeta}_{\cdot,2}$), and gradient norm ($\boldsymbol{\zeta}_{\cdot,3}$), are more important than others on noisy data, as shown in Fig. 4(a). Besides, the weights in the previous rounds ($\boldsymbol{\zeta}_{\cdot,7}$ and $\boldsymbol{\zeta}_{\cdot,9}$) are valid. For imbalanced data, margin ($\boldsymbol{\zeta}_{\cdot,2}$), gradient norm ($\boldsymbol{\zeta}_{\cdot,3}$), and information entropy ($\boldsymbol{\zeta}_{\cdot,4}$) are more effective, as shown in Fig. 4(b). Similar to the finding regarding noisy data, the role of weights in the previous rounds ($\boldsymbol{\zeta}_{\cdot,7}$ and $\boldsymbol{\zeta}_{\cdot,8}$) is also apparent, indicating that the variation of weight is generally not mutated. On standard data, margin ($\boldsymbol{\zeta}_{\cdot,2}$), gradient norm ($\boldsymbol{\zeta}_{\cdot,3}$), and information entropy ($\boldsymbol{\zeta}_{\cdot,4}$) are of greater importance than others, as shown in Fig. 4(c).

Four of the weighting characteristics (i.e., loss, margin, gradient norm, and information entropy) are commonly utilized to measure the learning difficulty of samples. We analyze their effectiveness in noisy and imbalanced scenarios. Three of them (i.e., loss, margin, and gradient norm) have significant differences between noisy and clean samples, as shown in Fig. 5. Noisy samples have a larger average loss, a smaller average margin, and a larger average gradient norm, indicating that noisy samples are harder to learn. Consequently, the three characteristics can be adopted as the difficulty measures on noisy data. As for imbalanced data, all four measures have distinct differences between samples in the head and tail categories, as shown in Fig. 6. The samples in the tail categories have a larger average loss, a smaller average margin, a larger average gradient norm, and a larger average information entropy, indicating that samples in the tail categories are harder to learn on average.

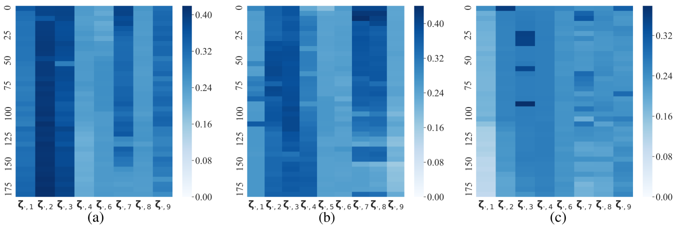


Fig. 4. (a): The importance of the eight characteristics on CIFAR-10 with 40% flip noise; (b): The importance of the nine characteristics on CIFAR-10 with an imbalance factor of 100; (c): The importance of the eight characteristics on standard CIFAR-10.

4.3 Undiscovered Prior Modes of the Learned Weights

Existing studies generally assume that the prior mode of sample weights is either easy-first (e.g., SPL) or hard-first (e.g., Focal Loss) based on particular difficulty measures. Our framework adopts a data-driven path and does not assume any prior mode. This subsection explores the prior modes contained in the weights learned by our framework.

4.3.1 Single Prior Modes

A single prior mode means that all samples in the training set take the same prior pattern. Almost all existing weighting strategies adopt this single way. For example, SPL and CL are under the easy-first mode throughout training, while Focal Loss and Class-Balanced Loss belong to the hard-first mode. Our experiments reveal suitable scenarios for these two priority modes. Spearman correlation is utilized to measure the correlation between the weights and the difficulty measures. These correlations indicate that a good set of weights on the noisy (imbalanced) data adopts the easy (hard)-first mode regardless of the employed difficulty measure. For example, the average Spearman correlation between the loss/margin/gradient norm and the weight on the entire noisy data is $-0.35/0.46/-0.45$, indicating a moderate negative/positive/negative correlation. Figs. 7(a) to (c) show the correlations between the weights and the three difficulty measures on the entire noisy data of the optimal model, in which 300 pieces of data are randomly sampled from the training data. These schemes all belong to the easy-first mode, as samples with small losses, large margins, or small gradient norms are easy to learn in general.

The learned weights of different types of samples under the noisy and imbalanced scenarios are further analyzed. Noisy samples have lower average weights than clean ones, as shown in Figs. 8(a) and (b). Thus, a good set of weights on noisy data should have larger values on clean samples, making the model less disturbed by noise. In the imbalanced classification, the average weights of the last five categories are higher than those of the first five categories, as shown in Figs. 8(c) and (d), indicating that a good set of weights on imbalanced data should have larger values on the samples in the tail categories. Therefore, we can obtain that the easy-first mode is more suitable for noisy data and the hard-first mode performs better on imbalanced data because noisy samples and samples in tail categories are generally hard ones, as stated in Section 4.2.

