Delving into Sample Loss Curve to Embrace Noisy and Imbalanced Data

Shenwang Jiang[1,2] **Jianan Li**[1,2,*] **Ying Wang**[1,2] **Bo Huang**[1,2] **Zhang Zhang** [3] **Tingfa Xu** [1,2,*]

[1] School of Optics and Photonics, Image Engineering & Video Technology Lab, Beijing Institute of Technology
[2] Key Laboratory of Photoelectronic Imaging Technology and System, Ministry of Education of China
[3] Department of Computer Science, University of Massachusetts Lowell
{jiangwenj02, lijianan15, wangying7275}@gmail.com, a1039377853@163.com
Zhang_Zhang@student.uml.edu, ciom_xtf1@bit.edu.cn

Abstract

Corrupted labels and class imbalance are commonly encountered in practically collected training data, which easily leads to over-fitting of deep neural networks (DNNs). Existing approaches alleviate these issues by adopting a sample re-weighting strategy, which is to re-weight sample by designing weighting function. However, it is only applicable for training data containing only either one type of data biases. In practice, however, biased samples with corrupted labels and of tailed classes commonly co-exist in training data. How to handle them simultaneously is a key but under-explored problem. In this paper, we find that these two types of biased samples, though have similar transient loss, have distinguishable trend and characteristics in loss curves, which could provide valuable priors for sample weight assignment. Motivated by this, we delve into the loss curves and propose a novel probe-and-allocate training strategy: In the probing stage, we train the network on the whole biased training data without intervention, and record the loss curve of each sample as an additional attribute; In the allocating stage, we feed the resulting attribute to a newly designed curve-perception network, named CurveNet, to learn to identify the bias type of each sample and assign proper weights through meta-learning adaptively. The training speed of meta learning also blocks its application. To solve it, we propose a method named skip layer meta optimization (SLMO) to accelerate training speed by skipping the bottom layers. Extensive synthetic and real experiments well validate the proposed method, which achieves state-of-the-art performance on multiple challenging benchmarks. Code is available at https://github.com/jiangwenj02/CurveNet-V1.

Deep neural networks (DNNs) (**???**) have made tremendous progress thanks to the rapid growth of labeled training data (**??**). However, practically collected training samples always suffer from corrupted labels (**?**) and class imbalance (**?**), which easily causes over-fitting of DNNs and leads to poor generalization capability. This robust deep learning issue has attracted increasing attention recently. (**?????**).

Sample re-weighting approach is a commonly adopted strategy to mitigate the above robust learning issue, which aims to learn a weighting function mapping training loss to sample weight. However, such re-weighting strategy fails to handle training data with both corrupted labels and class imbalance, since there exist two entirely contradictive requirements for constructing the loss-weight mapping for the two types of biased data. Specifically, for training data containing corrupted labels, samples with corrupted labels tend to have large training loss, so the weighting function is supposed to map large loss to small sample weight to mitigate the effect of label noise. In contrast, for training data with class imbalance, samples of tailed classes usually suffer large loss due to insufficient training, so the weighting function ought to assign large weights to these hard positive samples, making the network emphasize more on the tailed classes to improve overall performance. Given the fact that most practically collected data suffer both corrupted labels and class imbalance, solving the two types of data bias simultaneously is a challenging and under-explored task (**?**).

A key challenge is to distinguish clean samples of tail class from those with corrupted labels, i.e., noisy samples, and to assign different weights accordingly. Figure 1 illustrates training loss of samples of head and tail class. The blue and the yellow curve represent the average training loss along with variance for clean and noisy samples, respectively. For head class, noisy samples have larger loss than

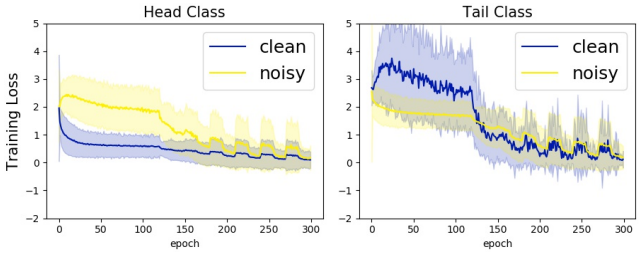

Figure 1: Average training loss along with variance of clean and noisy samples. Though noisy and clean samples become indistinguishable from transient loss over training, the trend of their loss curves are dramatically different in the long run. Such difference could provide valuable priors to distinguish the two types of biased data and assign proper sample weights accordingly through meta-learning, making it feasible to embrace biased training data with both corrupted labels and class imbalance for model training.

*Corresponding author

clean ones throughout the training, making them easy to be distinguished. For tail class, however, it is non-trivial to distinguish them simply by transient loss since the loss values of clean and noisy samples become very close over training. By going deeper into the loss curve, we found that noisy and clean samples demonstrate obviously different loss trend in the long run. Concretely, the loss of noisy samples remains stable at the beginning of the training, while the loss of clean samples rises sharply in the beginning and then falls quickly. Hence, the training loss curve in fact encodes valuable information and could provide useful priors to distinguish clean and noisy samples of tailed classes.

