Learning from Noisy Labels via Dynamic Loss Thresholding

Hao Yang, You-Zhi Jin, Zi-Yin Li, Deng-Bao Wang, Xin Geng, *Senior Member, IEEE*, and Min-Ling Zhang, *Senior Member, IEEE*

Abstract—Numerous researches have proved that deep neural networks (DNNs) can fit almost everything even given data with noisy labels, and result in poor generalization performance. However, recent studies suggest that DNNs tend to gradually memorize the data, moving from correct data to mislabeled data. Inspired by this finding, we propose a novel method named *Dynamic Loss Thresholding (DLT)*. During the training process, DLT records the loss value of each sample and calculates dynamic loss thresholds. Specifically, DLT compares the loss value of each sample with the current loss threshold. Samples with smaller losses can be considered as clean samples with higher probability and vice versa. Then, DLT discards the potentially corrupted labels and further leverages self-training semi-supervised learning techniques. Experiments on CIFAR-10/100, WebVision and Clothing1M demonstrate substantial improvements over recent state-of-the-art methods. In addition, we investigate two real-world problems. Firstly, we propose a novel approach to estimate the noise rates of datasets based on the loss difference between the early and late training stages of DNNs. Secondly, we explore the effect of hard samples (which are difficult to be distinguished) on the process of learning from noisy labels.

Index Terms—Deep Learning, learning from noisy labels, dynamic loss thresholding, semi-supervised learning.

1 Introduction

Although deep neural networks (DNNs) have achieved great success for image classification tasks [8], [16], [24], [41], their excellent performance mainly relies on large-scale datasets with clean label annotations. However, it is extremely expensive and time-consuming to label high-quality datasets, thus deep models are usually trained on data with lots of corrupted labels. As a result, dealing with label noise is a common adverse scenario which requires attention and has been extensively studied these years [1], [13], [15], [21], [27], [30], [48].

A recent study on the generalization capabilities of deep networks [60] demonstrates that DNNs can easily overfit to noisy labels and result in poor generalization performance. However, even though deep networks can fit everything in the end, they *learn patterns first* [3], and this suggests that DNNs gradually memorize the data, moving from correct data to mislabeled data. As shown in Figure 1(a), DNNs fit the correctly labeled samples (clean samples) before fitting noisy samples, resulting in notably larger loss values for noisy samples in the early training stage.

Existing methods for learning from noisy labels can be grouped into three main categories. The first one is based on label correction which aims to correct noisy labels to the ground-truth ones [29], [45], [46]. For example, the recent proposed *PENCIL* [58] utilizes back-propagation to correct image labels and update the network parameters simultaneously in an end-to-end manner. The second one is based on the robust loss function [32], [49], [62]. Ghoshet al. [10] proves that the loss functions which satisfy the symmetric condition, such as Mean Absolute Error (MAE), would be inherently tolerant to both uniform and class conditional label noise. The third one is based on sample selection which involves selecting correctly labeled samples from a noisy training dataset [1], [19], [27]. *Co-teaching* [15], [59] is a representative framework on this line which trains two networks where each network selects small-loss samples to teach another one. Most recently, sample selection based methods becomes perhaps the dominant paradigm for learning from noisy labels.

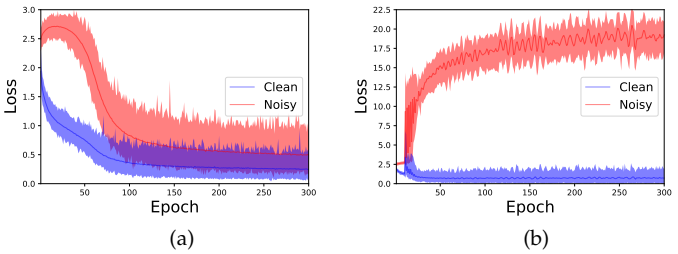


Fig. 1: Cross-entropy loss on CIFAR-10 under 50% symmetric label noise. (a) Training with cross-entropy loss results in fitting the noisy labels. (b) Using DLT avoids fitting label noise. The bold lines represent the means of losses and the shaded areas are the ranges of all samples.

In light of these recent advances, we propose *dynamic loss thresholding (DLT)* to avoid fitting noisy labels when training deep models, as shown in Figure 1(b). Specifically, DLT records the loss value of each sample during training and calculates loss thresholds dynamically. By comparing the loss value of each training sample with the current loss threshold, we expect to distinguish the potentially clean and

- *Hao Yang, You-Zhi Jin, Deng-Bao Wang, Xin Geng and Min-Ling Zhang are with the School of Computer Science and Engineering, Southeast University, Nanjing 210096, China, and the Key Laboratory of Computer Network and Information Integration (Southeast University), Ministry of Education, China. E-mail: {yang_h, jinyouzhi, wangdb, xgeng, zhangml}@seu.edu.cn. (Hao Yang and You-Zhi Jin contribute equally to this work.)*
- *Zi-Yin Li is with the Huawei Technologies. E-mail: liziyin94@gmail.com.*

noisy samples during training. Samples with smaller losses can be considered as correctly labeled samples and vice versa. Then, DLT discards these potentially corrupted labels and treats the corresponding samples as unlabeled data, thus we can leverage semi-supervised learning techniques like self-training strategies [55], [56] to improve the performance. Experiments on various benchmarks demonstrate substantial improvements over recent state-of-the-art methods. Furthermore, DLT is also of excellent generalization and flexibility and we empirically demonstrate that our proposed noisy label detection method via dynamic loss thresholding is still effective when combining with other methods.

In addition, the noise rates of datasets are always unknown but crucial to many real-world situations due to the fact that numbers of existing methods need the noise rate as their prior knowledge. In this paper, we further propose a novel method to estimate the noise rate based on the patterns of training loss values. In particular, our proposed method calculates the *loss difference* between the early and late deep network training stages and dynamically fits a two-component Gaussian Mixture Model (GMM) on per-sample loss difference to divide the training samples into a clean and noisy set. Accordingly, noise rate can be obtained by calculating the proportion of clean samples to total samples.

Besides, in real-world situations, there may exist a common class of samples which are always quite close to the decision boundary and hence difficult to be distinguished by DNNs. Intuitively, we call these samples *hard samples*. Hard samples can be defined as a subset of clean samples but easy to be mistaken for other classes. However, nearly no research discussed hard samples in the field of learning from noisy labels to our knowledge. By means of our proposed method, we investigate the effect of hard samples during deep network training for the first time. In summary, our main contributions are as follows:

- We propose a novel noisy label detection method DLT, which is based on *dynamic loss thresholds*. Combined with self-training semi-supervised learning techniques, our whole framework achieves state-of-the-art performance. We experimentally show that DLT is of excellent generalization and flexibility.
- We provide a method to estimate the noise rates of datasets based on *loss difference*. This is a key complement to eliminate the dependence on using noise rate as common prior knowledge.
- We provide insights into the effect of hard samples on the training of DNNs. We investigate this problem for not only clarifying the effectiveness of our method, but also finding out the patterns of hard samples when learning from noisy labels.

2 Related Work

2.1 Learning from Noisy Labels

Existing approaches for training DNNs from noisy labels can be roughly divided into four categories: 1) transition matrix based methods, 2) label correction methods, 3) loss correction methods, and 4) sample selection methods.

Transition matrix based methods. This kind of methods [7], [14], [18], [57] utilizes the underlying transition matrix to model label noise. Liu et al. [31] estimate the noise transition matrix within an importance reweighting framework and thereby build consistent classifiers. Patrini et al. [36] propose a loss correction method based on precalculated *Backward* or *Forward* noise transition matrix. Besides, Goldberger and Ben-Reuven [11] propose to model the noise transition and augment the correction architecture by adding an additional linear layer on top of the neural network. Xia et al. [53] propose an effective method to learn transition matrices from noisy data without employing anchor points, which represent clean samples with the highest probability of each class.

Label correction methods. This family of methods aims to relabel the corrupted labels. One research line tries to formulate explicit or implicit noise models to characterize the distribution of noisy and true labels using directed graphical models [54] , Conditional Random Fields [46] , knowledge graph [29], or neural networks [26], [47]. However, to recover the ground-truth labels, these approaches usually require the support from a small set of clean samples. Recently, Tanaka et al. [45] propose a method which relabels samples using network predictions by alternately updating network parameters and labels. *PENCIL* [58] updates both network parameters and label estimations in an end-to-end manner and does not need an auxiliary clean dataset. *Confident Learning (CL)* [34] estimate the joint distribution of label noise and then find out label errors.

Loss correction methods. Another line of learning from noisy labels seeks to modify the loss function to achieve robustness. Ghosh et al. [10] prove that for multi-class classification, the loss functions which satisfy the symmetric condition, such as Mean Absolute Error (MAE), would be inherently tolerant to both uniform and class conditional label noise. However, Zhang and Sabuncu [62] show that it is not able to achieve good performance by learning DNNs with MAE due to slow convergence caused by gradient saturation. Based on these findings, Zhang and Sabuncu [62] propose *Generalized Cross Entropy (GCE)*, which applies a negative Box-Cox transformation. *Symmetric cross entropy Learning (SL)* [49] combines Reverse Cross Entropy (RCE) (which satisfies the symmetric condition) together with the Cross Entropy loss. *Active Passive Loss (APL)* [32] combines two robust loss functions namely active loss and passive loss which mutually boost each other.

