

Received 20 October 2022, accepted 2 November 2022, date of publication 4 November 2022, date of current version 9 November 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3219810

RESEARCH ARTICLE

Self-Augmentation Based on Noise-Robust Probabilistic Model for Noisy Labels

BYOUNG WOO PARK¹, SUNG WOO PARK², AND JUNSEOK KWON^{1,2}, (Member, IEEE)

¹Graduate School of Artificial Intelligence, Chung-Ang University, Seoul 06974, South Korea

²School of Computer Science and Engineering, Chung-Ang University, Seoul 06974, South Korea

Corresponding author: Junseok Kwon (jskwon@cau.ac.kr)

This work was supported in part by the Chung-Ang University Research Grant in 2022; in part by the Ministry of Culture, Sports and Tourism; and in part by the Korea Creative Content Agency under Project R2021040032.

ABSTRACT Learning deep neural networks from noisy labels is challenging, because high-capacity networks attempt to describe data even with noisy class labels. In this study, we propose a self-augmentation method without additional parameters, which handles noisy labeled data based on small-loss criteria. To this end, we use small-loss samples by introducing a noise-robust probabilistic model based on a Gaussian mixture model (GMM), in which small-loss samples follow class-conditional Gaussian distributions. With this sample augmentation using the GMM-based probabilistic model, we can effectively solve over-parameterization problems induced by label inconsistency in small-loss samples. We further enhance the quality of the small-loss samples using our data-adaptive selection strategy. Consequently, our method prevents networks from over-parameterization and enhances their generalization performance. Experimental results demonstrate that our method outperforms state-of-the-art methods for learning with noisy labels on several benchmark datasets. The proposed method produced a remarkable performance gap of up to 12% compared with the previous state-of-the-art methods on CIFAR dataset.

INDEX TERMS Computer vision, deep learning, probability distribution.

I. INTRODUCTION

The generalization performance of supervised classification problems depends heavily on accurate label information. However, the construction of high-quality ground-truth class labels for large-scale datasets from domain experts is extremely expensive and time-intensive. Thus, in practice, data with labels are collected from the internet through various methods such as crowdsourcing [3] and online queries [4]. However, these data inevitably contain noisy labels, which results in a poor generalization performance of deep neural networks (DNNs).

To solve this problem, recent methods [1], [5], [6], [7], [8], [9] have adopted small-loss criteria. By training networks using samples that produce small losses, data samples can be employed with correct labels to improve the robustness of the network toward noisy labels. However, although networks

based on small-loss criteria can be trained with clean samples in the early training stages, they are gradually over-fitted with samples with false labels as the training proceeds, owing to the *memorization effect* of DNNs [10], [11], which degrades the learning performance of the networks.

The following two representative approaches can prevent the aforementioned degradation in learning samples with small-loss values. In the first approach, filtering methods [1], [8], [9] are used to filter out wrongly selected noisy samples from small-loss samples, thereby reducing the accumulated errors induced by noisy samples. For this, two networks are utilized simultaneously and the different learning abilities of the networks are shared. However, employing multiple networks requires additional learning parameters, inducing a high computational cost for training. In addition, as shown in Fig. 1(b), the learning abilities of the two networks do not significantly differ from each other after over-parameterization. Thus, they fail to effectively filter out noisy samples. In the second approach, sophisticated small-loss selection rules are

The associate editor coordinating the review of this manuscript and approving it for publication was Pietro Savazzi¹.

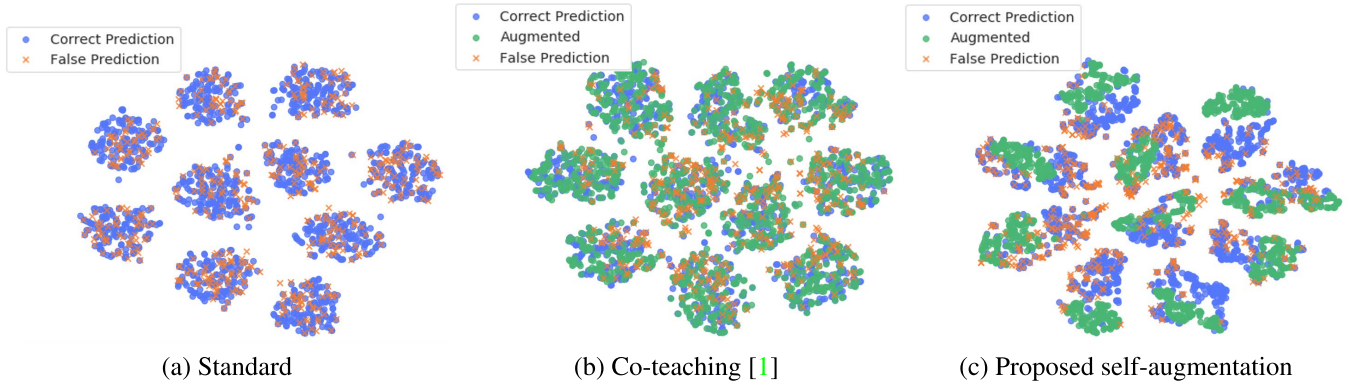


FIGURE 1. Visualization of learned representations. We conducted experiments using the CIFAR10 train dataset with 50% symmetric noise and visualized the pre-softmax layer using 2D t-SNE (t-distributed stochastic neighbor embedding) [2]. Each cluster indicates a set of data with the same prediction class inferred by the network. **Blue** colors represent samples with correct labels and correct network predictions. **Orange** colors represent samples with correct labels but false network predictions. **Green** colors represent network prediction made by additional networks in (b) and newly generated sample from the proposed GMM in (c). (a) Deep neural networks that are directly trained on noisy datasets (*i.e.*, Standard) cannot differentiate between samples with correct (*i.e.*, blue) and false (*i.e.*, orange) predictions owing to network over-parameterization. (b) Co-teaching [1] augments samples (*i.e.*, green) using different learning abilities of dual networks. However, owing to network over-parameterization, learning abilities of the two networks become similar. Thus, they fail to differentiate between samples with correct (*i.e.*, blue) and false (*i.e.*, orange) predictions. (c) Our method enables reliable augmentation (*i.e.*, green) using GMM, which prevents the network from being overly fitted to noisy samples. Thus, correct (*i.e.*, blue) and false (*i.e.*, orange) predictions have relatively better representations than previous methods in the light of separation.

used. In [1], small-loss samples are empirically selected using manually tuned selection strategies based on the network memorization effect, which is not suitable for real-world data. Thus, to find the optimal selection strategy, a data-adaptive selection strategy [12] has been proposed using automated machine learning. However, it requires a large amount of high computational resources to determine the optimal strategy.