4.3.2 Hybrid Prior Modes

The hybrid prior mode refers to both the easy-first and the hard-first modes appearing in partial samples in the training set, while each existing weighting method adopts the same prior mode for all samples. This novel prior mode is observed from the weights learned by our framework. Although the prior mode of the entire noisy data is easy-first, we find that the prior mode of individual clean (noisy) samples is hard (easy)-first regardless of the employed difficulty measure (i.e., loss, margin, and gradient norm). Thus, different prior modes are adopted by clean and noisy samples. Alternatively, the hybrid prior mode is applied to noisy data. Figs. 7(d) and (e) show the prior modes of individual clean and noisy samples when the loss is used

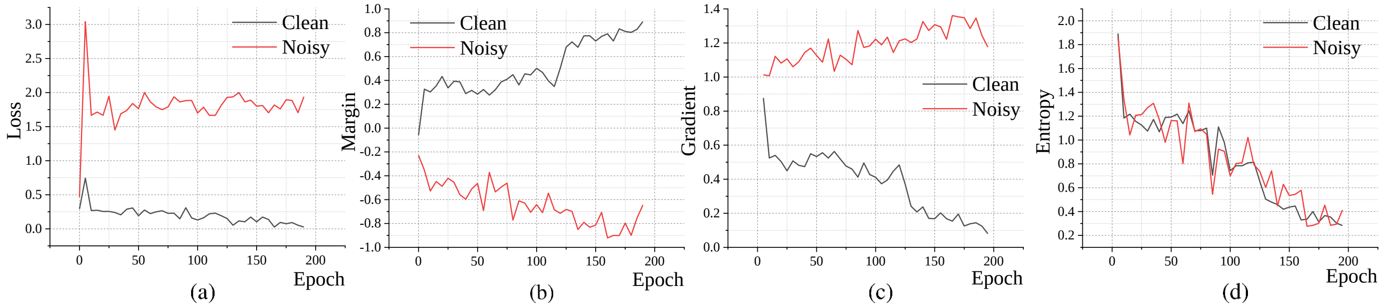


Fig. 5. The average loss, margin, gradient norm, and information entropy of clean and noisy samples on CIFAR-10 with 40% uniform noise.

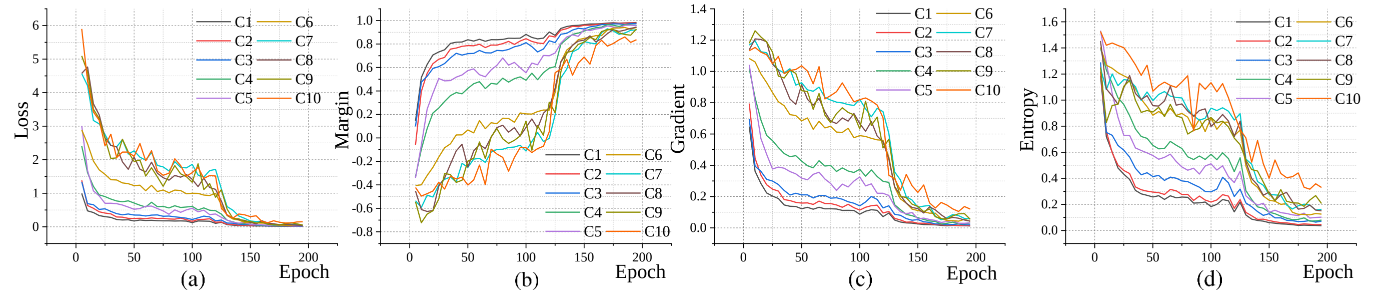


Fig. 6. The average loss, margin, gradient norm, and information entropy of samples in ten categories on CIFAR-10 with an imbalance factor of 200. C1 to C10 are from the first head category to the last tail category.

as the difficulty measure. Loss and weight are positively correlated on the clean samples with a correlation coefficient of 0.33 and they are negatively correlated on the noisy samples with a correlation coefficient of -0.61.

4.3.3 Shifting Prior Modes

The novel shifting prior mode can also be observed from the learned weights. Under this paradigm, the prior mode of the entire dataset transforms from easy (hard)-first to hard (easy)-first during the training process, whereas the prior modes in the previous studies were maintained as easy-first or hard-first modes during the entire training procedure. In our experiments involving imbalanced classification, the prior mode is easy-first in the early training stage, and it shifts to and maintains hard-first in the later training stage, as shown in Fig. 9. The loss and the weight are negatively correlated in the early training stage, that is, the larger the loss, the smaller the weight, belonging to the easy-first mode. In the later stage, they are positively correlated, which belongs to the hard-first mode. The same findings are also obtained under the other three difficulty measures (i.e., margin, gradient norm, and information entropy). In addition, the shifting prior mode does not appear on the noisy datasets where the prior mode is always easy-first during the entire training procedure.

We provide an explanation for this phenomenon. Given that the weighting network is optimized on the meta data, the small unbiased meta dataset plays a crucial role in guiding the weighting network to assign appropriate weights to samples. Additionally, for deep learning models, there exists a tendency to prioritize the learning of easier samples initially, gradually progressing towards more challenging samples as the training progresses [49]. Consequently, during the early training stage, the weighting network assigns higher weights to easy samples. Thus, the easy-first mode is adopted in both imbalanced and noisy-label learning scenarios. Once the easy samples have been effectively learned, the balanced meta data will guide the weighting network to assign high weights on samples belonging to tail categories in imbalanced learning scenarios, which are generally hard ones. This facilitates the model to perform well on balanced datasets. Thus, the prior mode of hard-first is adopted in the medium and later training stages for imbalanced learning. In the case of noisy data, the clean meta data will always guide the weighting network to assign high weights to clean samples, which are typically easier than noisy samples, enabling the model to perform well on clean datasets. Therefore, the easy-first mode is maintained on noisy datasets.