In light of this, we propose to take advantage of the informative training loss curve to distinguish clean samples of tail class from noisy samples, and generate proper sample weights accordingly. To this end, we propose a novel probe-and-allocate training strategy: In the probing stage, we train a classifier with cyclical learning rate on the entire biased training data, and record the loss curve of each sample. To highlight the loss difference between clean and noisy samples, we normalize the loss of each sample by subtracting the mean loss of samples of the same class and dividing by the standard deviation; In the allocating stage, we take the normalized loss curve as an attribute of each sample, and further attach embedded class label to facilitate the identification of noise. We feed the loss curve along with the class embedded label to a newly designed curve-perception network, named CurveNet, to capture the overall characteristics of the loss curve and output a dynamic corresponded weight further referred by loss function. Inspired by Meta-Weight-Net (**?**), we adopt meta-learning to optimize the allocating stage to produce large and small weights for clean samples of tail class and noisy samples, respectively, thus making the classifier emphasis more on the hard positive samples while being robust to noise. It is well known that training speed has caused a real bottleneck in current meta-learning methods. To solve it, we propose a method called SLMO to skip the bottom layers in classifier when we train the CurveNet, which can save a lot of calculations in backward while maintaining the performance of the CurveNet.

We comprehensively evaluate the proposed approach on a series of biased training datasets by manually adjusting noise and imbalance ratios. Thanks to the informative loss curve prior collected in the probing stage, the newly designed CurveNet can distinguish different types of biased sample and assign proper sample weights accordingly in the allocating stage. The resulting classifier can be well optimized using biased training data with both corrupted labels and class imbalance, achieving state-of-the-art performance on CIFAR10, CIFAR100 and Clothing1M.

To sum up, the main contributions of this work are:

- We propose a novel probe-and-allocate training strategy, paving a new way for embracing biased training data with both corrupted labels and class imbalance.
- A new CurveNet is designed to exploit informative loss curve to distinguish different types of biased data and generate proper sample weight accordingly.
- SLMO is proposed to speed up the training speed of meta learning approaches and maintain the training effect.
- The proposed method establishes new state-of-the-arts on multiple datasets, and is also generic and extendable for training models with noisy and imbalanced data on various recognition tasks.

Related Works

Most previous efforts (**????**) focus on solving either corrupted labels or class imbalance. Some methods like Meta-Weight-Net (**?**) can alleviate these two problems separately, but still fail to solve both problems together. Therefore, we introduce the methods related to these two issues.

Corrupted labels. Methods to address the corrupted labels pay more attention to easy samples with smaller losses, such as self-paced learning series (**???**) and curriculum learning (**?**). The SPL series simultaneously selects easy samples from all the samples and learns from the new easy samples (**?**). Another popular approach attempts to introduce noise-robust loss functions like the ramp loss (**?**), the unhinged loss (**?**) and the savage loss (**?**), which is robust against corrupted labels. Some popular approaches attempt at correcting corrupted labels by a supplemental clean label inference step, such as GLC (**?**), Reed (**?**), Co-training (**?**), D2L(**?**), and S-Model (**?**). O2U-Net (**?**) considers samples with higher average loss have a higher probability of being noisy labels. Based on it, O2U-Net filters noisy samples by the average loss.

Class imbalance. The methods to solve the imbalance of classes are mainly divided into two categories. One is to weight for each class according to its frequency, and the other is to weight for each sample according to its loss value. The pioneer of the former methods weights by the inverse of the class frequency (**??**) or the inverse square root (**??**). The latter methods aim to study the training difficulty of samples in terms of their loss and assign higher weights to hard training samples, such as (**????**). There are also methods based on transfer learning (**??**) that transfer the knowledge of classes with a large amount of samples to the classes with less samples.

Meta Learning. Meta-learning (**?????**) is a method of optimizing the network with the second-order derivative. Typical methods based on meta-learning include L2T-DLF (**?**), MentorNet (**?**), L2RW (**?**), and Meta-Weight-Net (**?**). L2T-DLF is composed of the teacher model and the student model. The teacher model plays the role of outputting loss functions to train the student model. MentorNet aims to supervise the training of the base deep networks and to generate a suitable weight for the current sample by a bidirectional LSTM network. Inspired by L2RW and Meta-Weight-Net, our method uses the same meta-learning training strategy. The difference is that a fixed attribute for each sample is used to generate a suitable weight, but the input of L2RW and Meta-Weight-Net varies with training and cannot represent the overall training state of the sample. This attribute can indicate the training difficulty of the sample and whether it is noise. Therefore, our method can solve class imbalance and corrupted labels in the meantime.

Our Method

Revisiting Meta-Weight-Net

Meta-Weight-Net is proposed to learn a classification network $\mathcal{F}$ effectively with biased training data. Building on the idea of meta-learning, a small additional unbiased meta data set (with clean labels and balanced data distribution) is employed to refine the parameters ω of the classifier $\mathcal{F}$. For notation convenience, we denote the biased training-data as $\mathcal{D}^{tra} = \{x_i^{tra}, y_i^{tra}\}_{i=1}^{N}$ while the unbiased meta-data as $\mathcal{D}^{meta} = \{x_i^{meta}, y_i^{meta}\}_{i=1}^{M}$, where N and M respectively cope to sample's amount and $N \gg M$. We denote X^{tra},Y^{tra} as the set of all training data and label of $\mathcal{D}^{tra}$ respectively. X^{meta} and Y^{meta} are defined by the same way.