In this study, we propose a self-augmentation method using only a single network, which can efficiently handle noisy labeled data based on small-loss criteria. Our method solves the existing problems of the two aforementioned approaches without additional parameters using the augmentation method. The method effectively regularizes over-parameterization based on the network's own inference. In particular, to protect deep neural networks from over-parameterization, we present a noise-robust probabilistic model as a strong statistical structure, such as Gaussian mixture model (GMM), in which small-loss samples follow class-conditional multivariate Gaussian distributions. Using the proposed GMM, we augment the samples to effectively regularize the over-parameterization of the network. Unlike [13], where training samples follow a Gaussian distribution during the test time with a pre-trained network, we conduct self-augmentation during training without additional parameters and supplementary information. Furthermore, we present a novel data-adaptive selection strategy based on the covariance structure of the proposed GMM. Using our data-adaptive selection strategy, we can considerably enhance the learning performance of the networks by training with high-quality small-loss samples (*i.e.*, less noisy). Fig.1 shows the effectiveness of the proposed method.

Our main contributions are as follows:

- We propose a novel self-augmentation method based on a single network that handles noisy labeled data

based on small-loss criteria. To this end, we present a robust probabilistic model, the GMM, that employs class-conditional Gaussian distributions with moving weighted Gaussian moments across learning steps. Then, sample augmentation using the proposed GMM effectively prevents over-parameterization.

- We introduce a novel data-adaptive selection strategy that enhances the quality of small-loss samples. By training networks with our high-quality small-loss samples, our method considerably improves the generalization performance of the networks.
- We experimentally demonstrate that our method significantly outperforms other state-of-the-art methods for supervised classification tasks with noisy labels. Small-loss samples selected by our sample selection strategy are less noisy than those of other strategies.

Organization is as follows. Section II relates the proposed method with existing methods. Section III introduces notations and terminologies used in this study. Section IV presents the proposed self-augmentation method to reduce the negative effects of noisy samples in small-loss data. We conclude the study in Section VII.

II. RELATED WORK

While there are a long history on methods that deal with noisy labels, we only discuss relevant approaches, including loss or label correction, small-loss selection, and mixture model methods. Table 1 highlights differences between the proposed method and other methods.

A. LOSS OR LABEL CORRECTION

Noisy labels have been addressed during training via loss or label correction. For example, a loss correction method FW [14] estimated the probability that each class was

TABLE 1. Differences of the proposed method to other representative methods.

Methods	Parametric	Distillation	Sample-Selection	Post-hoc
Co-teaching	✓	✓	✓	×
JoCoR	✓	✓	✓	×
DivideMix	✓	×	✓	×
RoG	×	×	×	✓
Ours	×	×	✓	×

corrupted into another and used the probability during training. This method was enhanced by utilizing additional information data with trusted labels GLC [15]. In [16], samples were re-weighted based on their gradient directions to avoid overfitting to label noise. In addition, these noisy labels were handled by considering the consistency of predictions in a loss function [17]. In SELFIE [7], noisy samples with consistent label predictions were exploited and corrected to prevent noise accumulation. They transformed noisy labels either explicitly or implicitly into clean labels by correcting classification losses. In this case, the influence of noisy labels can be accumulated when the correction mechanism does not work appropriately. In contrast, small-loss selection-based approaches with well-designed selection strategies like our method can be free from this noise accumulation by discarding noisy samples.

B. SMALL-LOSS SELECTION

Noisy labels has been avoided via small-loss selection. However, the small-loss samples could still contain noisy samples. To solve this problem, a pre-trained teacher network was used to select small-loss samples that guide the training of a student network Mentornet [5]. In Co-teaching [1], Co-teaching+ [9], small-loss samples were exploited and exchanged via two networks to update the network parameters. Similar to [1], a joint training strategy was employed in JoCoR [8] to select clean samples effectively. However, employing additional networks resulted in high computational cost and increased the number of parameters. In contrast, we propose a self-augmentation strategy that uses only a single network to train the network without additional parameters.

C. MIXTURE MODEL

Mixture models have been introduced for noise-robust training. In [6], and [18] mixture models were employed to compute the confidence of the samples being clean. In particular, a two-component mixture model with beta distributions [18] was used to estimate the sample confidence. Similarly, a two-component mixture model with Gaussian distributions DivideMix [6] was used for this purpose. However, two-component mixture models cannot sufficiently capture the class-wise properties in the training data. In RoG [13], generative classifiers were induced and integrated into pre-trained networks to obtain more accurate decision boundaries in embedded feature spaces. This method assumes that

class-conditional distributions follow a multivariate Gaussian distribution. As in [13], our method assumes that small-loss samples follow class-conditional multivariate Gaussian distributions. However, unlike in [13], our method introduces no additional classifiers, parameters, and supplemental information during training using a novel self-augmentation method based on data-adaptive probabilistic models.