4.4 Weighting Rules Derived from the Trained NRT

A large number of weighting rules can be obtained from the trained NRT, which reflects how the combination of weight-

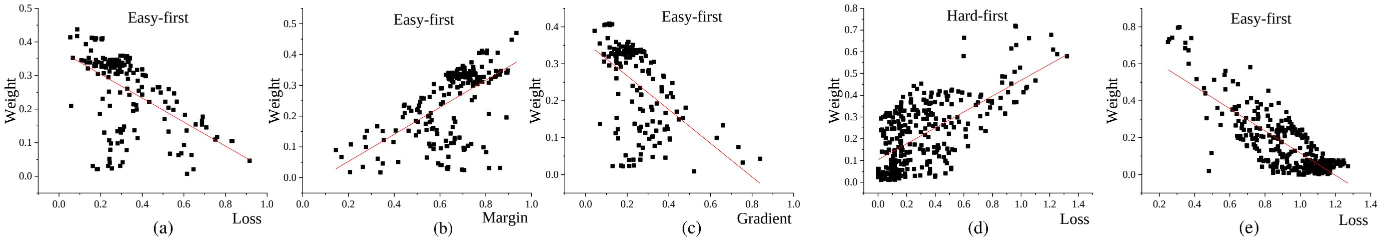


Fig. 7. (a)-(c): Prior modes on CIFAR-10 with 20% flip noise when loss, margin, and gradient norm are adopted as the difficulty measure; (d) and (e): Prior modes of individual clean (d) and noisy (e) samples on CIFAR-10 with 20% flip noise when the loss is adopted as the difficulty measure.

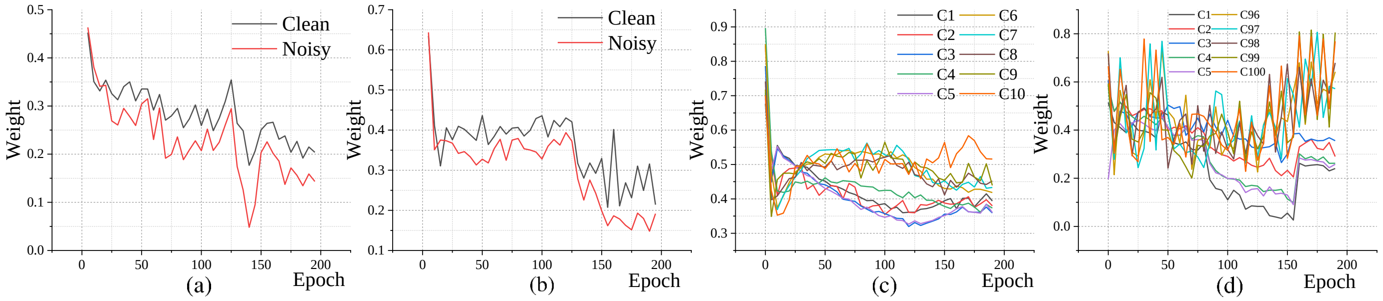


Fig. 8. (a) and (b): Weights of noisy and clean samples on CIFAR-10 (a) and CIFAR-100 (b) with 20% flip noise; (c) and (d): Weights of samples in the first and last five categories on CIFAR-10 (c) and CIFAR-100 (d) with an imbalance factor of 100.

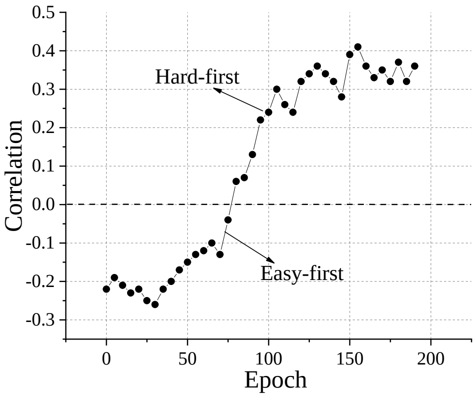


Fig. 9. Variation of the correlation coefficient between loss and weight during training on CIFAR-10 with an imbalance factor of 100.

TABLE 5
Accuracy (%) on imbalanced (with imbalance factors of 20 and 100) and noisy (with 20% and 40% flip noise) CIFAR-10 with different values of ε_- and c.

ε_-	20	100	c	20%	40%
0.1	86.82	77.18	1	91.28	89.63
0.2	86.87	77.62	2	91.76	90.36
0.3	86.41	77.71	3	92.14	90.67
0.4	85.93	76.97	4	91.87	90.88
0.5	85.30	76.03	5	90.66	89.25

ing characteristics determines the sample weights. As stated in Section 3.3, NRT is not concerned with the order of layers in which the weighting characteristics are located during the training process. As a means of visualizing the decision trees, the hierarchical order of the weighting characteristics is constructed using the information gain similar to that of conventional decision trees. Specifically, the weights are first clustered into three categories (i.e., low, medium, and high) using k-means to calculate the information gain. Then, the weighting characteristic with the highest information gain is placed in the first layer.