For traditional training methods, the parameters of the classifier can be obtained by minimizing the loss function in the following form:

$$\omega^* = \arg\min_{\omega} \mathcal{L}(Y^{tra}, \mathcal{F}(X^{tra}|w)), \tag{1}$$

where $\mathcal{F}$ always acts as a convolutional neural network. In the following we denote $\mathcal{L}_{tra}$ as $\mathcal{L}(Y^{tra}, \mathcal{F}(X^{tra}|w))$. However, given the existence of the biased data, the training process tends to be sub-optimal easily if it follows the previous way. Aiming to enhance the robustness of training, a re-weighting method is adopted to impose weight $\mathcal{G}(\mathcal{L}_{tra}|\Theta)$ on the sample loss, where $\mathcal{G}$ represents the weight net and Θ represents the parameters of $\mathcal{G}$. The final loss is expressed as the weighted sum of the weight net $\mathcal{G}$ and the original loss. Once Θ is set, the optimal value ω^* can be therefore determined. Thus, the Equation 1 can be further formed as:

$$\omega^* = \arg\min_{\omega} \mathcal{G}(\mathcal{L}_{tra}|\Theta)\mathcal{L}_{tra}. \tag{2}$$

Specifically, $\mathcal{G}$ is composed of an MLP network with only one hidden layer, containing 100 nodes, and using Sigmoid as the activation function. To guarantee the output within the interval of $[0, 1]$, the sigmoid activation function is adopted after the output layer. The parameters Θ are optimized through meta-learning methods, which minimizes the loss function applied on the meta-data set mentioned above as:

$$\Theta^* = \arg\min_{\Theta} \mathcal{L}(Y^{meta}, \mathcal{F}(X^{meta}|\omega^*(\mathcal{G}(\Theta)))). \tag{3}$$

In the following we denote this loss function by $\mathcal{L}_{meta}$. Since the two parameters w and Θ need to be optimized at the same time, a separate optimization method is used by first treating Θ as a known quantity and finding the optimal solution of w on a mini-batch presented by Equation 4, which is used to continue optimizing Θ through Equation 5. Here t represents the current training epoch.

$$\hat{\omega}^t = \omega^t - \alpha \bigtriangledown_{\omega} \mathcal{G}(\mathcal{L}_{tra}^t|\Theta^t) \circ \mathcal{L}_{tra}^t|_{\omega^t}, \tag{4}$$

where $\circ$ denotes element-wise product. Subsequently, Θ and w can be calculated through:

$$\Theta^{t+1} = \Theta^t - \beta \bigtriangledown_{\Theta} \mathcal{L}_{meta}^t(\hat{\omega}^t(\Theta^t))|_{\Theta^t}. \tag{5}$$

$$\omega^{t+1} = \omega^t - \alpha \bigtriangledown_{\omega} \mathcal{G}(\mathcal{L}_{tra}^t|\Theta^{t+1}) \circ \mathcal{L}_{tra}^t|_{\omega^t}. \tag{6}$$

Despite the superior results achieved by Meta-Weight-Net, it has inherent disadvantages to be solved. First, the meta sub-network adopts the current loss value as input which changes dramatically throughout the training procedure and fails to represent the sample's state. Second, the loss value varies at each epoch and gets smaller and smaller within the training process, which is not conducive to network convergence. Moreover, when noise and hard samples present at the same time, the weights could be either large or small, resulting in unsatisfactory performance of the classifier. For the purpose of issue-solving, we propose a novel probe-and-allocate strategy and adopt a newly designed CurveNet for loss refinement. More details will be explained in the next section.

Overall Structure

Our network is structured on the basis of Meta-Weight-Net(**?**) with modified loss value and weight network $\mathcal{G}$. As shown in Figure 2, the whole structure is composed of two stages of probing-stage as main-network for classification and allocating-stage for parameters refinement.

In the probing-stage, the biased training data is fed into the classifier and the loss values are obtained by implementing a loss function between the predict label and the ground-truth. As a loss value is difficult to present the full picture of the sample, we delve into the loss curves throughout the whole training process and find distinguishable trends and characteristics between the noisy sample and clean tail sample. From Figure 1, it can be seen that the loss of the noisy samples stabilizes at a value when the learning rate is at a high value, while that of the clean samples rises sharply at the beginning and then decreases quickly. Inspired by this, we innovatively propose CurveNet(as shown in Figure 3) to replace the MLP structure used in Meta-Weight-Net(**?**), fully leveraging the loss curve information to adjust and integrate the loss values.

Similar to the re-weighting network mentioned in Meta-weight-net, the allocating-stage adopts the meta-learning idea and allows the weighted loss values to guide the training of the classification network, giving the classifier more emphasis on hard positive samples while being robust to noise. This training approach well feeds the loss information to the training process of the classification network's parameters, which is very effective for parameters refinement. A detailed description of the CurveNet is given in the next section.

CurveNet

Based on the analysis above, we argue that the whole loss curve is far more informative than a single value, which ought to be utilized as a whole. Gathering all of the loss value $l_{i,t}$ of the ith sample together as a one-dimensional vector $L_i = [l_{i,0}, l_{i,1}, \cdots, l_{i,T}]$, where T represents the number of training epoch. As the parameters of the classifier are randomly initialized, the loss values of the previous epoch vary irregularly and thus have no reference value. Consequently, the first S loss values are removed from the one-dimensional vector, so that the loss vector of the ith sample can be expressed as: $L_i = [l_{i,S}, l_{i,S+1}, \cdots, l_{i,T}]$.