III. NOTATIONS AND PROBLEM SETUP

For classification tasks with C classes, $\tilde{D} = \{\tilde{x}_i, \tilde{y}_i\}_{i=1}^N$ denotes N pairs of training data in noisy training dataset $\tilde{D}$, where $\tilde{x}_i \in \mathbb{R}^d$ is the i -th sample paired with the probably noisy label $\tilde{y}_i \in \{1, \dots, C\}$. We denote the network predicted class of data sample as $\hat{y}$ and a deep neural network with parameters θ as f_θ . We then optimize the network parameters by minimizing the following objective loss function (*i.e.*, cross-entropy loss, $\mathcal{L}_{ce}(\tilde{D}, \theta)$).

$$\mathbb{E}_{\tilde{D}}[\mathcal{L}_{ce}(\tilde{D}, \theta)] = -\frac{1}{N} \sum_{i=1}^N \sum_{c=1}^C \tilde{y}_i \log p_c(f_\theta(\tilde{x}_i)), \quad (1)$$

where $f_\theta(\tilde{x}_i)$ outputs an estimated label and $p_c(\cdot)$ transforms the output into a probability. However, because of the existence of noisy labels, learning the network parameters θ by optimizing (1) inevitably induces over-parameterization. Thus, the test accuracy decreases considerably, leading to poor generalization performance of the network.

To solve this over-parameterization problem, small-loss criteria have been widely adopted to deal with noisy labeled data. Intuitively, training samples that produce small losses are likely to be correctly labeled. Thus, by training DNNs with small-loss samples, the networks can be made robust to noisy samples. Thus, to select high-quality small-loss samples, we employ a cross-entropy-based selection strategy similar to that used in [1] and [5]. In particular, we select $1 - C(k)^1$ percent of samples, which have low cross-entropy loss values in (1) at each learning step k .

While the label information is not considered by the proposed selection strategy, the selected small-loss training samples contain inconsistent label information (*e.g.*, flipped labels). Overall, the proportion of noisy samples tends to gradually increase as the training proceeds [1]. Then, the network trained with small-loss samples eventually induces over-parameterization problems. Thus, we require an additional strategy to address this phenomenon.

In Section IV, we present a self-augmentation method to reduce the negative effects of noisy samples in small-loss data. In addition, we propose a data-adaptive selection strategy $C(k)$ in Section IV-C, which determines high-quality small-loss samples at each learning step. For readability, we denote a small-loss sample as x and the corresponding network output (pre-softmax logit) as $X = f_\theta(x)$. Furthermore, $x_{c,k,i}$ represents the i -th small-loss sample that is predicted to the c -th class at learning step k , and $X_{c,k,i}$ denotes

¹The detailed definition of $C(k)$ is introduced in Section IV-C.

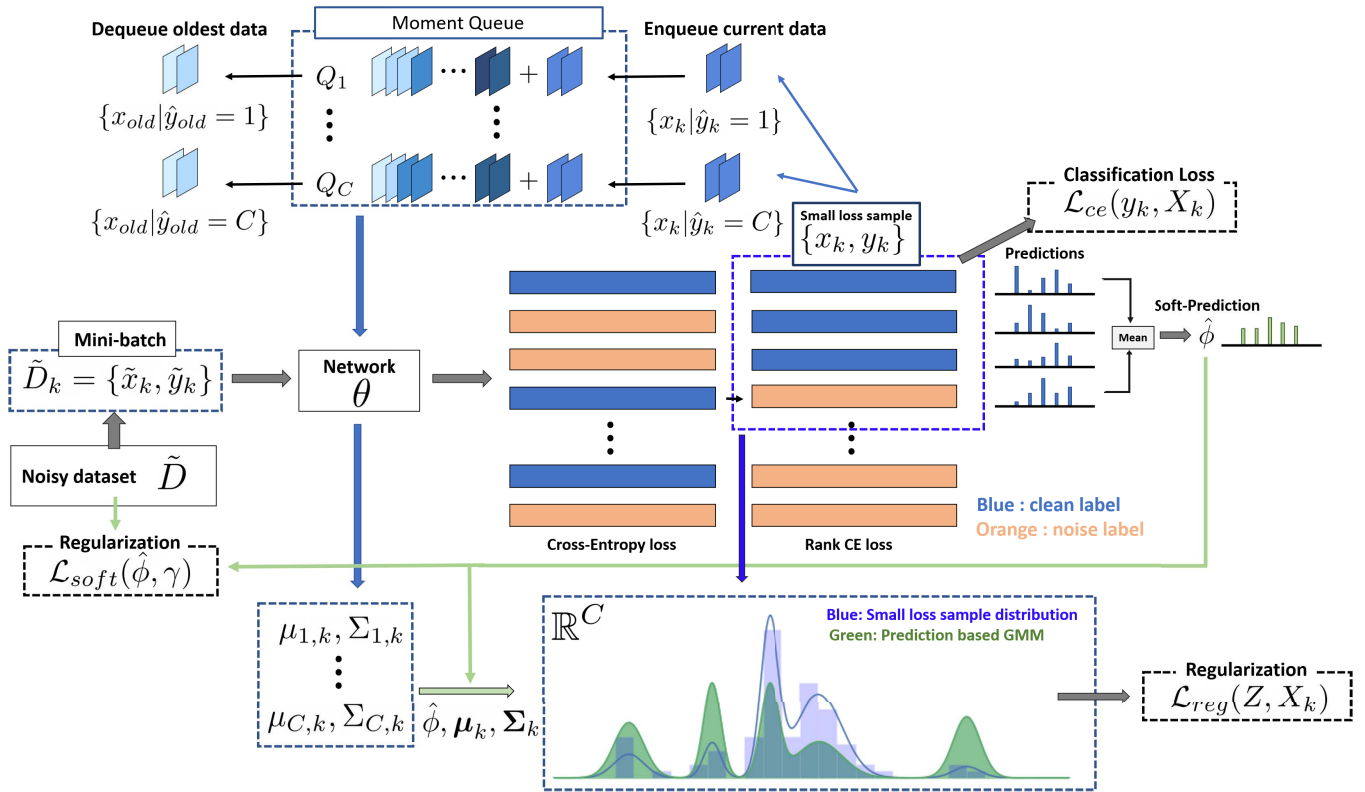


FIGURE 2. Pipeline of the proposed method. For each mini-batch of training data, we select samples with small cross-entropy loss values, i.e., small-loss samples (Section IV-C). Then, we update the *moving weighted moment queue* (Section IV-B) based on network predictions of selected samples. In particular, we enqueue current data with the same predicted label and dequeue old data for each predicted class. Subsequently, we induce GMM (Section IV-A) using weighted inference statistics of data in queue at a previous learning step and soft-prediction results at a current step.

their corresponding logit. To avoid confusion and improve readability, we omit the indices if they are understood in the context.