Due to space limitations, we only display the first four layers of the decision trees obtained at the last iteration of the four training epochs on noisy CIFAR-10, as shown in Fig. 10. The importance of weighting rules in decision trees is measured based on the usage frequency of the rules, which are shown below the leaf nodes. The most frequently used weighting rules corresponding to the three levels of weights are $\boldsymbol{\zeta}_{\cdot,7} \geq 0.50 \,\&\, \boldsymbol{\zeta}_{\cdot,3} < 0.12 \,\&\, \boldsymbol{\zeta}_{\cdot,2} \geq 0.87 \rightarrow$ High, $\boldsymbol{\zeta}_{\cdot,2} \geq 0.75 \,\&\, \boldsymbol{\zeta}_{\cdot,3} < 0.38 \,\&_{\cdot,7} < 0.50 \rightarrow$ Medium, and $0.31 \leq \boldsymbol{\zeta}_{\cdot,2} < 0.71 \,\&\, \boldsymbol{\zeta}_{\cdot,3} < 0.16 \,\&\, \boldsymbol{\zeta}_{\cdot,1} < 0.02 \rightarrow$ Low. By analyzing the obtained weighting rules, the following findings can be obtained:

- Most samples have relatively high weights in the early training stage and relatively low weights in the later training stage. As shown in Fig. 11(a), the weighting rules at the 10-th epoch all correspond to high or medium-level weights, whereas those at the 180-th epoch all correspond to low-level weights.
- As these rules are from noisy scenarios, they mainly follow the easy-first mode. Specifically, a large margin ($\boldsymbol{\zeta}_{\cdot,2}$), a small gradient norm ($\boldsymbol{\zeta}_{\cdot,3}$), or their combination is associated with relatively high weights.
- The weight in the current epoch is positively relevant to that in the previous epoch ($\boldsymbol{\zeta}_{\cdot,7}$) of a sample.

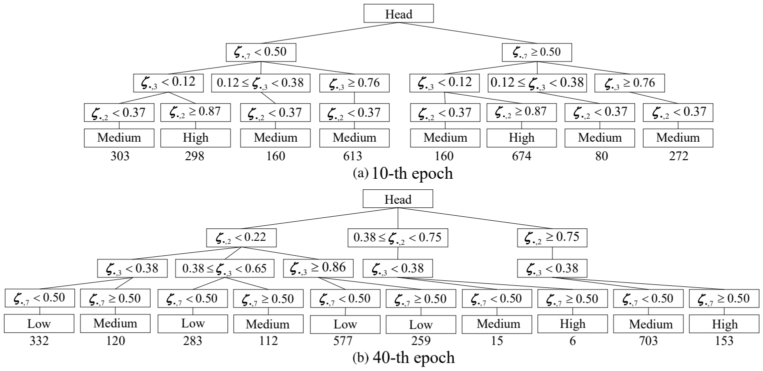
4.5 Ablation Studies

As for the hyperparameters in NRT, the maximum number of the cut points c and the deactivating threshold ε_- are crucial. The larger the value of c is, the wider the tree is. The larger the value of ε_- is, the higher the degree of pruning is. We investigate how the performance of the proposed interpretable weighting framework changes with different c and ε_- values. The results are shown in Table 5. The performance is relatively stable when ε_- locates in $[0.1, 0.3]$. A certain degree of pruning will improve the performance but excessive pruning will inhibit it. In addition, c is suitable to be selected in $\{2, 3, 4\}$. If its value is quite large, the model is prone to overfitting, and the CUDA memory will increase. If its value is quite small, the modeling ability of NRT will weaken. In our experiments, all of the best results appear in the case in which c belongs to $\{2, 3, 4\}$.

Moreover, ablation studies for the step sizes of the classifier η_1 and the weighting network η_2 are undertaken. The outcomes are depicted in Fig. 11. Based on the findings, the model attains optimal performance when the step sizes for the classifier and the weighting network are set to 0.1 and 0.001, respectively.

4.6 Time Complexity

Our algorithm does not increase the training and inference time complexity compared with other meta-learning methods, such as Meta-weight-net [7]. Moreover, by adjusting the update frequency of the weighting network, the training efficiency can be further improved on the premise of ensuring performance. In other words, the weighting network (i.e., MLP or NRT) does not need to be updated in each iteration in many cases. We record the time cost on a Linux platform



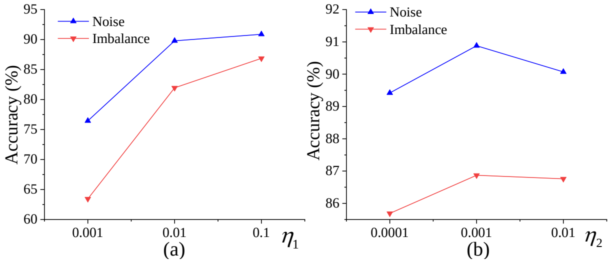
(a) 10-th epoch

(d) 180-th epoch

(b) 40-th epoch

(c) 100-th epoch

Fig. 10. Four decision trees obtained at the 10-th, 40-th, 100-th, and 180-th epochs on CIFAR-10 with 20% flip noise.