Sample selection methods. Besides the above three kinds of methods, numerous studies involve selecting potentially clean samples from a noisy training dataset. *Decouple* [33] trains the model using selected samples based on the discrepancy between the two classifiers. *MentorNet* [21] introduces a data-driven curriculum learning paradigm in which a pre-trained mentor network guides the training of a student network. *Co-teaching* [15] trains two DNNs simultaneously, and let them teach each other with some selected samples during every mini-batch. *O2U-Net* [19] selects samples with small losses and keeps the status of the learned network in transferring from overfitting to underfitting cyclically. Xia et al. [51] investigate the uncertainty of

losses during training and adopt confidence interval estimation to improve the robustness of models. *Me-Momentum* [4] leverages the idea of momentum and alternately refines the classifier to extract confident hard examples. *CDR* [50] divides all network parameters into the critical and non-critical ones, and then perform different update rules for them respectively. Besides, Arazo et al. [1] calculate sample weight by modeling training losses with a Beta Mixture Model (BMM). A recent study [27] proposed a method named *DivideMix* which trains two networks simultaneously and fits a Gaussian Mixture Model (GMM) on its per-sample loss distribution to divide the training samples into a labeled set and an unlabeled set. By leveraging the semi-supervised learning technique *MixMatch* [5], they achieved state-of-the-art performance on several benchmarks.

2.2 Semi-Supervised Learning

Semi-Supervised learning (SSL) methods are used to learn in the presence of both labeled and unlabeled data, which is naturally applicable to the problems of learning from noisy labels after discarding the labels of potentially mislabeled samples. Current SSL methods can be broadly divided into the following categories. **Consistency regularization** [25], [35], [38] which is based on the assumption that if a realistic perturbation was applied to the unlabeled data, the prediction should not be changed significantly. **Proxy-label methods** [20], [37] which leverage a pre-trained model on the labeled samples to produce additional training samples by labeling unlabeled samples. **Generative models** [9], [22] which model the real data distribution from the training dataset and then generate synthetic samples as augmentations. An emerging line of work is a set of holistic approaches that try to combine different dominant methods in SSL [5], [42]. Moreover, a learning principle, namely *Mixup* [61], is usually employed in those hybrid methods.

3 Methodology

We begin by introducing our dynamic loss threshold based method DLT. Besides, a novel approach is proposed to estimate the noise rates of datasets through loss difference. Lastly, we introduce two strategies to generate hard samples.

3.1 The Framework of DLT

Formally, let $D = (X, Y) = \{(x_i, y_i)\}_{i=1}^{N}$ denote the training data, where x_i is an image and $y_i \in \{0,1\}^C$ is the one-hot label over C classes. Let b denote the size of each mini-batch and $m = \lceil N/b \rceil$ denote the number of batches in each epoch. Then, the loss values of training samples in each mini-batch can be recorded as $L = \{l_1, l_2, ..., l_b\}$. The cross-entropy loss for sample x_i is:

$$l_i = \sum_{c=1}^{C} y_i^c \log(P_{model}^c(x_i)), \tag{1}$$

where P_{model}^c is the deep model's softmax output for class c. Let w be the noise rate and q be the selection proportion for training, which is set as:

$$q = (1 - w) \quad 100\%. \tag{2}$$

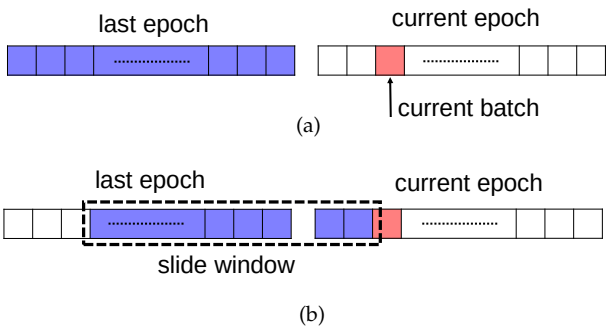


Fig. 2: (a) The last-epoch strategy. (b) The slide-window strategy. The purple windows represent the batches which are utilized to calculate dynamic loss thresholds.

Besides, $Quantile(L, q)$ represents the function to calculate the quantile that separates L. In practice, we utilize the NumPy.quantile() function to calculate this value.

In general, DLT can be divided into two steps: noisy label detection based on dynamic loss thresholds and then exploit both potentially clean and noisy samples along with SSL techniques.

3.1.1 Noise Detection by Dynamic Loss Thresholds

As we described above, DNNs usually fit clean samples before noisy ones, resulting in notably larger loss values for noisy samples in the early training stage. Therefore, a straight strategy is to identify noisy labels based on a specific loss threshold. Here we introduce two strategies to obtain dynamic loss thresholds.

The last-epoch strategy. An intuitive way is leveraging training samples of last epoch to calculate current loss thresholds, as shown in Figure 2(a). Notably, all samples in current epoch are compared with the same loss threshold when using this strategy. Specifically, when training at the t_{th} epoch, the loss values can be recorded as:

$$L^t = L_1^t \cup L_2^t, ..., L_m^t. \tag{3}$$

Based on selection proportion q, the loss threshold τ^t can be calculated as:

$$\tau^t = Quantile(L^{t-1}, q), \tag{4}$$

where the Quantile function returns the value such that at least q 100 percent of elements in L^{t-1} are smaller than this value.

The slide-window strategy. In order to make better use of past training samples to guide current training, DLT also utilizes the idea of slide windows. Let s denote the length of slide window. As shown in Figure 2(b), when training on current batch, loss thresholds can be calculated from the last s batches. We denote $L_{[-s:]}$ as the set that consists of the loss values of samples in the last s batches. Notably, this slide-window strategy reflects the learned model more timely. Based on selection proportion q, τ^t can be calculated as:

$$\tau^t = Quantile(L_{[-s:]}, q). \tag{5}$$

Then, for each sample x_i, DLT compares its loss value l_i with the loss threshold τ^t:

$$\begin{cases} Clean, & l_i \leq \tau^t, \\ Noisy, & l_i > \tau^t. \end{cases} \tag{6}$$

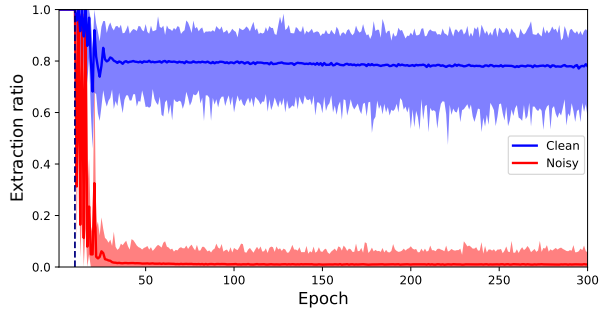


Fig. 3: The ratios of clean and noisy samples which are feed into DLT for training under 0.5 symmetric noise on CIFAR-10.

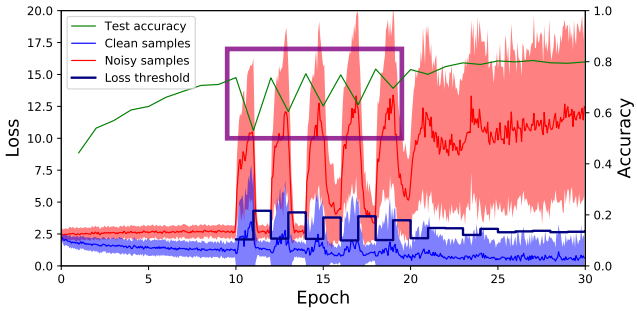


Fig. 4: Test accuracy fluctuates in the early stage of training (warm up for 10 epochs) under 0.5 symmetric noise on CIFAR-10.

Accordingly, training data X can be divided into two parts: X_{clean} and X_{noisy}, representing the potentially clean and noisy set respectively. Besides, we use an illustrative experiment to verify the effectiveness of DLT. As shown in Figure 3, most of the clean samples are retained during training, while few noisy samples are infiltrated in the training process.

3.1.2 Utilizing Semi-supervised Learning Techniques

After obtaining X_{clean} and X_{noisy}, self-training SSL techniques are leveraged in the following steps. Firstly, as is typical in many SSL methods, we use data augmentation [40], such as random clipping, translation and folding, on both clean and noisy data for K times. Then, clean and noisy samples are utilized respectively.

Utilize clean samples. For X_{clean}, *Mixup* [61] augmentation is leveraged to make DNNs favor linear behavior in-between training examples. It has been verified to effectively improve robustness of DNNs in many weakly supervised learning tasks [2], [5], [27]. In particular, *Mixup* randomly selects two samples (x_i, y_i) and (x_j, y_j) from total training data. Then, a mixed sample pair (x' and y') can be computed by:

$$\lambda \quad Beta(\alpha, \alpha), \tag{7}$$

$$\lambda' = max(\lambda, 1 \quad \lambda), \tag{8}$$

$$x' = \lambda' x_i + (1 \quad \lambda') x_j, \tag{9}$$

$$y' = \lambda' y_i + (1 \quad \lambda') y_j. \tag{10}$$

After leveraging *Mixup* to potentially clean samples, cross-entropy loss L_{clean} for X_{clean} is obtained according to Eq.(1), which denotes the first term in total loss of DLT.