IV. SELF-AUGMENTATION WITH NETWORK PREDICTION

We propose a self-augmentation method using a noise-robust probabilistic model as a GMM with network inference. We assume that small-loss samples with the same predicted class follow class-conditional multivariate Gaussian distributions. Fig.2 shows the pipeline of our method.

A. NOISE-ROBUST PROBABILISTIC MODEL

For C -class classification, we define C Gaussian distributions and induce a probabilistic model as a GMM with a class-wise mixture probability prior ϕ_c as follows:

$$P(X) = \sum_{c=1}^C \phi_c \mathcal{N}(X | \mu_c, \Sigma_c), \quad (2)$$

where $\mathcal{N}(X | \mu_c, \Sigma_c)$ denotes a normal distribution with mean μ_c and standard deviation Σ_c . To adopt the GMM as an augmented sample generative model, we compute data-adaptive mixture probabilities and Gaussian statistics (i.e., μ_c and Σ_c) with respect to network outputs X , where each term of GMM

is computed as follows.

$$\phi_c = \frac{|X_c|}{|X|}, \sum_c \phi_c = 1, \quad (3)$$

where $0 \leq \phi_c \leq 1, \forall c \in [C]$, and

$$\begin{aligned} \mu_c &= \mathbb{E}[X_c], \\ \Sigma_c &= \text{Cov}[X_c] = \mathbb{E}[X_c X_c^T] - \mathbb{E}[X_c] \mathbb{E}[X_c]^T, \end{aligned} \quad (4)$$

where $X_c = X \cdot 1_{\hat{y}=c}$ denotes the logits predicted to the c -th class and $1_{(\cdot)}$ denotes an indicator function for the predicted label of X . To prevent over-parameterization, we regularize the network by minimizing the discrepancy between small-loss samples and augmented samples Z from GMM (i.e., $Z \sim P(X)$ in (2)). However, the probabilistic model in (2) cannot guarantee the avoidance of over-parameterization. Thus, to prevent over-parameterization by guaranteed augmentation, we introduce an additional regularization term.

In course of minimizing the conventional objective function in (1), the confidence of the network prediction can be skewed toward specific uncertain classes [19]. If a network is eventually over-parameterized, it tends to assign most labels to a single class [20]. Thus, the network predictions for selected cross-entropy-based small-loss samples are also skewed to particular classes. In addition, a mixture probability prior ϕ_c in (3) based on network prediction for small-loss

Algorithm 1 Moving Weighted Moment Queue

Require: x_k : data with small-loss values at a current step, M : the number of collected samples for each class, $Q_{k-1} = \{Q_{1,k-1}, \dots, Q_{C,k-1}\}$: queue at a previous step.
for $c = 1$ to C (i.e., C -class classification) **do**
 Calculate $X_k = f_\theta(x_k)$
 Predict $\hat{y}_k = \arg \max_c p_c(X_k)$
 // Enqueue current samples with the same predicted class.
 Enqueue $Q_{c,k} = Q_{c,k-1} \cup \{x_k | \hat{y}_k = c\}$
 if $|Q_{c,k}| \geq M$ **then**
 // Dequeue $|Q_{c,k}| - M$ old samples.
 Dequeue $Q_{c,t} = Q_{c,t} / \{x_{old} | \hat{y}_{old} = c\}$
 end if
end for
Obtain $Q_k = \{Q_{1,k}, \dots, Q_{C,k}\}$
Calculate $\mu_{c,k}, \Sigma_{c,k}$ using (7)
Out $\mu_k = [\mu_{1,k}, \dots, \mu_{C,k}]$, $\Sigma_k = [\Sigma_{1,k}, \dots, \Sigma_{C,k}]$

samples peaks at few classes. Subsequently, the proposed probabilistic model built with ϕ_c cannot deal with all classes fairly. In this case, we cannot ensure reliable augmented samples.

To address this issue, we replace ϕ_c with soft-mixture probability $\hat{\phi}_c$ to attenuate the skewness of the probabilistic model [21]. The soft mixture probability $\hat{\phi}_c$ is defined as

$$\hat{\phi}_c = \sum_{i=1}^{|X|} \frac{p_c(X_i)}{|X|}. \quad (5)$$

Using $\hat{\phi}_c$, we can consider the network confidence for all classes, where $\hat{\phi}_c$ in (5) becomes flatter than ϕ_c in (3), which reflects only few skewed classes. In addition, we present an additional regularization loss term to further attenuate the skewness of the mixture probability as the training proceeds. The regularization loss, $\mathcal{L}_{soft}$, for the soft mixture probability $\hat{\phi}_c$ is defined as follows:

$$\mathcal{L}_{soft}(\hat{\phi}, \gamma) = D_{KL}(\hat{\phi} || \gamma) = \sum_{c=1}^C \hat{\phi}_c \log \frac{\hat{\phi}_c}{\gamma_c}, \quad (6)$$

where D_{KL} denotes the Kullback–Leibler (KL) divergence, and γ_c is the prior probability distribution of the c -th class among all training samples.