(a)

(b)

Fig. 11. Ablation studies for the step sizes η_1 (a) and η_2 (b) in both imbalanced (CIFAR-10 with an imbalance factor of 20) and noisy (CIFAR-10 with 40% flip noise) learning scenarios.

with a 24Gb RTX 3090 GPU. When the weighting network is updated in every iteration, the training of the Meta-weight-net model consumes 14,820 seconds for 200 epochs on standard CIFAR-10 data, while the training of our interpretable weighting framework consumes 15,006 seconds. In the case of updating the weighting network every 20 iterations, the training time for our framework is 2,560 seconds, which is an improvement of 82.94% compared to updating the weighting network in each iteration.

5 Discussions

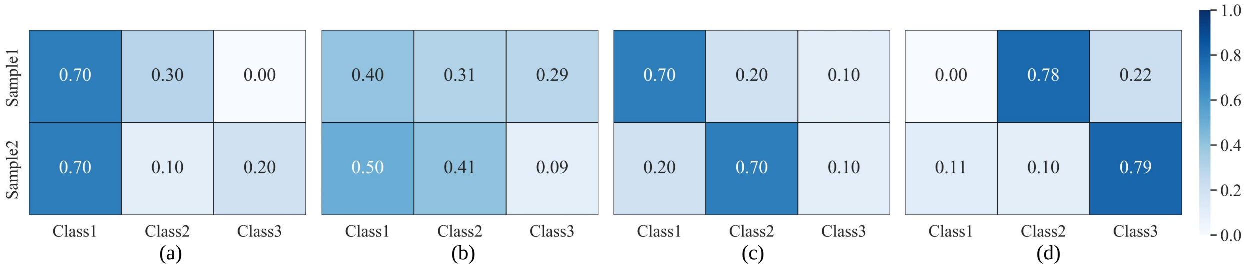
Our interpretable weighting framework well investigates weighting mechanisms in a purely data-driven manner, as well as solving the four questions presented in Section 1.

(i) The first question refers to the effectiveness of each weighting characteristic. Based on our investigation, two weighting characteristics, namely margin and the norm of loss gradient, are generally more important in three typical learning scenarios. Besides, the loss characteristic has relatively high importance in the noisy scenario. Information entropy is crucial in imbalanced and standard scenarios. These weighting characteristics should be given priority in the design of future weighting schemes.

(ii) Our exploring indicates that a single weighting characteristic can not sufficiently determine the importance of a training sample as shown in Fig. 12. Multiple weighting characteristics are involved in our trained NRT. The weighting rules obtained from our framework can inspire how the weighting characteristics should be combined. For example, as shown in Fig. 10, the margin and the gradient norm characteristics are important in the noisy scenario. Moreover, the model tends to assign higher weights to samples with large margins and small gradient norms. Therefore, a weighting function that monotonically increases with respect to the margin and monotonically decreases with respect to the gradient norm can be constructed in the noisy scenario.

(iii) The third question refers to the prior mode. Through analyzing the prior modes for the weights generated by our interpretable weighting framework, a reliable conclusion can be driven: both the easy-first and the hard-first prior modes are reasonable. Nevertheless, their applied scenarios are different. The easy-first mode is more suitable for noisy scenes, and the hard-first one is more suitable for the imbalanced scenario. Our conclusion is supported by previous studies. For example, Wu et al. [17] empirically found that the typical easy-first weighting strategy CL mainly works on noisy data.

(iv) Besides the easy-first and the hard-first modes, two more delicate prior modes, namely the hybrid and



(a)

(b)

(c)

(d)

Fig. 12. Four illustrative cases where a single weighting characteristic cannot well indicate the importance of a training sample. There are a total number of 3 classes and Class1 is the ground truth class of the two samples. (a): The two samples have identical losses yet different other characteristics (i.e., margin, gradient norm, and information entropy); (b): The two samples have identical margins yet different other characteristics; (c): The two samples have identical information entropy yet different other characteristics; (d): The two samples have similar gradient norms yet different other characteristics.

the shifting modes, are observed. Alternatively, the prior mode is not necessary to be fixed and single in training, which answers the fourth question.

It is worth noting that the space complexity of NRT is relatively high when numerous characteristics are utilized due to the use of the Kronecker product. Nevertheless, this issue can be avoided by training a forest with random subspace [46] on the basis of sacrificing certain interpretability.