Utilize noisy samples For X_{noisy}, its original corrupted labels will be discarded and our model $f_\theta(\,)$ will generate pseudo-labels Y_{pseudo} for them:

$$Y_{pseudo} = f_\theta(X_{noisy}). \tag{11}$$

Then a sharpening function is applied on Y_{pseudo}:

$$\hat{Y}^c_{pseudo} = {Y^c_{pseudo}}^{1/T} \Big/ \sum_{k=1}^{C} {Y^k_{pseudo}}^{1/T}, \quad c = 1, 2, ..., C, \tag{12}$$

where T denotes the sharpening temperature and we set hyperparameter $T = 0.5$.

Then, we also leverage *Mixup* to X_{noisy}. In order to utilize the potential information from noisy samples effectively, we randomly select a part of samples X_{clean} X_{clean} to *Mixup* with X_{noisy}. Mixed noisy samples are as follows:

$$\hat{X}_{noisy} = X_{clean} \; [\; X_{noisy}. \tag{13}$$

Notably, the samples in X_{clean} with smallest loss values, considered as clean samples with highest probability, are selected to *Mixup* with X_{noisy}. According to the work in [2], the mixing ratio of X_{clean} and X_{noisy} is:

$$|X_{clean}| : jX_{noisy}j = 16 : 100. \tag{14}$$

Moreover, mean squared error is employed for X_{noisy}. For each sample x_i in X_{noisy}, the mean squared error is calculated as follows:

$$l_i = \sum_{c=1}^{C} ky_i^c \quad P^c_{model}(x_i)k_2^2 . \tag{15}$$

L_{noisy} for X_{noisy} is obtained according to Eq.(15), which denotes the second term in total loss of DLT.

Lastly, to avoid assigning all samples to a single class, we apply the regularization term used by [1], [27], [45], which uses a uniform prior distribution ρ to regularize the model output in the mini-batch:

$$L_{reg} = \sum_{c=1}^{C} \rho_c \quad log\left(\rho_c \Big/ \frac{1}{jXj} \sum_{i=1}^{N} P^c_{model}(x_i)\right), \tag{16}$$

where

$$\rho_c = 1/C. \tag{17}$$

In summary, the loss L_{clean} on X_{clean} is the cross-entropy loss and the loss L_{noisy} on X_{noisy} is the mean squared error. Along with the regularization term L_{reg}, the total loss of DLT is:

$$L = L_{clean} + \lambda_n \quad L_{noisy} + \lambda_r \quad L_{reg} \tag{18}$$

In our experiments, we set λ_r as 1 and use λ_n to control the contribution of noisy samples.

Algorithm 1: The DLT approach.

Input: D : training set $\{(x_i, y_i)\}_{i=1}^{N}$;
m : the number of mini-batch in one epoch;
M : two modes to obtain loss thresholds;
K : the number of data augmentations;
T_{total} : the number of total training epochs;
T_{warm} : the number of warm-up epochs;

Output: θ : Model parameters .

for $t = 1 \ldots T_{total}$ **do**
  for $p = 1 \ldots m$ **do**
    Fetch mini-batch B from D;
    Calculate loss by $\boldsymbol{L} = L_{ce}(B, \theta)$ using eq. (1);
    if $t < T_{warm}$ **then** ▷ Warm up stage
      Update $\theta = SGD(\boldsymbol{L}, \theta)$;
    end
    else
      Obtain q using Eq. (19);
      if $M = LAST\ EPOCH$ **then** ▷ Last-epoch mode
        Obtain τ^t using Eq. (4);
      end
      else ▷ Slide-window mode
        Obtain τ^t using Eq. (5);
      end
      Obtain $\hat{B}$ =Augment(B, K);
      Obtain $B_{clean} = \{\hat{B}_i | l_i < \tau^t, l_i \in \boldsymbol{L}\}$;
      Obtain $B_{noisy} = \{\hat{B}_i | l_i > \tau^t, l_i \in \boldsymbol{L}\}$;
      Calculate $\hat{Y}_{pseudo}$ using Eq. (12);
      Calculate L_{clean} for X_{clean} using Eq. (1);
      Calculate L_{noisy} for X_{noisy} using Eq. (15);
      Calculate L_{reg} using Eq. (16);
      Calculate the total loss L using Eq. (18);
      Update $\theta = SGD(L, \theta)$;
    end
  end
end
return θ.

3.1.3 Leveraging Decreasing Loss Thresholds

For initial convergence of the algorithm, DLT needs to *warm up* the model in the beginning for a few epochs by training on all data using standard cross-entropy loss. As Figure 4 illustrates, the test accuracy of DLT always fluctuates in the early stage of training. The observed fluctuation is caused by that DLT will receive data with different distribution after the warm-up stage, which results in poor generalization performance. To solve this dilemma, DLT leverages a decreasing loss threshold strategy.

Specifically, the loss threshold is set to decrease gradually as training goes on. Formally, let T_{warm} denote the number of warm-up epochs and T_{grad} denote the number of loss threshold dynamically decreasing epochs. At the t_{th} epoch, we can get the selection proportion q as:

$$q = \begin{cases} 100\%, & t \in [1, T_{warm}], \\ 1 - \frac{t - T_{warm}}{T_{grad}}, & t \in (T_{warm}, T_{warm} + T_{grad}], \\ 1 - w, & otherwise. \end{cases} \tag{19}$$

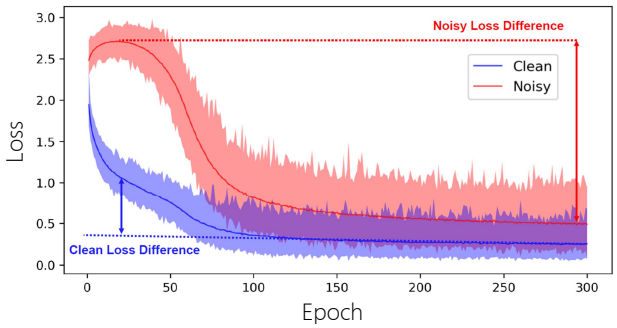


Fig. 5: Loss difference between early and late deep network training stages.

The proposed approach DLT is summarized in Algorithm 1. The warm-up stage is essential in the beginning phase, since the divergence of clean and noisy samples will highlight in this stage. Then, DLT leverages two strategies to calculate dynamically decreasing loss thresholds and divides total training samples into the potentially clean and noisy samples. Lastly, DLT generates pseudo-labels for noisy samples and leverages *Mixup* data augmentation and self-training SSL techniques to jointly utilize the clean and noisy samples.

3.2 Estimating Noise Rate by Loss Difference

Based on the discrepancy between loss values of clean and noisy samples, we propose a novel method to estimate the noise rates of datasets. As observed in Figure 5, loss values of clean and noisy samples are significantly discrepant in the early stage. In contrast, loss values of them are similar in the late stage of training. As a result, noisy samples can be detected by calculating the loss difference between the early and late stage. Intuitively, samples with larger loss differences can be considered as noisy samples.

Specifically, the loss difference values between the early and late stage for each sample can be obtained as $D = \{d_1, d_2, ..., d_n\}$. Then, we randomly initialize a GMM with two components using Expectation Maximization algorithm. Besides, the number of EM iterations to perform is set as 10. For each sample x_i, our model calculates the clean probability ϕ_i, which is the posterior probability $p(g|x_i)$ and g is the Gaussian component with smaller mean (smaller loss). Next, X_{clean} can be obtained as follows:

$$X_{clean} = \{x_i | \phi_i > 0.5\}. \tag{20}$$

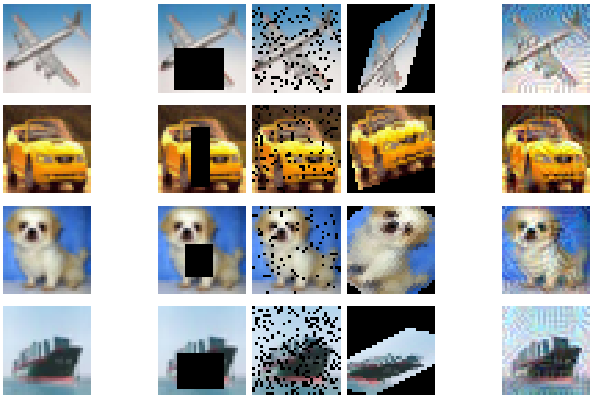
After separating clean samples from total samples according to loss difference, the noise rate r can be estimated as:

$$r = (1 - \frac{|X_{clean}|}{n}) \times 100\%. \tag{21}$$

3.3 Evaluating the Effect of Hard Samples

As mentioned above, we suppose hard samples are always quite close to the decision boundary thus difficult to be classified by DNNs, as shown in Figure 6. To further provide some insights into what role hard samples play during the

Fig. 6: Examples of hard samples (muffin or dog?).