B. MOVING WEIGHTED MOMENT QUEUE

Because the selected small-loss training samples inescapably contain inconsistent label information, the network is gradually fitted to noise-labeled samples. Thus, the decision boundaries of the network on the test dataset fluctuate as the training proceeds, which leads the network to produce inconsistent predictions. Thus, the accumulated error from inconsistent network predictions eventually induces over-parameterization problems. Because the Gaussian parameters are computed using inconsistent network prediction, the GMM with naive Gaussian statistics in (4) cannot provide

guaranteed augmentation to prevent the network from over-parameterization.

In this section, we aim to build the machinery that reduces the aforementioned fluctuations in decision boundaries. Instead of computing Gaussian parameters based only on the network prediction of a current step, we consecutively collect the first- and second-order statistical moments (i.e., μ, Σ) of small-loss samples with our moving weighted moment queue Q_k at each learning step k . In particular, we dynamically collect M small-loss samples based on the network predictions in the queue. For this, we update the c -th queue with $\beta_{c,k}$ small-loss samples, whose network prediction is the c -th class at learning step k . By weighting the statistical moments of small-loss samples by $\beta_{c,k}$ for a different learning step k , we can consider the confidence of network prediction for previous learning steps altogether. In updating (i.e., enqueue) the queue with small-loss samples at the current learning step, we discard (i.e., dequeue) $M - \beta_{c,k}$ small-loss samples of the early learning steps. Thus, small-loss samples selected from the early steps are forced to have less influence on network parameter updates. To reflect the aforementioned considerations for adopting a queue, we define the *moving weighted moment queue* Q_k in Algorithm 1, and the corresponding Gaussian moments as

$$\begin{aligned} \mu_{c,k} &= \frac{\beta_{c,k} \mathbb{E}[X_{c,k}] + \dots + \beta_{c,k-l_c} \mathbb{E}[X_{c,k-l_c}]}{\sum_{j=k-l_c}^k \beta_{c,j}}, \\ \Sigma_{c,k} &= \frac{\beta_{c,k} \mathbf{Var}[X_{c,k}] + \dots + \beta_{c,k-l_c} \mathbf{Var}[X_{c,k-l_c}]}{\sum_{j=k-l_c}^k \beta_{c,j}}, \end{aligned} \quad (7)$$

where $X_{c,k}$ is a small-loss sample with the prediction to the c -th class at learning step k and $\beta_{c,k}$ is the number of small-loss samples with the prediction to the c -th class at learning step k (i.e., $\beta_{c,k} = |\{x_k | \hat{y}_k = c\}|$). In (7), l_c is a moving window that can be adaptively determined by $l_c = \arg \min_{j_c} \max[(\sum_{k-j_c}^k \beta_{c,k-j_c} - M), 0]$.

Using new Gaussian statistics $\mu_{c,k}, \Sigma_{c,k}$ in (7) and the proposed soft mixture probability $\hat{\phi}_c$ in (5), we improve the

conventional probabilistic model in (2) into a noise-robust probabilistic model as follows:

$$P(X_k) = \sum_{c=1}^C \hat{\phi}_c \mathcal{N}(X_k | \mu_{c,k}, \Sigma_{c,k}). \quad (8)$$

Subsequently, we regularize the network parameters by minimizing the KL-divergence between the small-loss samples and augmented samples from a modified GMM in (8) without label information as follows:

$$\begin{aligned} \mathcal{L}_{reg}(Z, X_k) &= D_{KL}(p(Z) || p(X_k)) \\ &= \frac{1}{|X_k|} \sum_{i=1}^{|X_k|} \sum_{c=1}^C p_c(Z) \log \frac{p_c(Z_i)}{p_c(X_{k,i})}, \end{aligned} \quad (9)$$

where $Z \stackrel{i.i.d}{\sim} P(X_k)$ are *i.i.d* $|X_k|$ samples. The augmented samples Z are directly sampled from the proposed GMM. In (9), our augmentation helps the network to mitigate the accumulated error from inconsistent predictions by providing consistent guidance to the network for each learning step. Therefore, it enables the network to avoid overfitting to noisy samples and prevents over-parameterization. Algorithm 1 describes our moving weighted moment queue.

C. DATA ADAPTIVE SELECTION STRATEGY

In contrast to conventional methods that manually tune selection thresholds, we propose a data-adaptive small-loss selection strategy based on eigenvalues of the second moments Σ_k in the queue. Please note that we assume that each covariance matrix $\Sigma_{c,k}$ is diagonal for easy computation. Eigenvalues can be considered as an indicator of network certainty in each class. In particular, if a network is over-parameterized, it can confidently predict classes, even for noisy samples. In this case, the eigenvalues of Σ_k have small values owing to their certainty. For example, if the network is perfectly fitted to training samples, its prediction can be represented as a one-hot vector (over certainty); then, the eigenvalues of $\Sigma_{c,k}$, $\lambda_{c,k}$, have small values for all classes. In contrast, if the network is under-parameterized, the certainty of network prediction for the training sample is relatively lower than that of the over-parameterized case, which means $\lambda_{c,k}$ has large values for all classes. Thus, the following inequality holds:

$$\underbrace{\arg \min_j \mathbb{E}_c[\lambda_{c,j}]}_{\text{Over-parameterization}} \leq \underbrace{\mathbb{E}_c[\lambda_{c,k}]}_{\text{Prediction certainty}} \leq \underbrace{\arg \max_j \mathbb{E}_c[\lambda_{c,j}]}_{\text{Under-parameterization}}, \quad (10)$$

where $\mathbb{E}_c[\lambda_{c,k}]$ denotes the mean eigenvalues of the second moments $\Sigma_{c,k}$ for all classes at learning step k . Fig.3 shows under- and over-parameterization in terms of eigenvalues.