Our framework also demonstrates its versatility in explaining other training components. We present two examples here. Firstly, our framework can help investigate the mechanisms of logit adjustment, a widely utilized technique that has shown success in various learning scenarios, such as long-tail learning [57], [58] and implicit semantic data augmentation [59], [60]. As the adjustment terms for the logits of samples are also determined by the characteristics of samples, the weighting network (i.e., NRT) in the framework can be regarded as an adjustment network and the adjustment terms can be generated. Thus, the mechanisms of logit adjustment can be investigated through our interpretable framework. Notably, if the logits for all classes are perturbed, similar to the manner in LA [57], the dimension of the output layer in NRT needs to be modified. Secondly, our interpretable weighting framework has the potential to provide insights into the mechanisms that determine the weights assigned to different teacher networks in knowledge distillation tasks [48]. However, the input of NRT is supposed to be modified. Specifically, instead of relying on the characteristics of samples, the characteristics of each teacher network should be incorporated. Therefore, it is worth noting that when our interpretable framework is applied in other scenarios, the input or the dimension of the output layer in NRT may need to be modified. Besides, our interpretable framework has a limitation in that NRT is primarily well-suited for tabular data. Thus, our framework mainly well interprets the components that are determined by tabular data.

6 Conclusions

In this study, a data-driven approach is adopted to investigate the weighting mechanism of a good set of sample weights. To this end, an interpretable weighting framework is constructed to infer the weights for training samples. It contains a backbone classifier network and an elaborately designed NRT as the weighting network. The comparison of benchmark datasets indicates that the proposed interpretable weighting framework can achieve quite competitive performance. A package of weighting mechanisms including the important sample characteristics, prior modes, and combination rules is further revealed from the NRT learned by our framework. These findings give reasonable answers to the questions related to the weighting mechanism issue. Weighting strategies with novel prior modes, such as hybrid prior modes and shifting prior modes, based on more effective weighting characteristics may be considered as a future research direction.

References

[1] T. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal loss for dense object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pp. 2999–3007, 2017.

[2] Y. Freund, and R. E. Schapire, "Experiments with a new boosting algorithm," in *Proc. Int. Conf. Mach. Learn.*, pp. 148–156, 1996.

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

[4] M. Pawan Kumar, B. Packer, and D. Koller, "Self-paced learning for latent variable models," in *Proc. 24th Int. Conf. Neural Inf. Process. Syst.*, pp. 1189–1197, 2010.

[5] Y. Cui, M. Jia, T. Y. Lin, Y. Song, and S. Belongie, "Class-balanced loss based on effective number of samples," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pp. 9260–9269, 2019.

[6] K. R. M. Fernando and C. P. Tsokos, "Dynamically weighted balanced loss: class imbalanced learning and confidence calibration of deep neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 99, pp. 2162–2388, 2021.

[7] J. Shu, Q. Xie, L. Yi, Q. Zhao, S. Zhou, Z. Xu, and D. Meng, "Meta-weight-net: Learning an explicit mapping for sample weighting," in *Proc. 33rd Int. Conf. Neural Inf. Process. Syst.*, pp. 1919–1930, 2019.

[8] M. Ren, W. Zeng, B. Yang, and R. Urtasun, "Learning to reweight examples for robust deep learning," in *Proc. Int. Conf. Mach. Learn.*, pp. 6900–6909, 2018.

[9] L. Jiang, Z. Zhou, T. Leung, L. Li, and F. Li, "Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels," in *Proc. Int. Conf. Mach. Learn.*, pp. 3601–3620, 2018.

[10] J. Byrd, and Z. C. Lipton, "What is the effect of importance weighting in deep learning?," in *Proc. Int. Conf. Mach. Learn.*, pp. 1405–1419, 2019.

[11] D. Xu, Y. Ye, and C. Ruan, "Understanding the role of importance weighting for deep learning," in *Proc. 9th Int. Conf. Learn. Representations*, 2020.

[12] X. Wang, Y. Chen, and W. Zhu, "A survey on curriculum learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 1, no. 1, pp. 4555–4576, 2021.

[13] B. Li, Y. Liu, and X. Wang, "Gradient harmonized single-stage detector," in *Proc. 33rd AAAI Conf. Artif. Intell.*, pp. 8577–8584, 2019.

[14] B. Frénay, and M. Verleysen, "Classification in the presence of label noise: a survey," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 5, pp. 845–869, 2013.

[15] D. Chen, Y. Lin, W. Li, P. Li, J. Zhou, and X. Sun, "Measuring and relieving the over-smoothing problem for graph neural networks from the topological view," in *Proc. 34th AAAI Conf. Artif. Intell.*, pp. 3438–3445, 2020.

[16] J. Zhang, J. Zhu, G. Niu, B. Han, M. Sugiyama, and M. Kankanhalli, "Geometry-aware instance-reweighted adversarial training," in *Proc. 10th Int. Conf. Learn. Representations*, 2021.

[17] X. Wu, E. Dyer, and B. Neyshabur, "When do curricula work?," in *Proc. 10th Int. Conf. Learn. Representations*, 2021.

[18] C. Santiagoa, C. Barataa, M. Sasdellib, G. Carneirob, and J. C.Nasciment, "LOW: Training deep neural networks by learning optimal sample weights," *Pattern Recognition*, vol. 110, pp. 107585, 2021.

[19] E. Aguilar, B. Nagarajan, R. Khatun, M. Bolaños, and P. Radeva, "Uncertainty modeling and deep learning applied to food image analysis," in *BIOSTEC*, pp. 3–16, 2020.