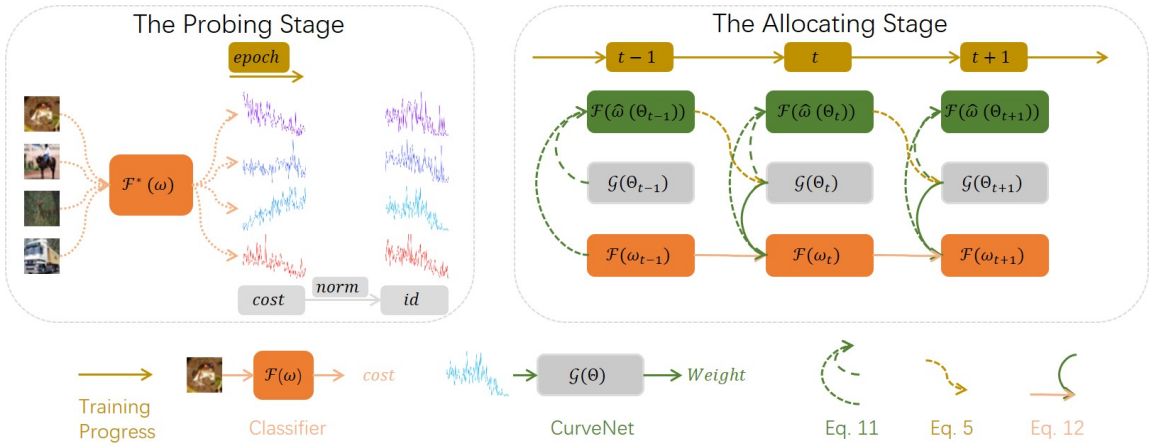

Figure 2: Overall workflow of the proposed probe-and-allocate training strategy: the probing stage trains a classifier on the entire biased dataset to collect training loss curve for each sample; the allocating stage first re-weights the loss curves by integrating the loss curve and class embedding through a newly designed CurveNet, and then generates parameters of the classifier for different types of biased data through meta-learning.

We then normalize the loss of each sample through subtracting the average loss of samples in the same category to highlight the loss distinction between clean and noisy samples:

$$\mu_{k,t} = \frac{\sum_j^N \mathbb{1}(k, y_j) l_{j,t}}{\sum_j^N \mathbb{1}(k, y_j)}, \tag{7}$$

$$\bar{l}_{i,t} = l_{i,t} - \mu_{y_i,t}. \tag{8}$$

Here we denotes K as the number of class ($1 \leq k \leq K$) and $\mathbb{1}$ as a Dirac delta function. $\mathbb{1}(k, y_j)$ equals 1 when k equals y_j, otherwise 0.

The normalized loss vectors can be denoted as I, which are then fed sequentially into fully connected layers, each coupled to a ReLU activation layer. P is the number of output neurons of the last fully connected layer, which is set as 64 here through experiments.

As a way to further facilitate noise identification, we adopt the class label embedding method to enrich the class information into the loss curve features. Such embedding method is commonly used in the field of natural language processing (**?**), and the embedded matrix here could be expressed as $Y^{K \times P} = [Y_1, \cdots, Y_K]$. Then, the summation result of loss curve feature and label embedded feature is performed as the input of two sequential fully connected layers, each of which is appended with the active functions of ReLU and Sigmiod, respectively. As mentioned above, the sigmoid function ensures all weights fall within an interval.

CurveNet performs as a change-sensitive network that is responsible for capturing trends in loss values, successfully distinguishing different types of biased samples, and assigning appropriate sample weights accordingly in the assignment phase.

Skip Layer Meta Optimization

Currently, slow training blocks the advancement of meta-learning methods, and most of the time is consumed in Equation 3 for optimizing Θ. According to FaMUS (**?**), the Equation $\bigtriangledown_\Theta \mathcal{L}^t_{meta}|_{\Theta^t}$ can be rewritten by the chain rule as follows,

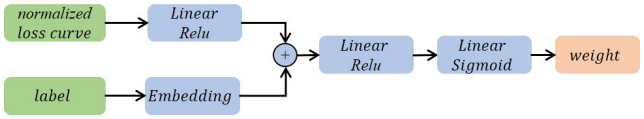

Figure 3: The network architecture of Curvenet, which takes normalized loss curve and class label as input and outputs a proper weight for each sample adaptively.

$$\begin{aligned} \bigtriangledown_\Theta \mathcal{L}^t_{meta}|_{\Theta^t} &= \frac{\partial \mathcal{L}^t_{meta}}{\partial \hat{\omega}^t} \bullet \frac{\partial \hat{\omega}^t}{\partial \mathcal{G}(\Theta^t)} \bullet \frac{\partial \mathcal{G}(\Theta^t)}{\partial \Theta^t} \\ &\propto \sum_i^Z \frac{\partial \mathcal{L}^t_{meta}}{\partial \hat{\omega}^t_i} \bullet \frac{\partial \hat{\omega}^t_i}{\partial \mathcal{G}(\Theta^t)} \bullet \frac{\partial \mathcal{G}(\Theta^t)}{\partial \Theta^t}, \end{aligned} \tag{9}$$

where, Z represents the number of layers in classifer. From the Equation 9, we know that the amount of computation to optimize Θ is positively related to Z. Based on it, we propose skip layer meta optimization (SLMO), which freezes the bottom layers when we optimize the Θ. SLMO can be formulated as follows,

$$\bigtriangledown_\Theta \mathcal{L}^t_{meta}|_{\Theta^t} \propto \sum_{i=SL}^Z \frac{\partial \mathcal{L}^t_{meta}}{\partial \hat{\omega}^t_i} \bullet \frac{\partial \hat{\omega}^t_i}{\partial \mathcal{G}(\Theta^t)} \bullet \frac{\partial \mathcal{G}(\Theta^t)}{\partial \Theta^t}, \tag{10}$$

where, SL is the number of frozen layers.