To enhance the generalization performance of the networks, we balance between under-parameterization and over-parameterization by selecting an appropriate threshold for selecting small-loss samples at each learning step. To determine an appropriate threshold dynamically, we design a

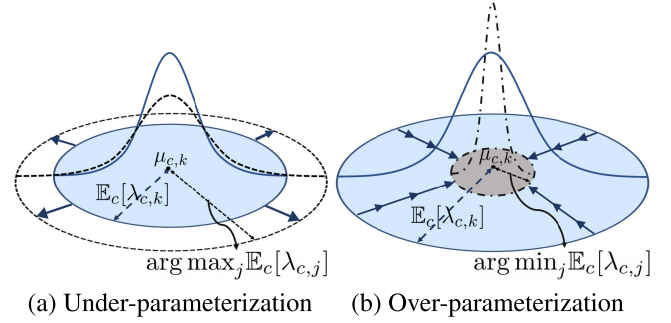


FIGURE 3. Determination of under or over parameterization using eigenvalues. (a) If a network does not begin to memorize noisy samples, mean eigenvalues of all classes $\mathbb{E}_c[\lambda_{c,k}]$ increase due to under-parameterization. (b) After the network begins to memorize noisy sample, $\mathbb{E}_c[\lambda_{c,k}]$ decreases due to over-parameterization.

data-adaptive selection strategy in proportion to the difference between the maximum and minimum values of the mean of eigenvalues $\mathbb{E}_c[\lambda_{c,k}]$ across the previous learning steps k as

$$C(k) = \alpha \log \left(\arg \max_k \mathbb{E}_c[\lambda_{c,k}] - \arg \min_k \mathbb{E}_c[\lambda_{c,k}] + 1 \right), \quad (11)$$

where α is a hyperparameter that determines the selection scale. Our selection strategy $C(k)$ is designed to satisfy the following conditions: (1) $C(k) \in [0, 1]$; (2) $C(k)$ is a non-decreasing function on k . Because of the *memorization effect* of DNNs, they learn clean samples at the beginning of training, and then gradually memorize noisy sample which induce over-parameterization. Thus, $C(k)$ needs to be increased to reduce the number of selected small-loss samples as the training proceeds to ignore noisy samples. Specifically, before the network memorizes the noisy samples, the network is safely updated with clean samples. The certainty of network prediction increases and $\mathbb{E}_c[\lambda_{c,k}]$ reaches the highest value at a certain step, which is immediately before the memorization. Then, when the network starts to memorize noisy samples, it is gradually over-parameterized. Hence, $\mathbb{E}_c[\lambda_{c,k}]$ decreases as the training proceeds. Consequently, the gap between the maximum and minimum values of $\mathbb{E}_c[\lambda_{c,k}]$ across the previous learning steps increases. The number of small-loss samples decreases as the training proceeds. Thus, the network parameters are updated safely with qualified small-loss samples.

D. OBJECTIVE FUNCTION

Our main objective function is expressed as follows:

$$\mathcal{L} = \mathcal{L}_{ce}(y_k, X_k) + \mathcal{L}_{reg}(Z, X) + \omega \mathcal{L}_{soft}(\hat{\phi}, \gamma), \quad (12)$$

where $\mathcal{L}_{ce}$ in (1) denotes a cross-entropy loss function, $\mathcal{L}_{reg}$ in (9) and $\mathcal{L}_{soft}$ in (6) denote regularization loss functions for augmented samples and soft mixture probability, respectively, and ω is a hyperparameter that weighs the regularization effect of soft-mixture probability. By minimizing (12),

Algorithm 2 Self-Augmentation in a Single Network

Require: f : Neural network, θ : network parameters, $\tilde{D}$: training dataset, Q : moment queue, M : number of samples in queue, C : number of classes, γ : class probability prior, α , ω : hyperparameters, and η : learning rate.

for $t = 1$ to T (i.e., total number of training epochs) **do**

Shuffle training set $\tilde{D}$.

for $k = 1$ to K (i.e., total number of iterations) **do**

Fetch mini-batch $\tilde{D}_k$ from $\tilde{D}$.

Obtain small-loss sample $\{x_k, y_k\}$ // sample $(1 - C(k))$ percent of data with low cross-entropy losses in $\tilde{D}_k$.

Calculate $X_k = f_{\theta_k}(x_k)$.

Calculate $\hat{\phi} = \frac{1}{|X_k|} \sum_{X_k} \text{softmax}(X_k)$ using (5).

Update the moving weighted moment queue Q_k using Algorithm 1.

if $|Q_k| < C \times M$ **then**

Update $\theta_{k+1} = \theta_k - \eta[\nabla \mathcal{L}_{ce}(Y_k, X_k) + \lambda \nabla \mathcal{L}_{soft}(\hat{\phi}, \gamma)]$.

end if

if $|Q_k| = C \times M$ **then**

Obtain Sample $Z \stackrel{i.i.d}{\sim} P(X_k) = \sum_c \hat{\phi}_c \mathcal{N}(X_k | \mu_{c,k}, \Sigma_{c,l})$ using (8).

Update $\theta_{k+1} = \theta_k - \eta[\nabla \mathcal{L}_{ce}(Y_k, X_k) + \mathcal{L}_{reg}(Z, X_k) + \omega \nabla \mathcal{L}_{soft}(\hat{\phi}, \gamma)]$ using our objective function in (12).

Obtain eigenvalues $\lambda_{c,k}$ from $\Sigma_{c,k}$ in (7).

Update $C(k) = \alpha \log \left(\arg \max_k \mathbb{E}_c[\lambda_{c,k}] - \arg \min_k \mathbb{E}_c[\lambda_{c,k}] + 1 \right)$ in (11)

end if

end for

end for

TABLE 2. Detailed network architectures used for CIFAR-10 and CIFAR-100 datasets.