(a) (b) (c)

Fig. 7: (a) Original samples. (b) Samples with incomplete information. (c) Samples suffered adversarial perturbation.

training process, we plan to conduct some experiments by involving some hard samples. However, it is very difficult to collect hard samples in real world. Therefore, to evaluate the effect, we propose to generate two types of hard samples by using some techniques refer to image data augmentation [40] from different aspects:

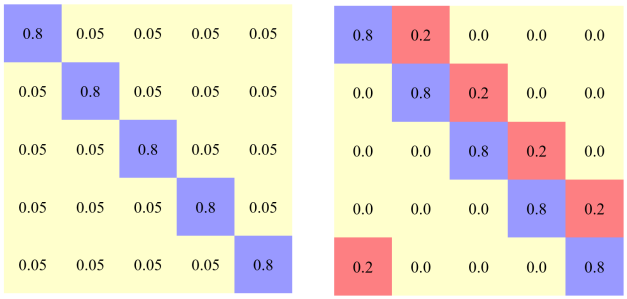
Samples with incomplete information. As shown in Figure 7(a), by random erasing, cropping, rotation, resizing and affine transformation, we can generate hard samples by removing some features from original samples.

Samples suffered adversarial perturbation. As Figure 7(b) illustrates, these samples are generated by adding some disturbances to origin samples. The adversarial examples of deep neural networks (DNNs) have attracted widespread attention recently [44]. We propose to add a small number of adversarial perturbations to original samples and consider them as hard samples. Practically, this perturbation strategy is attacking against the well-trained model according to FGSM method [12]. Formally, let ϵ be adversarial coefficient, $J(x, y)$ be the loss function. For each sample x_i, its generated adversarial sample x_i^{adv} can be described as:

$$x_i^{adv} = x_i + \epsilon \ \ sign(\nabla_x J(x_i, y_i)) \tag{22}$$

4 Experiments

We first conduct experiments in extensive settings and compare DLT to recent outstanding baselines. Besides, we empirically verify the proposed method for estimating noise rate by loss difference. Lastly, the effect of hard samples is also discussed in this section.



(a) symmetric noise (b) asymmetric noise

Fig. 8: Noise transition matrix of different noise types under 0.2 noise level with 5 classes.

4.1 Experimental Setup

4.1.1 Benchmark Datasets

To verify the superiority of our approach, we conduct experiments on two commonly used image classification datasets, namely CIFAR-10, CIFAR-100 [23]. Both of these two datasets contain 50,000 training and 10,000 test images, consisting of 32 32 color images arranged in 10 and 100 classes, respectively. In addition, we also conduct experiments on two large-scale real-world datasets, WebVision [28] and Clothing1M [54]. WebVision contains noisy-labeled images collected from Flickr and Google and maintains the same 1,000 classes as the ImageNet ILSVRC 2012 dataset [39]. Similar to previous work [27], we perform experiments on the first 50 classes from the Google image subset of WebVision, which contains approximately 66 thousand images. The noise level of WebVision is estimated at 20% [28]. Clothing1M contains 14 classes with 1 million noisy training samples collected from online shopping websites, e.g. t-shirt, sweater and jacket. The labels are generated by the surrounding text of images and thus extremely noisy. Its overall noise rate is approximately 38.5% [30]. This dataset is seriously imbalanced and the label noise mostly happen between similar classes.

4.1.2 Noise Setting

Following previous works [1], [27], [49], we generate two types of label noise: symmetric (uniform) noise and asymmetric (class-conditional) noise. We propose to corrupt these clean datasets manually by the noise transition matrix Q, where $Q_{ij} = P(\tilde{y} = j | y = i)$ given that noisy label $\tilde{y}$ is flipped from its corresponding clean label y. As shown in Figure 8(a), symmetric noisy labels are generated by randomly replacing the labels for a percentage of training data with all possible labels (i.e. the true label could be randomly maintained). While asymmetric noisy labels are only replaced by a specific set of classes, as shown in Figure 8(b). For example, for CIFAR-10, flipping BIRD ! AIRPLANE, CAT $ DOG, TRUCK ! AUTOMOBILE, DEER ! HORSE.

4.1.3 Compared Methods

We compare DLT with multiple recent excellent approaches for learning from noisy labels:

Standard CE, which is the most fundamental baseline. The model is directly trained on the origi-

TABLE 1: The value of loss weight λ_u under different noise types and noise rates.

Hyperparameter	Symmetric Noise				Asymmetric Noise		
	0.2	0.4	0.6	0.8	0.2	0.3	0.4
λ_u (CIFAR-10)	25	25	25	50	0	0	0
λ_u (CIFAR-100)	25	150	150	150	150	150	150

TABLE 2: Comparison with state-of-the-art methods in test accuracy (%) on CIFAR-10 with different noise types and levels. We re-implement all methods under the same setting and the best results are boldfaced.

Noise Type		Symmetric Noise				Asymmetric Noise		
Methods / Noise Rate		0.2	0.4	0.6	0.8	0.2	0.3	0.4
Standard CE	Best	88.1	84.3	77.7	63.2	90.6	89.4	85.0
	Last	81.7	65.0	44.9	26.5	87.8	83.4	72.3
Forward [36]	Best	88.5	84.9	78.6	64.0	89.3	88.7	84.3
	Last	82.3	69.2	50.8	27.5	86.7	82.6	71.9
GCE [62]	Best	91.2	88.9	82.3	70.6	90.2	88.7	84.8
	Last	89.3	83.7	70.4	56.3	87.9	82.3	75.1
SL [49]	Best	92.9	90.9	87.0	76.8	92.3	91.2	87.9
	Last	90.0	81.7	68.7	59.7	90.1	85.5	79.9
PENCIL [58]	Best	92.4	91.2	87.4	77.5	92.2	90.7	89.3
	Last	92.0	90.5	86.5	76.5	92.1	90.5	88.5
APL [32]	Best	89.8	85.4	77.8	69.3	87.2	85.7	83.6
	Last	89.3	84.0	76.4	67.5	87.0	84.6	82.2
Me-Momentum [4]	Best	91.5	88.4	81.6	30.0	92.2	90.7	84.2
	Last	—	—	—	—	—	—	—
CDR [50]	Best	89.8	85.1	76.5	39.6	91.3	89.9	82.4
	Last	81.7	63.8	38.7	16.6	79.7	69.8	57.0
CNLCU-S [51]	Best	83.2	78.3	67.3	28.2	82.9	79.0	74.2
	Last	82.9	78.0	67.0	27.0	82.5	78.5	71.5
M-correction [1]	Best	94.0	93.3	90.1	86.8	—	—	—
	Last	93.6	93.1	89.7	86.6	—	—	—
DivideMix [27]	Best	95.6	93.9	93.7	92.4	94.2	93.3	92.7
	Last	95.2	93.6	93.0	92.1	93.4	92.5	91.5
DLT (last-epoch)	Best	95.7	95.6	**94.8**	92.5	**94.4**	93.5	92.5
	Last	95.4	94.8	**94.5**	91.7	**93.9**	93.0	91.6
DLT (slide-window)	Best	**96.3**	**95.7**	94.7	**93.0**	94.3	**93.8**	**92.8**
	Last	**95.8**	**95.4**	94.3	**92.7**	93.9	**93.1**	**92.2**

nal dataset with noisy labels using standard cross-entropy loss.

Forward [36], a loss correction method by multiplying the network prediction with the ground truth noise matrix.

GCE [62], which proposes a theoretically noise-robust loss function that can be seen as a generalization of MAE and CE.

SL [49], which combines the cross-entropy loss with a noise robust counterpart named Reverse Cross Entropy (RCE).

PENCIL [58], which jointly optimize both network parameters and label estimations in an end-to-end manner.

APL [32], which combines two robust loss functions namely active loss and passive loss that mutually boost each other.

CNLCU-S [51], which investigates the uncertainty of losses during training and adopt confidence interval estimation.

CDR [50], which divides all network parameters into the critical and non-critical ones, and then perform different update rules.

Me-Momentum [4], which leverages the idea of momentum and alternately refines the network parameters.

M-correction [1], which calculates sample weight by modeling training losses with BMM. This method is specifically designed for symmetric noise, and thus we only report its results under symmetric noise setting.

DivideMix [27], which is currently the state-of-the-art method for learning from noisy labels and leverages MixMatch [5].

Notably, the last two methods M-correction and DivideMix

TABLE 3: Comparison with state-of-the-art methods in test accuracy (%) on CIFAR-100 with different noise types and levels. We re-implement all methods under the same setting and the best results are boldfaced.