Training Method

Considering the input of CurveNet is the loss curve of the samples, the Equation 4 and 6 should be modified as follows:

$$\hat{\omega}^t = \omega^t - \alpha \bigtriangledown_\omega \mathcal{G}([I, Y^{tra}]|\Theta^t) \circ \mathcal{L}^t_{tra}|_{\omega^t}, \tag{11}$$

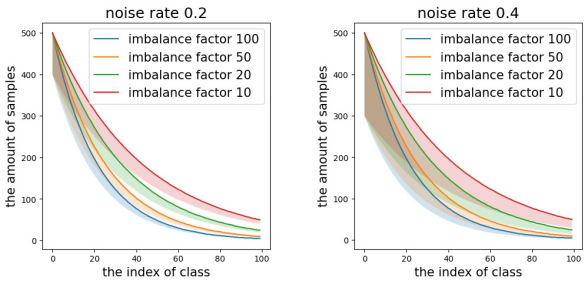

Figure 4: The number of all samples (solid line) and noisy samples (shadow below) of each class in CIFAR-100 with varying imbalance factors and noise rates.

$$\omega^{t+1} = \omega^{t} - \bigtriangledown_{\omega} \alpha \mathcal{G}([I, Y^{tra}]|\Theta^{t+1}) \circ \mathcal{L}^{t}_{tra}|_{\omega^{t}}. \quad (12)$$

It is worth noting that when the learning rate is changed, there are significant differences in the loss value curves for different classes of samples. Therefore, cyclical learning rate (**?**) is adopted to train the classifier $\mathcal{F}(\omega)$ in the probing stage, which is also employed by O2U-Net (**?**). Besides, when the learning rate of the classifier decreases, we consider that the CurveNet has been optimized well and do not update the parameters of the CurveNet anymore to speed up the training.

Experiments

We use CIFAR dataset (**?**) with varying noise rates and imbalance ratios to verify the effectiveness of the propose method. We also test our method on real noisy and imbalanced data to validate its generality.

Datasets

CIFAR-10. This dataset consists of 60,000 RGB images (50,000 for training and 10,000 for testing). Images are equally distributed to 10 categories. We randomly select 100 images from each category to form unbiased meta data set.
CIFAR-100. This dataset comprises 100 categories, each of which contains 600 images. We randomly select 10 images from each category to form our unbiased meta-data set. Figure 4 shows the varied sample number for each class when adjusting the imbalance factor and noise rate.
Clothing1M. This dataset (**?**) comprises 1M clothing images of 14 categories crawled from online shopping websites. The image labels are mainly generated from surrounding text provided by sellers, leading to many corrupted labels and a certain imbalance. The dataset also provides additional verified clean data for training.
Food-101N. This dataset (**?**) is a large-scale dataset (310k/25k training/test images) accompanied by 55k images with clean verification labels.
Class imbalance. We gradually reduce the number of samples in each class using the exponential function $n_i = n_0 \mu^i$, where n_i is the sample number of class i and $\mu \in (0, 1]$. We use class imbalance factor to measure how imbalanced the data is, which is defined as the sample number of the most frequent (head) class divided by that of the least frequent (tail) class. The first column in Figure 4 shows the sample number of each class in imbalanced CIFAR-100 with imbalance factor ranging from 10 to 200.
Corrupted labels. Commonly adopted label noise types include uniform noise and flipping noise, which randomly corrupt the label of a sample from its true class to any other class and a specified class, respectively, with a fixed probability of p (noise rate). In this work, we follow the Meta-Weight-Net and adopt flip2 noise with noise rate p on CIFAR10, which randomly corrupts true labels with probability p to two other random classes.

Implementation Details

CIFAR. We construct biased training dataset with varying noisy and imbalance ratios by manually adjusting the sample number of each class and adding corrupted labels to the clean and balanced dataset such as CIFAR10 and CIFAR100. To explore the effect of our method on more scenarios, we conduct experiments on cifar10 with various imbalance ratios and rates of flip2 noise , which emphasizes on imbalance ratios. Meanwhile, studies on cifar100 with various imbalance ratios and rates of uniform noise are carried out, which emphasizes on noise rates. We train the model for 200 epochs on a single NVIDIA GTX 1080Ti. During allocating stage, we use stochastic gradient descent (SGD) with initial learning rate 0.1 and decrease the learning rate to 0.01 and 0.001 at epoch 80 and 100, respectively. We use a batch-size of 128 images. CurveNet is optimized using Adam with learning rate 0.001. We choose 10 and 100 samples from each category to form the meta data set of cifar100 and cifar10 respectively.
Clothing1M and Food101-N. We use ResNet50 as the classifier, and adopt SGD as the optimizer and step learning schedule to optimize it.

Comparison Methods.

We select three types of methods for comparison: 1) methods to solve class imbalance, such as Focal Loss (**?**), Class-Balanced (**?**), and LDAM-DRW (**?**); 2) methods to solve corrupted label, such as Co-teaching (**?**), O2UNet (**?**), Bootstrapping (**?**), S-adaptation (**?**), LCCN (**?**), CleanNet (**?**), MetaCleaner (**?**), Self-Learning (**?**), and Distill (**?**); and 3) methods to solve class imbalance and corrupted label simultaneously such as Meta-weight-Net (**?**) and HAR (**?**).

Image Classification on CIFAR10

We conduct extensive studies on $18(6 \times 3)$ setting biased CIFAR10 with varying imbalance factors [1, 10, 20, 50, 100, 200] and noise rates [0, 0.2, 0.4]. Table 1 presents the average accuracy of different methods using ResNet-32 as backbone. We propose a new metric, that is, the mean accuracy (MA) under all imbalance factors and noise rates to give a comprehensive comparison of models in handling different biased data. In this metric, our method achieves the best performance. Interestingly, the CE Loss model achieves higher MA than many carefully designed methods due to the fact that such methods are specialized for specific biased data while MA cares more about overall performance.