CNN on CIFAR-10 / CIFAR-100
32 × 32 RGB Image
3 × 3 conv, 128 LReLU, BN
3 × 3 conv, 128 LReLU, BN
3 × 3 conv, 128 LReLU, BN
2 × 2 Max-pool dropout, $p = 0.25$
3 × 3 conv, 256 LReLU, BN
3 × 3 conv, 256 LReLU, BN
3 × 3 conv, 256 LReLU, BN
2 × 2 Max-pool dropout, $p = 0.25$
3 × 3 conv, 512 LReLU, BN
3 × 3 conv, 256 LReLU, BN
3 × 3 conv, 128 LReLU, BN
Avg-pool
Dense 128 → 10 / 100

we prevent over-parameterization and make the network robust to noisy labels. Algorithm 2 describes the overall algorithm of the proposed method.

V. EXPERIMENTS**A. EXPERIMENTAL SETTINGS****1) NETWORK ARCHITECTURE**

Table 2 shows detailed network architectures of baseline methods with the 9-layered CNN models [1], in which conv, LReLU, BN, and Dense denote a convolution filter, leaky

TABLE 3. Hyperparameter settings for different datasets.

Datasets	noise rate	α	M	λ
CIFAR-10	Symmetric 20%	0.3	50	2
	Symmetric 50%	0.5	50	2
	Asymmetric 45%	0.6	50	2
CIFAR-100	Symmetric 20%	0.1	5	2
	Symmetric 50%	0.18	5	2
	Asymmetric 45%	0.25	5	2
Clothing-1M	—	0.15	5	2

rectified linear unit, batch normalization, and a fully connected layer, respectively.

2) HYPERPARAMETERS AND BASELINES

For fair comparisons on the CIFAR-10/100 datasets, we followed the experimental settings suggested by [1]. We used a single 9-layered CNN with a batch size of 128 as the baseline architecture. The Adam optimizer with a momentum of 0.9 was used with an initial learning rate of 0.001 and was linearly decayed to zero as training proceeded. Because the CIFAR dataset contains the same number of images for all 10 classes (5K images for each class), we set the class-probability γ_c in (6) as $\frac{1}{10}$ for all classes. For appropriate selection, we control $C(k)$ by α in (11) with different noise settings. In particular, we set $\alpha = \{0.3, 0.5, 0.6\}$ for CIFAR-10 and $\alpha = \{0.1, 0.18, 0.25\}$ for CIFAR-100, respectively.

For the Clothing1M dataset, a pre-trained ResNet50 was used with a batch size of 64 as the baseline. We also adopted Adam optimizer with momentum of 0.9. We run 200 epochs in total and set a learning rate to 5×10^{-5} and 8×10^{-6} for

TABLE 4. Average test accuracy (%) on the CIFAR-10 dataset over last 10 epochs with various noise corruptions. (a) Single network-method results. (b) Two network-method results. The symbol * indicates scores provided by the corresponding authors. The best results are boldfaced.

	Methods	Symmetric 20%	Symmetric 50%	Asymmetric 45%
(a)	Standard	71.91 $\pm$.43	49.54 $\pm$.41	49.06 $\pm$ 1.02
	GCE	84.68 $\pm$.05	61.80 $\pm$.11	61.09 $\pm$.18
	SL	84.04 $\pm$.04	58.50 $\pm$.56	48.87 $\pm$.19
	NPCL*	84.30 $\pm$.07	77.66 $\pm$.09	—
(b)	MentorNet	80.76 $\pm$.36	71.10 $\pm$.48	58.14 $\pm$.38
	Co-teaching	78.23 $\pm$.27	72.81 $\pm$.20	70.46 $\pm$.58
	Co-teaching ⁺	80.64 $\pm$.15	58.43 $\pm$.30	70.78 $\pm$.11
	JoCoR	85.73 $\pm$.19	79.41 $\pm$.25	64.21 $\pm$.12
	DivideMix	—	81.13 $\pm$.18	68.93 $\pm$.33
	Ours	87.35 $\pm$.02	81.47 $\pm$.07	80.92 $\pm$.03

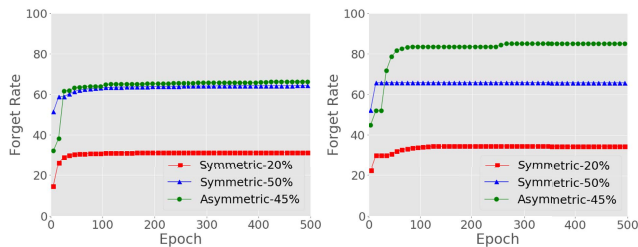


FIGURE 4. Forget rate $C(k)(\%)$ vs. epochs. Left: CIFAR-10 dataset, Right: CIFAR-100 dataset.

TABLE 5. Summary of the datasets used in the experiments.

	# of training	# of test	# of class	size
CIFAR-10	50,000	10,000	10	32×32
CIFAR-100	50,000	10,000	100	32×32
Clothing1M [23]	1,000,000	10,000	14	224×224

first and second 40 epochs, respectively, and 8×10^{-6} from 80 epochs. Because data imbalance existed, we randomly selected balanced subsets using noisy labels to mitigate data imbalance [22] and used them for training. These subsets had 270K images for all classes. For each epoch, we selected 32K images for efficient training. For pre-processing, we conducted random center cropping, random flipping, and normalization to 224×224 pixels. We set γ_c in (6) to $\frac{1}{14}$ for all classes. Table 3 shows detailed hyperparameter settings.

3) SMALL LOSS SAMPLE SELECTION SETTINGS

The network was trained with entire noisy data (*i.e.*, $C(k) = 0$) for first 5 epochs as the warm-up stage. In this warm-up stage, the network was adapted to training data and its certainty for training samples was stabilized. After the warm-up stage, we gradually increased the forget rate $C(k)$ as training proceeds based on our selection strategy to reduce the number of selected small-loss samples and avoid the memorization effect, as shown in Fig.4. For appropriate sample selection, we adopted hyperparameter α in (11) to scale down $C(k)$ to $[0, 1]$ and clip $C(k)$ to 0.85 if $C(k) > 0.85$.