Noise Type		Symmetric Noise				Asymmetric Noise		
Methods / Noise Rate		0.2	0.4	0.6	0.8	0.2	0.3	0.4
Standard CE	Best	61.4	53.2	42.0	20.7	63.3	56.0	46.2
	Last	57.5	41.2	24.4	9.2	59.6	51.8	42.6
Forward [36]	Best	61.0	53.4	42.8	21.1	62.4	55.8	45.9
	Last	57.2	42.0	25.3	9.3	58.9	51.5	42.2
GCE [62]	Best	70.1	65.0	54.7	30.4	69.0	64.7	50.7
	Last	62.4	56.2	40.9	16.2	61.5	55.8	42.3
SL [49]	Best	60.1	53.7	42.8	21.5	60.2	55.6	45.8
	Last	56.8	41.3	24.8	9.8	58.2	50.3	40.8
PENCIL [58]	Best	73.2	68.1	57.8	30.7	74.0	71.8	62.7
	Last	72.7	66.7	55.9	20.3	72.6	70.5	60.3
APL [32]	Best	71.2	66.7	54.0	31.2	70.1	66.4	55.5
	Last	66.6	58.3	44.5	20.6	60.0	49.9	44.1
Me-Momentum [4]	Best	69.4	63.3	51.2	19.8	67.8	61.5	49.9
	Last	—	—	—	—	—	—	—
CDR [50]	Best	64.8	56.0	39.0	19.9	66.7	60.1	49.0
	Last	61.1	45.8	24.7	6.1	61.9	54.0	43.4
CNLCU-S [51]	Best	50.2	42.8	28.5	11.7	46.3	40.0	32.3
	Last	49.7	42.2	28.1	11.3	46.0	39.4	31.7
M-correction [1]	Best	73.9	71.8	59.7	48.2	—	—	—
	Last	73.4	71.1	50.7	47.6	—	—	—
DivideMix [27]	Best	75.7	73.9	69.6	53.5	75.4	72.7	59.7
	Last	74.9	73.0	69.1	52.3	74.9	72.0	51.2
DLT (last-epoch)	Best	76.6	74.8	69.7	**59.3**	**76.5**	72.4	66.7
	Last	75.5	73.8	69.0	**58.5**	**75.1**	71.3	65.5
DLT (slide-window)	Best	**77.1**	**75.2**	**70.1**	58.9	76.1	**72.8**	**68.0**
	Last	**75.9**	**74.0**	**69.2**	58.0	74.7	**72.1**	**66.7**

apply *Mixup* [61] augmentation.

4.1.4 Evaluation Metrics

Following previous work [21], [27], [58], test accuracy on clean testing samples is employed to evaluate the performance of DLT and other state-of-the-art baselines in this paper. For a classifier $h(\ )$, test accuracy is calculated as follows:

$$accuracy(h) = \frac{1}{N}\sum_{i=1}^{N}[\![h(x_i) = y_i]\!]. \tag{23}$$

Here, $[\![\pi]\!]$ returns 1 if predicate π holds and 0 otherwise. The test accuracy evaluates the fraction of correctly classified examples, i.e. the predicted label is identical to the ground-truth label.

4.1.5 Implementation Details

The implementation is based on PyTorch and experiments were carried out with NVIDIA Tesla V100 GPU. For hyper-parameters of DLT, we set temperature parameter $T = 0.5$, data augmentation times $K = 2$, the number of loss threshold dynamically decreasing epochs $T_{grad} = 10$ and table 1 shows the value of loss weight λ_u 2 $f0, 25, 50, 150g$ which we use under different noise types and noise rates.

For CIFAR-10 and CIFAR-100, we use an 18-layer PreAct Resnet [17] and train it using SGD with a momentum of 0.9, a weight decay of 0.0005, and a batch size of 128. The network is trained for 300 epochs. The warm up period is 10 epochs for CIFAR-10 and 30 epochs for CIFAR-100. We set the initial learning rate as 0.02 and reduce it by a factor of 10 after 150 epochs and set the slide window size s as 200. In addition, we set *Mixup* hyperparameter α as 4. We re-implement all the comparison methods using the same network architecture (PreAct-ResNet-18) with our proposed method and the parameters are tuned according to their respective literatures. Note that DivideMix [27] trains two networks simultaneously and uses the ensemble results in inference phase. For a fair comparison, we follow previous work [48] and only use the prediction from a single network. And we show that we can improve DLT by using the same averaging strategy at the inference phase in the following generalization experiments.

For WebVision and Clothing1M, following previous work [27], we use Inception-ResNet-V2 [43] and ResNet-50 [16] with ImageNet pretrained weights respectively. The SGD optimizer is utilized with a momentum of 0.9 and a weight decay of 0.0005. The slide window size s is set as 500 and the value of loss weight λ_u is set as 0. In addition, we set *Mixup* hyperparameter α as 0.5. For WebVision, the network is trained for 100 epochs and the warm up period is 1 epoch. The initial learning rate is set as 0.01 and reduced by a factor of 10 after 50 epochs. For Clothing1M, the network is trained for 80 epochs and the warm up period is 1 epoch. The initial learning rate is set as 0.002 and reduced by a factor of 10 after 40 epochs. Following previous work [27], for each epoch, we sample 1000 mini-batches from the training data

TABLE 4: Comparison with state-of-the-art methods in test accuracy (%) on WebVision and ILSVRC2012 validation sets.

Methods	WebVision		ILSVRC12	
	top1	top5	top1	top5
Standard CE	73.72	90.92	69.60	90.24
Forward [36]	61.12	82.68	57.36	82.36
GCE [62]	55.56	66.52	54.52	67.24
SL [49]	75.80	91.68	72.20	91.24
APL [32]	72.32	88.64	69.08	88.04
DivideMix [27]	77.32	91.64	**75.20**	90.84
DLT (last-epoch)	77.60	92.12	74.44	91.64
DLT (slide-window)	**78.36**	**92.76**	74.68	**92.36**

TABLE 5: Comparison with state-of-the-art methods in test accuracy (%) on Clothing1M.

Methods	Test Accuracy
Standard CE	69.21
Forward [36]	69.84
GCE [62]	69.75
SL [49]	71.02
PENCIL [58]	73.49
M-correction [1]	71.00
DivideMix [27]	**74.76**
DLT (last-epoch)	74.24
DLT (slide-window)	73.98

while ensuring the labels are balanced.

Notably, the selection proportion q is intuitively set as $1 - w$ both for CIFAR-10 and CIFAR-100. For WebVision and Clothing1M, q needs to be tuned, and we set it as 0.85 and 0.75 respectively in our paper.

4.2 Experimental Results

4.2.1 Experiments on CIFAR-10/100

We compare the proposed DLT with multiple baseline methods with symmetric label noise (noise rate $r = 0.2, 0.4, 0.6, 0.8$) and asymmetric label noise (noise rate $r = 0.2, 0.3, 0.4$). Besides, for asymmetric noise, the noise rate $> 50\%$ means over half of the training data have wrong labels which is theoretically indistinguishable without additional assumptions. Table 2 and 3 present the results on benchmark datasets CIFAR-10 and CIFAR-100 with different noise types and levels. Both the best test accuracy across all epochs and the averaged test accuracy over the last 10 epochs are reported for each case.

The results indicate that DLT with slide-window strategy outperforms existing state-of-the-art methods across all noise rates with different noisy types, especially on CIFAR-10 under 0.4 and 0.6 symmetric noise, and is substantial (about 6% in accuracy) for the more challenging CIFAR-100 under 0.8 symmetric noise. Besides, merely on CIFAR-10 under 0.4 asymmetric noise and CIFAR-100 under 0.3 asymmetric noise, DLT with the last-epoch strategy falls short of the state-of-the-art method DivideMix. Note that the performance of slide-window strategy is slightly superior to the last-epoch strategy in most cases. We postulate this is because the loss values of the latest batches which in the slide window can reflect more real-time information from DNNs, thus guide the training process of DLT more effectively.

4.2.2 Experiments on Real-world Datasets

We have seen that DLT achieves excellent performance on datasets with manually corrupted noisy labels. Next, we conduct another experiment on two real-world large-scale noisy datasets WebVision and Clothing1M. Experimental results are reported in Table 4 and 5. For WebVision, we re-implement GCE, SL, APL under our experimental settings, except for their exclusive parameters which are followed in literature [32]. Results of Forward and DivideMix are copied from literature [27]. For Clothing1M, results compared methods are copied from their corresponding papers.

As Table 4 shows, we test the performance of DLT on WebVision and measure top1 and top5 test accuracy on WebVision validation set and ImageNet ILSVRC12 validation set respectively. DLT consistently outperforms compared methods in most cases, which indicates the effectiveness of our proposed approach under real-world label noise.

As Table 5 illustrates, DLT significantly improves the classification performance in most cases on Clothing1M. However, the test accuracy of DLT is slightly lower than that of DivideMix [27] about 0.5%. We think this limitation is caused by the instance-dependent noisy labels [6], [52], [63] in Clothing1M dataset and our proposed method is not robust to this type of noise. Furthermore, compared to DLT, the result of DivideMix is based on the ensemble of predictions from two networks.

4.2.3 Generality of the Proposed Approach

To demonstrate that DLT has an advantage in generalization and flexibility, we apply it to DivideMix [27]. Note that we use the same network, hyperparameters (except we fix the label co-refinement coefficient w_b as 0.5), and learning rate policy as DivideMix. Experiments are conducted under symmetric noise with noise rate $w \in \{0.2, 0.5, 0.8\}$. DLT* denotes our proposed noisy label detection method via dynamic loss thresholding with the slide-window strategy. As shown in Table 6, our noisy label detection method consistently outperforms DivideMix, even when the noise rate is extremely high. Note that combined with the schemes reported in DivideMix, DLT achieves great improvement on CIFAR-100 under 0.2 and 0.5 symmetric noise. This indicates that DLT is of excellent generality and can still be effective when applied to other methods.