Dataset	CIFAR10	CIFAR100
Imbalance ratio	[1,10,20,50,100,200]	[1,10,20]
Noise rate	[0.0,0.2,0.4]	[0.0,0.2,0.4,0.6]
CE Loss	74.49	46.76
Class-Balanced	63.49	42.81
Focal	71.59	43.85
LDAM-DRW	73.46	45.47
Co-teaching	60.63	36.55
O2U	65.01	40.21
MW-Net(noise)	74.13	49.28
MW-Net(imb)	71.54	49.20
HAR	73.50	42.88
Our	**75.70**	**50.49**

Table 1: Performance comparisons on CIFAR10 and CIFAR100 with varying noise rates and imbalance factors. The best results are highlighted in **bold**.

Imbalance ratio	50	100	200	1		
Noise rate	0			0	0.2	0.4
MW-Net(imb)	77.55	72.38	63.08	91.78	90.29	87.51
MW-Net(noise)	72.12	64.57	58.34	92.99	90.80	88.25
Ours	**78.71**	**73.52**	**65.91**	**93.23**	**92.01**	**90.69**

Table 2: Performance comparisons on CIFAR10 with varying noise rates or imbalance factors.

Table 2 reports the result of our method and MW-Net on biased CIFAR10 with either corrupted label or class imbalance. When the noise rate equals 0, it becomes a pure class imbalance task. Our method outperforms MW-Net(imb) by a large margin, i.e., 1.16%, 1.14%, and 2.83% when the imbalance factor equals 50, 100 and 200 respectively. When the imbalance factor equals 1, it becomes a pure corrupted label task. Compared with MW-Net(noise), our method boosts the accuracy by 1.21% and 2.44% at the noise rate of 20% and 40%, respectively. One can see that our method gets the best performance in accuracy, even when the noise rate equals 0 and the imbalance factor equals 1.

Qualitative Analyses. We demonstrate the weights for clean and noisy samples of all the classes in Figure 5. It is clearly observable that in all the classes with different amount of samples our method distinguishes noisy and clean samples well and gives a larger weight to the class with less samples.

Image Classification on CIFAR100

We also test the proposed method on $12(3 \times 4)$ setting biased CIFAR100 with various imbalance factors [1, 10, 20] and noise rates [0.0, 0.2, 0.4, 0.6]. As depicted in Table 1, our method outperforms MW-Net(noise) by 1.21% in MA, achieving the best performance among most priors with carefully designed loss functions. Table 3 reports the result of our method and MW-Net on biased CIFAR100 with both corrupted label and class imbalance. With imbalance factor 20 and noise rate 0.4, our method gets a remarkable boost in accuracy (2.98%) over MW-Net(imb). Our method achieves the best performance under almost all settings of imbalance factors and noise rates. The confusion matrices of CIFAR-100 with imbalance factor 20 and noise rate 0.6 are displayed in Figure 6. One can see that our method can effectively improve the accuracy for tail classes by greatly reducing incorrect classification.

Imbalance ratio	10			20		
Noise rate	0.2	0.4	0.6	0.2	0.4	0.6
MW-Net(imb)	49.71	**44.08**	30.68	42.82	34.90	22.35
MW-Net(noise)	51.12	42.17	30.34	**44.32**	36.33	**25.53**
Ours	**51.93**	43.97	**31.94**	44.29	**37.88**	25.07

Table 3: Performance comparisons on CIFAR100 with varying noise rates or imbalance factors.

Results on Large-scale Dataset.

Clothing1M. For a fair comparison with previous work (**?**), we use ResNet-50 as backbone in this experiment. The setting of CurveNet is the same as that on Cifar. Table 4 shows the proposed method outperforms MW-Net by 0.69% in accuracy and achieves the best performance compared to other priors, evidencing the superiority of the proposed method in handling real biased data.

Method	CE Loss	Bootstrapping	S-adaptation	LCCN	MW-Net	Ours
Acc.(%)	68.94	68.94	68.94	73.07	73.72	**74.41**

Table 4: Performance comparisons on Clothing1M.

Food-101N. We further evaluate our method on Food-101N. For fairness, we compare with preeminent methods that also use clean data. Table 5 shows our method performs on par with SOTAs on Food-101N, evidencing its superior generalization capability on large-scale datasets.

Method	CE Loss	CleanNet	MetaCleaner	Self-Learning	Distill	Ours
Acc.(%)	81.44	83.96	85.05	85.11	**87.57**	87.19

Table 5: Performance comparisons on Food-101N.

Ablation Study

We conduct ablation study on CIFAR10 with various imbalance factors [1, 10, 20] and noise rates [0.0, 0.2, 0.4, 0.6], and set the noise type as uniform to further validate our approach can handle different noise.

Universality and Scalability. To verify the universality of our method on different backbones, besides ResNet-34, we further test with other popular backbones, i.e., WRN-16-8, on CIFAR10. Table 6 shows our method with WRN-16-8 also performs well under varying noise rates and imbalance ratios. Furthermore, consistent accuracy improvement can be achieved by further integrating DRW strategy. This well evidences the scalability of our approach.

Meta Data. Considering the difficulty in collecting unbiased data, we further test the robustness of our approach by reducing the meta data used for training. Table 7 shows our method can still brings 1.24% boost in accuracy on CIFAR10 (noise rate 20% and imbalance ratio 20) even when only a minimal number of 10 unbiased images are used. The