[20] K. Yang, Z. Yu, C.-L.-P. Chen, W. Cao, J. J. You and H. S. Wong, "Incremental weighted ensemble broad learning system for imbalanced data," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 12, pp. 5809–5824, 2021.

[21] S. Li, W. Ma, J. Zhang, C. H. Liu, J. Liang and G. Wang, "Meta-reweighted regularization for unsupervised domain adaptation," *IEEE Trans. Knowl. Data Eng.*, 2021.

[22] D. Soudry, E. Hoffer, M. S. Nacson, S. Gunasekar, and N. Srebro, "The implicit bias of gradient descent on separable data," *J. Mach. Learn. Res.*, vol. 19, no. 1, pp. 2822–2878, 2018.

[23] Y. Sasaki, and Y. Okajima, "Alternative ruleset discovery to support black-box model predictions," in *Proc. IEEE 21st Int. Conf. Data Mining*, pp. 1312-1317, 2021.

[24] N. Frosst, and G. Hinton, "Distilling a neural network into a soft decision tree," 2017, *arXiv:1711.09784*.

[25] A. Wan, L. Dunlap, D. Ho, J. Yin, S. Lee, H. Jin, S. Petryk, et al., "NBDT: Neural-backed decision trees," in *Proc. 10th Int. Conf. Learn. Representations*, 2021.

[26] Y. Yang, I. G. Morillo, and T. M. Hospedales, "Deep neural decision trees," 2018, *arXiv:1806.06988*.

[27] Q. A. Wang, "Probability distribution and entropy as a measure of uncertainty," *J. Phys. A*, vol. 41, no. 6, pp. 065004, 2008.

[28] J. A. Hartigan, and M. A. Wong, "Algorithm AS 136: A k-means clustering algorithm," *JRSSB*, vol. 28, no. 1, pp. 100–108, 1979.

[29] Z. Zhang, and T. Pfister, “Learning fast sample re-weighting without reward data,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pp. 725–734, 2021.

[30] C. Finn, P. Abbeel, and S. Levine, “Model-agnostic meta-learning for fast adaptation of deep networks,” in *Proc. 34th Int. Conf. Mach. Learn.*, pp. 1126–1135, 2017.

[31] A. Krizhevsky, and G. Hinton, “Learning multiple layers of features from tiny images,” *Technical Report*, pp. 1–60, 2009.

[32] B. Han, Q. Yao, X. Yu, G. Niu, M. Xu, W. Hu, I. W. Tsang, and M. Sugiyama, “Co-teaching: Robust training of deep neural networks with extremely noisy labels,” in *Proc. 32nd Int. Conf. Neural Inf. Process. Syst.*, pp. 8536–8546, 2018.

[33] X. Ma, Y. Wang, M. E. Houle, S. Zhou, S. Erfani, S. Xia, and J. Bailey, “Dimensionality-driven learning with noisy labels,” in *Proc. 35th Int. Conf. Mach. Learn.*, pp. 3361–3370, 2018.

[34] S. Zagoruyko, and N. Komodakis, “Wide residual networks,” *British Machine Vision Conference*, 2016.

[35] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pp. 770–778, 2016.

[36] X. Ma, H. Huang, Y. Wang, S. Romano, S. Erfani, and J. Bailey, “Normalized loss functions for deep learning with noisy labels,” in *Proc. 37th Int. Conf. Mach. Learn.*, pp. 6543–6553, 2020.

[37] D. Hendrycks, M. Mazeika, D. Wilson, and K. Gimpel, “Using trusted data to train deep networks on labels corrupted by severe noise,” in *Proc. 32nd Int. Conf. Neural Inf. Process. Syst.*, pp. 10456–10465, 2018.

[38] Y. Cui, Y. Song, C. Sun, A. Howard, and S. Belongie, “Large scale fine-grained categorization and domain-specific transfer learning,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pp. 4109–4118, 2018.

[39] K. Cao, C. Wei, A. Gaidon, N. Arechiga, and T. Ma, “Learning imbalanced datasets with label distribution-aware margin loss,” in *Proc. 33rd Int. Conf. Neural Inf. Process. Syst.*, pp. 1567–1578, 2019.

[40] iNaturalist 2018 dataset, https://github.com//visipedia//inat_comp, 2018.

[41] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, and M. Bernstein, “Imagenet large scale visual recognition challenge,” *Int. J. Comput. Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[42] G. V. Horn, O. M. Aodha, Y. Song, Y. Cui, C. Sun, A. Shepard, H. Adam, P. Perona, and S. Belongie, “The iNaturalist species classification and detection dataset,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pp. 8769–8778, 2018.

[43] S. Li, K. Gong, C. H. Liu, Y. Wang, F. Qiao, and X. Cheng, “MetaSAug: Meta semantic augmentation for long-tailed visual recognition,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pp. 5208–5217, 2021.

[44] B. Zhou, Q. Cui, X. Wei, and Z. Chen, “Bbn: Bilateral-branch network with cumulative learning for long-tailed visual recognition,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pp. 9719–9728, 2020.

[45] M. A. Jamal, M. Brown, M. H. Yang, L. Wang, and B. Gong, “Rethinking class balanced methods for long-tailed visual recognition from a domain adaptation perspective,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pp. 7610–7619, 2020.