4.2.4 Parameter Sensitivity

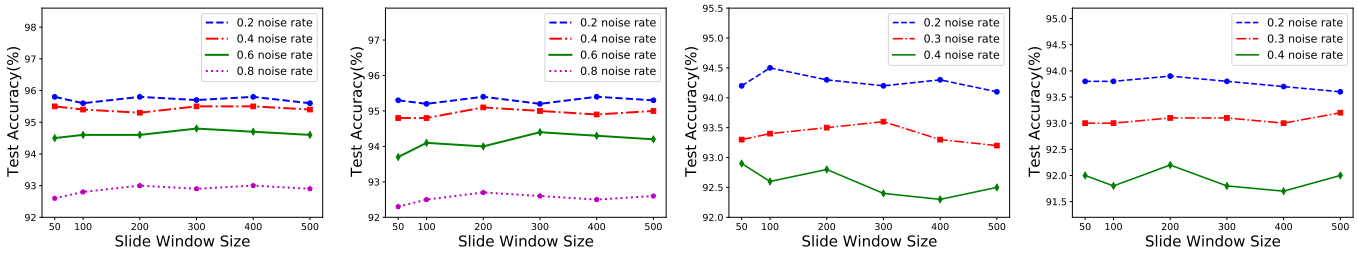
As mentioned above, DLT with the slide-window strategy exceeds all the other baselines in the majority of cases with both symmetric noise and asymmetric noise. As shown in Algorithm 1, DLT has one parameter to be tuned, i.e. s (the size of slide window). In this subsection, we further investigate how the performance of DLT changes with different values of s. Figure 9 and Figure 10 list results

TABLE 6: Comparison with DivideMix on CIFAR-10 and CIFAR-100 with different noise rates under symmetric label noise.

Datasets		CIFAR-10			CIFAR-100		
Methods / Noise Rate		0.2	0.5	0.8	0.2	0.5	0.8
DivideMix [27]	Best	96.1	94.6	93.2	77.3	74.6	60.2
	Last	95.7	94.4	92.9	76.9	74.2	59.6
DLT*	Best	**96.5**	**95.5**	**93.5**	**80.0**	**75.9**	**61.0**
	Last	**96.1**	**95.2**	**93.1**	**79.2**	**75.2**	**60.4**

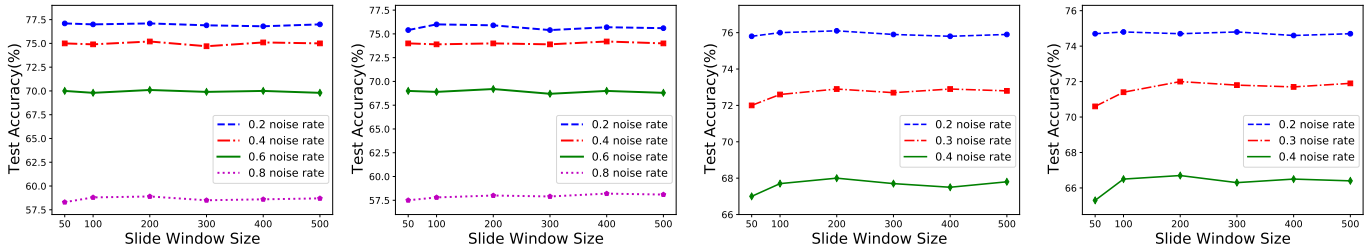
TABLE 7: Comparison with different K values on CIFAR-10 under symmetric label noise with noise rate = 0.5.

Datasets	CIFAR-10				CIFAR-100			
K ratio	8	16	32	64	8	16	32	64
Best	95.20	**95.24**	95.23	95.13	72.08	**72.74**	72.34	72.35
Last	94.26	**94.86**	94.74	93.76	71.30	**71.50**	71.35	71.11



(a) *Best* under **Sym.** (b) *Last* under **Sym.** (c) *Best* under **Asym.** (d) *Last* under **Asym.**

Fig. 9: The performance of DLT changes as the value of slide window size s under different noise types on CIFAR-10.



(a) *Best* under **Sym.** (b) *Last* under **Sym.** (c) *Best* under **Asym.** (d) *Last* under **Asym.**

Fig. 10: The performance of DLT changes as the value of slide window size s under different noise types on CIFAR-100.

of DLT with different window sizes for CIFAR-10 and CIFAR-100. In Figure 9 and Figure 10, "*Best*" denotes the best test accuracy across all epochs and "*Last*" denotes the averaged test accuracy over the last 10 epochs, s ranges in $\{50, 100, 200, 300, 400, 500\}$. It is shown that, the performance of DLT indeed fluctuates as the value of s varies, especially for CIFAR-10 and CIFAR-100 under 0.6 noise rate. Specifically, for CIFAR-100 under asymmetric noise, small s would lead to performance degradation of DLT. In this paper, we set s to 200, which denotes that samples in the latest 200 batches are utilized to calculate dynamic loss thresholds in our experiments. In addition, as shown in Table 7 and Table 8, Our proposed method DLT yields stable performance under different mixing ratios K and loss threshold strategies. Moreover, we find that 16% is a good choice for DLT compared with other ratios, which is consistent with the results in [2]. Besides, the proposed decreasing loss threshold strategy also outperforms these baselines on these settings.

4.2.5 Ablation Study

To analyze the effect of each component in DLT, we conduct an ablation study on CIFAR-10 and CIFAR-100 under different noise types and noise rates using the slide-window strategy, and the results are shown in Table 9 and 10. Notably, when removing *Mixup* for X_{clean}, the test accuracy decreases sharply, especially on CIFAR-10 under asymmetric noise and the results indicate that *Mixup* is essential for X_{clean}. For X_{noisy}, *Mixup* with X_{clean} (samples in X_{clean} with smallest loss values) yields better performance, especially on CIFAR-100. Besides, removing the gradually decreasing loss thresholds results in a decrease in test accuracy,

TABLE 8: Test accuracy (%) on CIFAR-10 and CIFAR-100 with different loss threshold strategies.

Dataset		CIFAR-10							CIFAR-100						
Noise Type		Symmetric				Asymmetric			Symmetric				Asymmetric		
Noise Rate		0.2	0.4	0.6	0.8	0.2	0.3	0.4	0.2	0.4	0.6	0.8	0.2	0.3	0.4
DLT	Best	**96.3**	**95.6**	**94.6**	**93.0**	94.3	**93.5**	**92.8**	**77.1**	**75.2**	**70.1**	**58.9**	**76.1**	**72.8**	**68.0**
w/ decreasing q	Last	95.8	**95.3**	**93.7**	**92.7**	93.9	**93.1**	**92.2**	**75.9**	**74.0**	**69.2**	**58.0**	**74.7**	**72.1**	**66.7**
w/ increasing q	Best	**96.3**	95.4	94.3	91.2	**94.4**	93.5	92.3	75.7	73.5	67.3	**58.9**	75.7	70.8	65.0
	Last	**95.9**	94.9	92.3	90.9	**94.0**	93.0	91.6	74.3	71.8	66.0	57.8	74.3	69.1	63.7
w/ fixed q	Best	95.6	95.4	94.4	92.7	94.1	93.2	92.2	76.7	74.4	69.3	57.0	75.4	71.8	66.5
	Last	95.1	95.0	93.7	92.2	93.6	92.8	91.9	75.1	73.5	68.8	56.2	74.3	70.6	64.4

TABLE 9: Ablation study results in test accuracy (%) on CIFAR-10 with different noise types and levels.

Noise Type		Symmetric Noise				Asymmetric Noise		
Methods / Noise Rate		0.2	0.4	0.6	0.8	0.2	0.3	0.4
DLT	Best	**96.3**	**95.6**	**94.6**	**93.0**	**94.3**	**93.5**	**92.8**
	Last	**95.8**	**95.3**	**94.0**	**92.7**	**93.9**	**93.1**	**92.2**
DLT w/o sharpening	Best	95.6	94.9	94.2	92.4	94.2	93.4	92.3
	Last	95.2	93.9	93.7	92.0	93.8	92.9	91.9
DLT w/o augmentation	Best	95.2	95.1	93.3	92.0	93.4	92.5	91.3
	Last	94.8	94.6	93.0	91.5	93.0	92.2	90.8
DLT w/o decreasing q	Best	95.6	95.2	94.2	92.7	94.1	93.2	92.2
	Last	95.1	95.0	93.7	92.2	93.6	92.8	91.9
DLT w/o *Mixup* for $\mathcal{X}_{clean}$	Best	95.1	94.8	93.7	91.2	91.5	88.6	86.5
	Last	94.9	93.9	93.7	90.1	88.3	87.3	81.8
DLT w/o *Mixup* $\bar{\mathcal{X}}_{clean}$ for $\mathcal{X}_{noisy}$	Best	95.6	95.2	94.1	92.3	94.2	93.4	92.4
	Last	95.0	94.8	93.7	91.6	93.8	93.0	92.1

TABLE 10: Ablation study results in test accuracy (%) on CIFAR-100 with different noise types and levels.