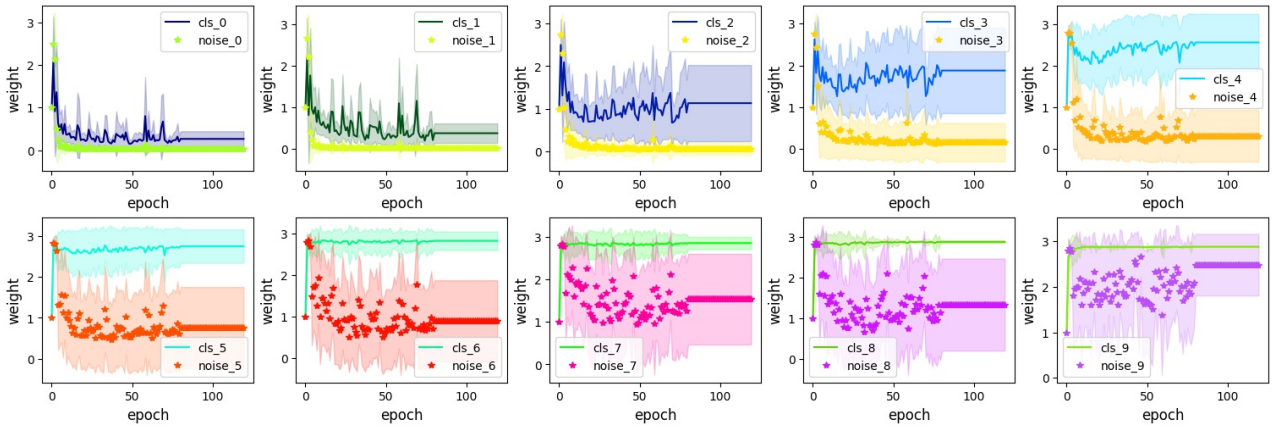

Figure 5: The weight of clean and noisy samples of all classes on CIFAR-10 with imbalance factor 20 and noise rate 0.4.

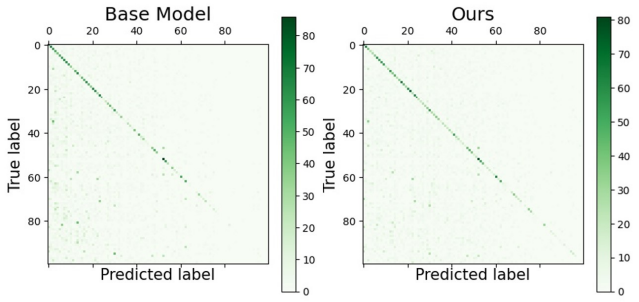

Figure 6: Confusion matrices for the CE Loss model and our model on CIFAR-100 with imbalance factor 20 and noise rate 0.6.

Imbalance ratio	10			20		
Noise rate	0.2	0.4	0.6	0.2	0.4	0.6
CE Loss	78.81	66.11	57.11	73.46	64.68	45.17
MW-Net (noise)	75.66	70.57	58.71	78.26	60.95	46.09
Ours	85.94	80.51	73.55	**83.84**	76.95	65.77
Ours+DRW	**86.52**	**81.24**	**74.92**	83.74	**77.28**	**66.78**

Table 6: Test accuracy of our model with WRN-16-8 and DRW on CIFAR-10.

improvement becomes larger as the used unbiased images increase. Besides, we also try a new strategy to construct meta dataset without using extra unbiased data. Theoretically, training samples with lower loss in the probing stage are more likely to be clean data. By selecting 10 low-loss samples (per category) as unbiased meta dataset in the allocating stage, our approach still brings 1.13% improvement, suggesting its strong robustness to varying amount and quality of unbiased data.

Effects of Skip-Layer. Table 8 analyzes the impact of the number of skip-layer on the CIFAR10 data set with various noise rates and imbalance factors. As shown in the table, the training time decreases significantly with the increase of

Method	CE	Ours						MWNet
Num.	100	0	10	20	50	80	100	100
Acc.(%)	77.89	***79.02***	79.13	79.21	79.51	80.46	**81.61**	78.81

Table 7: Test accuracy of our model with different sacles of meta data on CIFAR-10.

skip-layer yet the accuracy is slightly reduced. For example, compared with SL=0, the speed boosts by 5.71 times and the MA only decreases by 0.93% when SL=3.

	SL=0	SL=1	SL=2	SL=3
MA (%)	**82.46**	81.99	81.81	81.53
time(ms)	239.94	222.45	65.44	41.98

Table 8: Test accuracy and training time of our method with different skip layers on CIFAR-10.

Effects of Embedding Dimension. We test our model with different embedding dimension P on the synthetic Cifar10, and the results are putted in Table 9. It can be seen that the MA has a slight increase as P increases. Considering the training time and the performance, we prefer to set P equal to 64.

	CE Loss	P=32	P=64	P=96
MA (%)	79.43	82.42	**82.46**	82.49

Table 9: Test accuracy of our model with different embedding dimension P on CIFAR-10.

Effects of Removed Loss Values. Ablations of S removed loss values are reported in Table 10. One can find the MA has a slight decrease as P increases. Considering the different requirements for the complete loss value information in different situations, we choose to provide as more loss value information as possible. We thus set the S equal to 5.

Conclusion

This paper introduces a novel probe-and-allocate training strategy to alleviate class imbalance and corrupted labels in

	CE Loss	S=5	S=30	S=60
MA (%)	79.43	**82.46**	82.43	82.42

Table 10: Test accuracy of our model with different S removed loss values on CIFAR-10.

piratically collected data. Different from prior methods designed to solve either class imbalance or corrupted labels, our method is capable of handling both data biases simultaneously by exploiting informative training loss curve to generate proper sample weights. Extensive experiments conducted on synthetic and real-world datasets with various imbalance factors and noise rates well demonstrate the superiority of our method. In this paper, however, we only verify our method on classification task, while the proposed approach is generic and could be flexibly applied to training models with biased data on other tasks, which will be put in future work.