[46] T.-H. Ho, “The random subspace method for constructing decision forests,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 8, pp. 832–844, 1998.

[47] H. Zhang, M. Cisse, Y. N. Dauphin, and L.-P. David, “Mixup: Beyond empirical risk minimization,” in *Proc. 6th Int. Conf. Learn. Representations*, 2018.

[48] F. Yuan, L. Shou, J. Pei, W. Lin, M. Gong, Y. Fu and D. Jiang, “Reinforced multi-teacher selection for knowledge distillation,” in *Proc. 35th AAAI Conf. Artif. Intell.*, pp. 14284–14291, 2021.

[49] C. Zhang, B. Recht, S. Bengio, M. Hardt, and O. Vinyals, “Understanding deep learning requires rethinking generalization,” in *Proc. 5th Int. Conf. Learn. Representations*, 2017.

[50] S. Jiang, J. Li, Y. Wang, B. Huang, Z. Zhang, and T. Xu, “Delving into sample loss curve to embrace noisy and imbalanced data,” in *Proc. 36th AAAI Conf. Artif. Intell.*, pp. 7024–7032, 2022.

[51] H. Sun, C. Guo, Q. Wei, Z. Han, and Y. Yin, “Learning to rectify for robust learning with noisy labels,” *Pattern Recognition*, vol. 124, pp. 108467, 2022.

[52] C. Santiagoa, C. Barataa, M. Sasdellib, G. Carneirob, and J. C.Nasciment, “LOW: Training deep neural networks by learning optimal sample weights,” *Pattern Recognition*, vol. 110, pp. 107585, 2021.

[53] W. Wang, F. Feng, X. He, L. Nie, and T.-S. Chua, “Denoising implicit feedback for recommendation,” in *Proc. WSDM*, pp. 373–381, 2021.

[54] A.-D. Pozzolo, G. Boracchi, O. Caelen, C. Alippi and G. Bontempi, “Credit card fraud detection: A realistic modeling and a novel learning strategy,” *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 8, pp. 3784–3797, 2018.

[55] Y. Artan, M.-A. Haider, D.-L. Langer, T.-H. van der Kwast, A.-J. Evans, Y. Yang, M.-N. Wernick, J. Trachtenberg, and I.-S. Yetik, “Prostate cancer localization with multispectral MRI using cost-sensitive support vector machines and conditional random fields,” *IEEE Trans. on Image Processing*, vol. 19, no. 9, pp. 2444–2455, 2010.

[56] T. Castells, P. Weinzaepfel, and J. Revaud, “SuperLoss: A generic loss for robust curriculum learning,” in *Proc. 34th Int. Conf. Neural Inf. Process. Syst.*, pp. 4308–4319, 2020.

[57] A.-K. Menon, S. Jayasumana, A.-S. Rawat, H. Jain, A. Veit, and S. Kumar, “Long-tail learning via logit adjustment,” in *Proc. 9th Int. Conf. Learn. Representations*, 2021.

[58] K. Cao, C. Wei, A. Gaidon, N. Arechiga, and T. Ma, “Learning imbalanced datasets with label-distribution-aware margin loss,” in *Proc. 33rd Int. Conf. Neural Inf. Process. Syst.*, pp. 1567–1578, 2019.

[59] Y. Wang, X. Pan, S. Song, H. Zhang, C. Wu, and G. Huang, “Implicit semantic data augmentation for deep networks,” in *Proc. 33rd Int. Conf. Neural Inf. Process. Syst.*, pp. 12635–12644, 2019.

[60] S. Li, K. Gong, C.-H. Liu, Y. Wang, F. Qiao, and X. Cheng, “Metasaug: Meta semantic augmentation for long-tailed visual recognition,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, pp. 5208–5217, 2021.

Xiaoling Zhou received the B.Sc. degree in Mathematics from Tiangong University, Tianjin, China, in 2020, and the M.Sc. degree in Mathematics from the Center for Applied Mathematics, Tianjin University, Tianjin, China, in 2023. She is currently pursuing the Ph.D. degree with the National Engineering Research Center for Software Engineering, Peking University, Beijing. Her research interests include data mining, imbalance learning, and meta-learning.

Ou Wu received a B.Sc. degree in electrical engineering from Xi’an Jiaotong University, Xi’an, China, in 2003, and the M.Sc. and Ph.D. degrees in computer science from the National Laboratory of Pattern Recognition (NLPR), Institute of Automation, Chinese Academy of Sciences, Beijing, China, in 2006 and 2012, respectively. In 2007, he joined NLPR as an Assistant Professor. In 2017, he joined the Center for Applied Mathematics, Tianjin University, Tianjin, China, as a Full Professor. His research interests include data mining and machine learning.

Mengyang Li received the B.Eng. degree in automation from the Zhengzhou University of Aeronautics, Zhengzhou, China, in 2015, and the M.Eng. degree in pattern recognition and intelligent control with the Electronic Information and Automation College, Civil Aviation University of China, Tianjin, China, in 2018. He is currently a Ph.D. student with the Center for Applied Mathematics, Tianjin University, Tianjin. His research interests include data mining and deep learning.