Noise Type		Symmetric Noise				Asymmetric Noise		
Methods / Noise Rate		0.2	0.4	0.6	0.8	0.2	0.3	0.4
DLT	Best	**77.1**	**75.2**	**70.1**	**58.9**	**76.1**	**72.8**	**68.0**
	Last	**75.9**	**74.0**	**69.2**	**58.0**	**74.7**	**72.1**	**66.7**
DLT w/o sharpening	Best	76.3	73.5	67.5	54.5	74.2	70.0	64.3
	Last	75.3	72.3	66.1	53.0	73.2	68.5	62.8
DLT w/o augmentation	Best	75.5	73.3	68.6	55.3	72.9	71.2	64.0
	Last	74.5	72.5	67.6	54.1	71.6	70.0	62.7
DLT w/o decreasing q	Best	76.7	74.4	69.3	57.0	75.4	71.8	66.5
	Last	75.1	73.5	68.8	56.2	74.3	70.6	64.4
DLT w/o *Mixup* for $\mathcal{X}_{clean}$	Best	73.4	71.9	62.0	46.3	73.1	70.0	63.5
	Last	73.0	70.5	60.5	44.8	72.3	69.5	63.0
DLT w/o *Mixup* $\bar{\mathcal{X}}_{clean}$ for $\mathcal{X}_{noisy}$	Best	75.9	74.3	69.3	57.0	75.6	72.3	66.9
	Last	74.7	73.3	68.2	55.8	74.1	71.1	65.9

which suggests that our decreasing loss threshold strategy is effective for DLT. Lastly, we find that both temperature sharpening and input augmentation are also beneficial for DLT.

4.2.6 Experiments on Noise Rate Estimating

As we described in Section 3.2, the noise rate of the original dataset can be estimated by our proposed approach based on loss difference between the early (underfitting) and late (overfitting) training stages. Specifically, for CIFAR-10 and CIFAR-100, we simply choose the 30_{th} epoch as the early training stage and the 300_{th} (last) epoch as the late training stage respectively. Moreover, we obtain the loss difference value of each sample from these two selected epochs. Then we estimate the noise rate using Eq. (21). As shown in Table 11, we can get highly accurate estimations on CIFAR-10 and CIFAR-100 across the noise rate ranging from 0.2 to 0.6 under symmetric noise, which verifies the effectiveness of our proposed method.

4.2.7 Discussion about Hard Samples

We experimentally investigate the effect of hard samples in the DNN training from noisy labels. In order to simulate the real-world situations, we generate hard samples according to the data augmentation strategies introduced in Section

TABLE 11: Noise rate (%) estimation of our proposed method on CIFAR-10 and CIFAR-100.

	CIFAR-10					CIFAR-100				
Real Noise Rate	20	30	40	50	60	20	30	40	50	60
Estimated Noise Rate	21.6	31.8	40.9	49.5	59.1	25.3	31.5	39.8	47.8	56.6

TABLE 12: Test accuracy (%) on CIFAR-10 and CIFAR-100 while adding synthetic hard samples or not.

Datasets	CIFAR-10								CIFAR-100							
Noise Rate	0.2		0.4		0.6		0.8		0.2		0.4		0.6		0.8	
Whether adding hard samples?	Best	Last	Best	Last	Best	Last	Best	Last	Best	Last	Best	Last	Best	Last	Best	Last
No.	96.3	95.8	95.7	95.4	94.7	94.3	93.0	92.7	77.1	75.9	75.2	74.0	70.1	69.2	58.9	58.0
Yes.	95.8	95.3	95.4	94.8	94.0	93.6	92.1	91.6	75.8	74.6	72.5	71.8	66.9	66.5	55.9	54.9

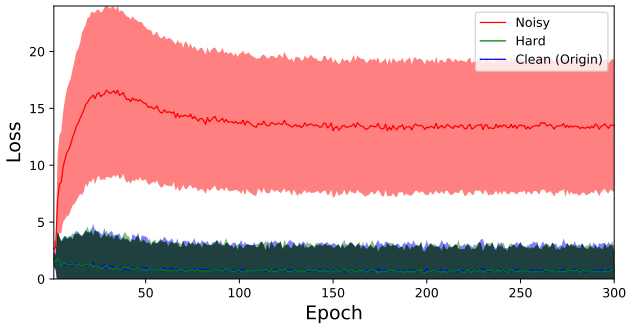
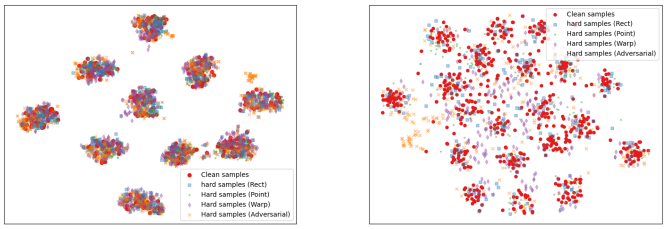


Fig. 11: Loss tendencies of noisy, hard and original clean samples under 0.4 symmetric noise on CIFAR-10.



(a) 300th epoch on CIFAR-10 (b) 300th epoch on CIFAR-100

Fig. 12: The t-SNE visualization of feature representations of CIFAR-10 and CIFAR-100 (20 classes among all 100 classes) trained by DLT.

3.3. In our experiments, we manually add the same proportion of hard samples as the original clean ones into datasets.

As shown in Figure 11, the loss value changing tendency of hard samples behaves similarly to that of original clean samples, but has significant differences from that of noisy samples. According to the diverse loss values, our method can divide the hard samples into the clean set rather than the noisy set. Table 12 also indicates the consistent results with Figure 11. When adding synthetic hard samples into training data, the test accuracy is solely a little inferior to that of the original symmetric label noise. Besides, Figure 12 illustrates that DLT regards hard samples as regular (easy and clean) samples through the t-SNE visualization. This observation shows that DLT can still work robustly while existing some indistinct samples, too.

5 Conclusion

In this paper, we propose a simple but effective method DLT, which is based on dynamic loss thresholds for learning from noisy labels. DLT takes advantage of different loss value distributions of clean and noisy samples and leverages self-training semi-supervised learning techniques to exploit the mislabeled data. Along with gradually decreasing loss thresholds and the slide-window strategy, DLT achieves state-of-the-art performance both on synthetic and real-world datasets. Besides, we propose a noise rate estimation method based on loss difference and achieve considerable results both on CIFAR-10 and CIFAR-100. In the end, we experimentally verify the effectiveness of our noisy label detection method while ensuring that some hard samples are contained in training data.

References

[1] E. Arazo, D. Ortego, P. Albert, N. E. O'Connor, and K. McGuinness, "Unsupervised label noise modeling and loss correction," in *Proceedings of International Conference on Machine Learning*, 2019, pp. 312–321.

[2] E. Arazo, D. Ortego, P. Albert, N. E. O'Connor, and K. McGuinness, "Pseudo-labeling and confirmation bias in deep semi-supervised learning," in *Proceedings of International Joint Conference on Neural Networks*, 2020, pp. 1–8.

[3] D. Arpit, S. Jastrzębski, N. Ballas, D. Krueger, E. Bengio, M. S. Kanwal, T. Maharaj, A. Fischer, A. Courville, Y. Bengio *et al.*, "A closer look at memorization in deep networks," in *Proceedings of International Conference on Machine Learning*, 2017, pp. 233–242.

[4] Y. Bai and T. Liu, "Me-momentum: Extracting hard confident examples from noisily labeled data," in *Proceedings of the IEEE International Conference on Computer Vision*, 2021, pp. 9312–9321.

[5] D. Berthelot, N. Carlini, I. Goodfellow, N. Papernot, A. Oliver, and C. Raffel, "Mixmatch: A holistic approach to semi-supervised learning," in *Proceedings of Advances in Neural Information Processing Systems*, 2019, pp. 5050–5060.

[6] H. Cheng, Z. Zhu, X. Li, Y. Gong, X. Sun, and Y. Liu, "Learning with instance-dependent label noise: A sample sieve approach," in *Proceedings of International Conference on Learning Representations*, 2021.

[7] L. Cheng, X. Zhou, L. Zhao, D. Li, H. Shang, Y. Zheng, P. Pan, and Y. Xu, "Weakly supervised learning with side information for noisy labeled images," in *Proceedings of European Conference on Computer Vision*, 2020, pp. 306–321.

[8] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255.

[9] M. Ehsan Abbasnejad, A. Dick, and A. van den Hengel, "Infinite variational autoencoder for semi-supervised learning," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5888–5897.

[10] A. Ghosh, H. Kumar, and P. Sastry, "Robust loss functions under label noise for deep neural networks," in *Proceedings of AAAI Conference on Artificial Intelligence*, 2017, pp. 1919–1925.

[11] J. Goldberger and E. Ben-Reuven, "Training deep neural-networks using a noise adaptation layer," in *Proceedings of International Conference on Learning Representations*, 2017.

[12] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *Proceedings of International Conference on Learning Representations*, 2015.

[13] B. Han, G. Niu, X. Yu, Q. Yao, M. Xu, I. Tsang, and M. Sugiyama, "Sigua: Forgetting may make learning with noisy labels more robust," in *Proceedings of International Conference on Machine Learning*. PMLR, 2020, pp. 4006–4016.