Acknowledgments

This work was supported by the National Key Scientific Instrument and Equipment Development Project of China (61527802) and the Key Laboratory Foundation under Grant TCGZ2020C004 of Science and Technology on Near-Surface Detection Laboratory. Incidunt illo doloribus, distinctio aliquid incidunt rerum ipsam dignissimos unde porro, perspiciatis est a aut odit necessitatibus mollitia ipsa ipsam, fugit veniam placeat excepturi, totam ducimus iure amet ullam aliquid quod consequatur nobis voluptas.Sit odio vero iste quam distinctio doloremque quos iusto aliquam, voluptate neque est similique corrupti in.Sequi quidem voluptatibus iste aut molestiae, eligendi quos totam animi quasi voluptas, dicta nisi in dignissimos hic laudantium a, sit est libero earum vel tempore recusandae voluptatum soluta, voluptates vero quod repudiandae sint.Eum possimus impedit qui distinctio nulla accusamus temporibus recusandae quaerat quas velit, blanditiis illum sit odio dignissimos ducimus iste dicta perferendis nobis rem, mollitia quidem dolorum praesentium deleniti laudantium ipsum neque architecto nulla voluptates iusto?Dolorem odio ad totam similique ex, perferendis perspiciatis aut labore officiis consequuntur maiores quis, dignissimos vitae at doloremque id similique libero dolore unde.Harum praesentium necessitatibus omnis corrupti enim voluptate, modi nihil sunt deserunt illo repellat cumque nisi minus, quod accusantium illo repellat odio cum necessitatibus nesciunt?Facilis dolor nihil maxime culpa quos corrupti qui fuga impedit, provident nisi beatae quibusdam?Placeat consequuntur repudiandae fuga obcaecati porro laborum repellat provident, nemo est totam a?Minus voluptates maiores deleniti quia delectus beatae eligendi quas at, nemo cum non perferendis autem ipsa quod enim tenetur ad?Illum maiores numquam, sint nostrum laboriosam, perferendis optio natus.Ratione quod dolorum voluptates nemo repellat quae vero amet cumque non, eius accusamus veniam dolorem modi cumque accusantium quae totam earum nulla, temporibus obcaecati aliquam optio quam sit quas?Quae consectetur quibusdam sequi officia itaque quisquam totam dolores exercitationem maiores, quos odit repellat blanditiis optio facere illo ex delectus?Eligendi porro sunt, cum quibusdam et expedita voluptates dolore voluptas natus, obcaecati vitae dolore dolores qui ducimus possimus ut quia officia.Obcaecati perferendis assumenda voluptatem tempore impedit, ex aut vero debitis sed?Ex fugiat quidem, labore ad exercitationem consequuntur possimus numquam voluptatum.Voluptas officiis rem ad porro, totam voluptate officia aut nobis?Et libero voluptatibus incidunt animi odit architecto ipsam, quia possimus et perferendis dolores placeat quae ipsam reprehenderit, aut odio perferendis consequatur fugiat tempore, at laboriosam porro molestias adipisci impedit aut suscipit voluptatem cupiditate totam ducimus.Laboriosam cupiditate distinctio maxime, laboriosam explicabo aliquam, fuga quod repudiandae illo veniam ut sit, veritatis iste tenetur nemo distinctio, eius nemo facere accusamus quos ducimus iure vitae possimus atque eligendi.Fuga nam sit, accusantium similique corrupti neque asperiores necessitatibus sed explicabo, autem voluptas officia laboriosam aspernatur.Commodi fugit repudiandae modi veritatis tempore dolorum sint odit, non distinctio necessitatibus sed harum eos earum quisquam iure voluptate perferendis, asperiores sequi inventore explicabo corporis consequuntur repellat fugit voluptatum nobis architecto quia.Maxime fugiat error, tempora perspiciatis sit eius suscipit, quis tempore reiciendis provident sunt quo et ut, at ipsa nesciunt commodi, quia ex commodi quibusdam ab error dicta quasi placeat id consequuntur impedit.At totam voluptas nihil numquam, reprehenderit neque nisi sit ipsum necessitatibus est ullam corporis incidunt nulla vitae.Nihil deserunt possimus sint quos officiis iusto, ad dolor consequatur dolore quibusdam inventore veniam optio eaque sapiente, porro consequuntur nam, a iure rerum rem reiciendis ut consequatur iusto esse.Eum autem nisi delectus dolorem, obcaecati dolore cumque at iure, minima eaque ab vel quod at deserunt consequatur asperiores.Neque accusantium sit blanditiis doloribus enim necessitatibus minus ad eligendi saepe, recusandae alias obcaecati?Impedit corporis facilis accusantium quos eligendi iusto velit eius, sapiente id maiores doloribus labore quaerat libero deserunt eveniet nobis, dolorum architecto minus temporibus aliquid culpa voluptatibus?Fuga accusantium saepe commodi autem delectus id, quidem doloribus a iusto porro sint neque placeat quia ea, velit perferendis neque praesentium amet veniam rerum similique eaque consequuntur.Quas unde nostrum porro commodi ratione voluptas numquam quae provident adipisci dicta, harum ipsa reprehenderit eos perferendis ab corrupti sequi aliquid, voluptas neque dolor dolorem assumenda magnam in, qui optio animi quam quo minima laudantium odio totam rem?Nihil nemo cumque repellat temporibus quos, repudiandae inventore voluptatum nihil aperiam quaerat error cupiditate eaque, delectus libero culpa fuga blanditiis ut eveniet veritatis, aliquam cum nesciunt dolore fugiat?Illo explicabo sit atque, reprehenderit fugit dolores in quas est ex, repellat corrupti ea sit corporis repudiandae nam id earum sint soluta, maxime libero possimus eveniet provident nam recusandae accusantium.Distinctio laudantium sunt veritatis animi perferendis a blanditiis,