[14] B. Han, J. Yao, G. Niu, M. Zhou, I. Tsang, Y. Zhang, and M. Sugiyama, "Masking: A new perspective of noisy supervision," *Proceedings of Advances in Neural Information Processing Systems*, vol. 31, 2018.

[15] 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 *Proceedings of Advances in Neural Information Processing Systems*, 2018, pp. 8536–8546.

[16] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[17] ——, "Identity mappings in deep residual networks," in *Proceedings of European Conference on Computer Vision*, 2016, pp. 630–645.

[18] D. Hendrycks, M. Mazeika, D. Wilson, and K. Gimpel, "Using trusted data to train deep networks on labels corrupted by severe noise," *Proceedings of Advances in Neural Information Processing Systems*, vol. 31, 2018.

[19] J. Huang, L. Qu, R. Jia, and B. Zhao, "O2U-Net: A simple noisy label detection approach for deep neural networks," in *Proceedings of IEEE International Conference on Computer Vision*, 2019, pp. 3326–3334.

[20] A. Iscen, G. Tolias, Y. Avrithis, and O. Chum, "Label propagation for deep semi-supervised learning," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5070–5079.

[21] L. Jiang, Z. Zhou, T. Leung, L.-J. Li, and L. Fei-Fei, "Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels," in *Proceedings of International Conference on Machine Learning*, 2018, pp. 2304–2313.

[22] D. P. Kingma, D. J. Rezende, S. Mohamed, and M. Welling, "Semi-supervised learning with deep generative models," in *Proceedings of Advances in Neural Information Processing Systems*, 2014, pp. 3581–3589.

[23] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009.

[24] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proceedings of Advances in Neural Information Processing Systems*, 2012, pp. 1097–1105.

[25] S. Laine and T. Aila, "Temporal ensembling for semi-supervised learning," in *Proceedings of International Conference on Learning Representations*, 2017.

[26] K.-H. Lee, X. He, L. Zhang, and L. Yang, "Cleannet: Transfer learning for scalable image classifier training with label noise," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 5447–5456.

[27] J. Li, R. Socher, and S. C. Hoi, "Dividemix: Learning with noisy labels as semi-supervised learning," in *Proceedings of International Conference on Learning Representations*, 2020.

[28] W. Li, L. Wang, W. Li, E. Agustsson, and L. Van Gool, "Webvision database: Visual learning and understanding from web data," *arXiv preprint arXiv:1708.02862*, 2017.

[29] Y. Li, J. Yang, Y. Song, L. Cao, J. Luo, and L.-J. Li, "Learning from noisy labels with distillation," in *Proceedings of IEEE International Conference on Computer Vision*, 2017, pp. 1910–1918.

[30] S. Liu, J. Niles-Weed, N. Razavian, and C. Fernandez-Granda, "Early-learning regularization prevents memorization of noisy labels," *Proceedings of Advances in Neural Information Processing Systems*, vol. 33, pp. 20 331–20 342, 2020.

[31] T. Liu and D. Tao, "Classification with noisy labels by importance reweighting," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 3, pp. 447–461, 2015.

[32] X. Ma, H. Huang, Y. Wang, S. Romano, S. Erfani, and J. Bailey, "Normalized loss functions for deep learning with noisy labels," in *Proceedings of International Conference on Machine Learning*, 2020, pp. 6543–6553.

[33] E. Malach and S. Shalev-Shwartz, "Decoupling "when to update" from "how to update"," in *Proceedings of Advances in Neural Information Processing Systems*, 2017, pp. 960–970.

[34] C. Northcutt, L. Jiang, and I. Chuang, "Confident learning: Estimating uncertainty in dataset labels," *Journal of Artificial Intelligence Research*, vol. 70, pp. 1373–1411, 2021.

[35] S. Park, J. Park, S.-J. Shin, and I.-C. Moon, "Adversarial dropout for supervised and semi-supervised learning," in *Proceedings of AAAI Conference on Artificial Intelligence*, 2018, pp. 3917–3924.

[36] G. Patrini, A. Rozza, A. Krishna Menon, R. Nock, and L. Qu, "Making deep neural networks robust to label noise: A loss correction approach," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 1944–1952.

[37] H. Pham, Z. Dai, Q. Xie, and Q. V. Le, "Meta pseudo labels," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2021, pp. 11 557–11 568.

[38] A. Rasmus, H. Valpola, M. Honkala, M. Berglund, and T. Raiko, "Semi-supervised learning with ladder networks," in *Proceedings of Advances in Neural Information Processing Systems*, 2015, pp. 3546–3554.

[39] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[40] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *Journal of Big Data*, vol. 6, no. 1, pp. 1–48, 2019.

[41] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proceedings of International Conference on Learning Representations*, 2015.

[42] K. Sohn, D. Berthelot, C.-L. Li, Z. Zhang, N. Carlini, E. D. Cubuk, A. Kurakin, H. Zhang, and C. Raffel, "Fixmatch: Simplifying semi-supervised learning with consistency and confidence," in *Proceedings of Advances in Neural Information Processing Systems*, 2020.

[43] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in *Proceedings of AAAI Conference on Artificial Intelligence*, 2017.

[44] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. J. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," in *Proceedings of International Conference on Learning Representations*, 2014.

[45] D. Tanaka, D. Ikami, T. Yamasaki, and K. Aizawa, "Joint optimization framework for learning with noisy labels," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 5552–5560.

[46] A. Vahdat, "Toward robustness against label noise in training deep discriminative neural networks," in *Proceedings of Advances in Neural Information Processing Systems*, 2017, pp. 5596–5605.

[47] A. Veit, N. Alldrin, G. Chechik, I. Krasin, A. Gupta, and S. Belongie, "Learning from noisy large-scale datasets with minimal supervision," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 839–847.

[48] D.-B. Wang, Y. Wen, L. Pan, and M.-L. Zhang, "Learning from noisy labels with complementary loss functions," in *Proceedings of AAAI Conference on Artificial Intelligence*, vol. 35, no. 11, 2021, pp. 10 111–10 119.

[49] Y. Wang, X. Ma, Z. Chen, Y. Luo, J. Yi, and J. Bailey, "Symmetric cross entropy for robust learning with noisy labels," in *Proceedings of IEEE International Conference on Computer Vision*, 2019, pp. 322–330.

[50] X. Xia, T. Liu, B. Han, C. Gong, N. Wang, Z. Ge, and Y. Chang, "Robust early-learning: Hindering the memorization of noisy labels," in *Proceedings of International Conference on Learning Representations*, 2021.

[51] X. Xia, T. Liu, B. Han, M. Gong, J. Yu, G. Niu, and M. Sugiyama, "Sample selection with uncertainty of losses for learning with noisy labels," *arXiv preprint arXiv:2106.00445*, 2021.

[52] X. Xia, T. Liu, B. Han, N. Wang, M. Gong, H. Liu, G. Niu, D. Tao, and M. Sugiyama, "Part-dependent label noise: Towards instance-dependent label noise," in *Proceedings of Advances in Neural Information Processing Systems*, vol. 33, 2020, pp. 7597–7610.

[53] X. Xia, T. Liu, N. Wang, B. Han, C. Gong, G. Niu, and M. Sugiyama, "Are anchor points really indispensable in label-noise learning?" *Proceedings of Advances in Neural Information Processing Systems*, vol. 32, 2019.

[54] T. Xiao, T. Xia, Y. Yang, C. Huang, and X. Wang, "Learning from massive noisy labeled data for image classification," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 2691–2699.

[55] Q. Xie, M.-T. Luong, E. Hovy, and Q. V. Le, "Self-training with noisy student improves imagenet classification," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2020, pp. 10 687–10 698.

[56] I. Z. Yalniz, H. Jégou, K. Chen, M. Paluri, and D. Mahajan, "Billion-scale semi-supervised learning for image classification," *arXiv preprint arXiv:1905.00546*, 2019.

[57] J. Yao, J. Wang, I. W. Tsang, Y. Zhang, J. Sun, C. Zhang, and R. Zhang, "Deep learning from noisy image labels with quality embedding," *IEEE Transactions on Image Processing*, vol. 28, no. 4, pp. 1909–1922, 2018.

[58] K. Yi and J. Wu, "Probabilistic end-to-end noise correction for learning with noisy labels," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 7017–7025.

[59] X. Yu, B. Han, J. Yao, G. Niu, I. Tsang, and M. Sugiyama, "How does disagreement help generalization against label corruption?" in *Proceedings of International Conference on Machine Learning*, 2019, pp. 7164–7173.

[60] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning requires rethinking generalization," in *Proceedings of International Conference on Learning Representations*, 2017.

[61] H. Zhang, M. Cissé, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," in *Proceedings of International Conference on Learning Representations*, 2018.

[62] Z. Zhang and M. R. Sabuncu, "Generalized cross entropy loss for training deep neural networks with noisy labels," in *Proceedings of Advances in Neural Information Processing System*, 2018, pp. 8792–8802.

[63] Z. Zhu, T. Liu, and Y. Liu, "A second-order approach to learning with instance-dependent label noise," in *In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2021, pp. 10 113–10 123.