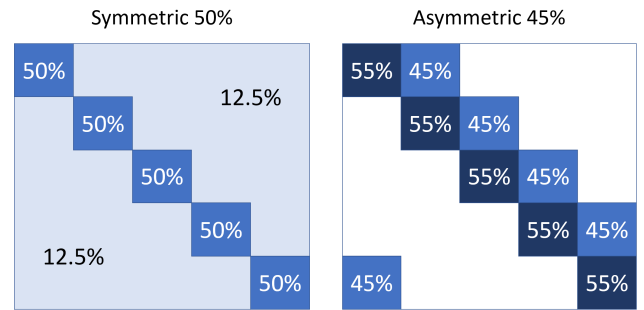


FIGURE 5. Examples of noise transition matrices for 5 classes according to different noise types.

4) TEST DATASETS AND EVALUATION METRICS

We verified the effectiveness of our method on three benchmark datasets, namely CIFAR-10, CIFAR-100, and Clothing1M. Clothing1M [23] contains real-world noisy data with 14 classes [8], [14], [24], [25]. Because all datasets, except for Clothing 1M contained only clean data, we manually injected label noise into these datasets using the noise transition matrix Q , where $Q_{ij} = P(\hat{y} = j | y = i)$ given that a clean label y is flipped into a noisy label $\hat{y}$. We assumed that Q has two typical structures [1]: (1) symmetric flipping [26], which involves flipping uniformly to all classes, and (2) asymmetric flipping [14], which involves flipping to a different class. The noise rate of label flipping, ϵ , was chosen to be $\{20\%, 50\%, 45\%\}$. For asymmetric flipping, if $\epsilon = 45\%$, the training data are *extremely* noisy.

We evaluated the compared methods in terms of test accuracy (*i.e.*, $\frac{\text{\# of correct predictions}}{\text{\# of test data}}$) and label precision (*i.e.*, $\frac{\text{\# of clean samples}}{\text{\# of selected samples}}$). A high label precision means that less noisy data can be used for training. Thus, networks are less affected by noisy data and are robust to noisy labels.

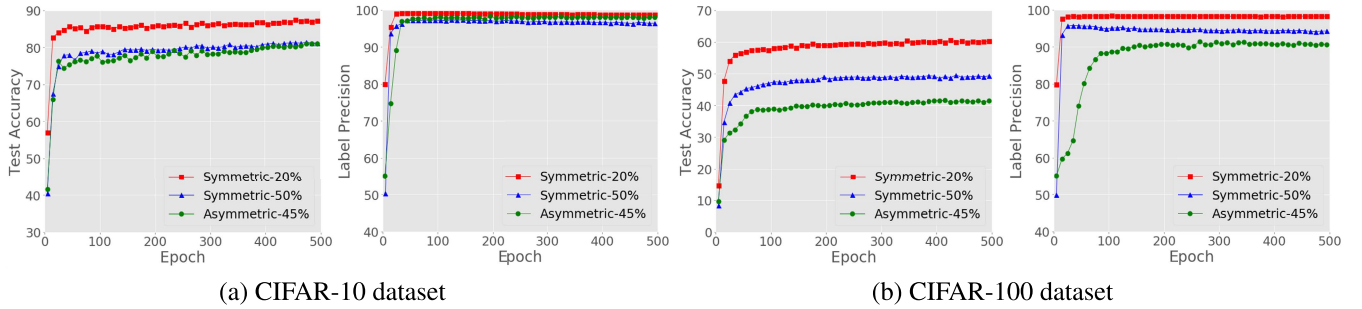
Table 5 describes the datasets in the experiments in details. We manually injected label noise into CIFAR-10/100 datasets using the noise transition matrix, as shown in Fig.5.

5) COMPARED METHODS

We compared our method against recent state-of-the-art single network- and two network-based methods. For single network-based methods, we compared our method against

TABLE 6. Average test accuracy (%) on the CIFAR-100 dataset over last 10 epochs with various noise corruptions. (a) Single network-method results. (b) Two network-method results. The symbol * indicates scores provided by the corresponding authors. The best results are boldfaced.

	Methods	Symmetric 20%	Symmetric 50%	Asymmetric 45%
(a)	Standard	40.44 ± .36	21.34 ± .27	31.85 ± .85
	GCE	51.86 ± .09	37.60 ± .08	33.13 ± .14
	SL	56.47 ± .07	32.70 ± .13	38.46 ± .09
	NPCL*	55.30 ± .09	42.56 ± .06	—
(b)	MentorNet	52.13 ± .40	39.00 ± 1.00	31.60 ± .51
	Co-teaching	53.89 ± .09	34.96 ± .50	34.55 ± .12
	Co-teaching ⁺	56.15 ± .09	37.88 ± .06	32.88 ± .25
	JoCoR	53.01 ± .04	43.49 ± .46	26.51 ± .32
	DivideMix	—	49.41 ± .25	34.24 ± .63
	Ours	60.32 ± .05	49.52 ± .08	41.59 ± .03

**FIGURE 6.** Test accuracy and label prediction of the proposed method according to noise levels.

(i) a standard deep network that is directly trained on noisy datasets with a cross-entropy loss, (ii) GCE [27], which uses the advantages of both MAE [28] and cross-entropy losses, (iii) SL [19], which combines a reversed cross-entropy loss with a standard cross-entropy loss, and (iv) NPCL [29], which extracts true-labeled samples from a noisy dataset with a manually specified threshold. For two network-based methods, we compared our method against (v) MentorNet [5], which uses an extra pre-trained network to select small-loss samples, (vi) Co-teaching [1], which develops two interactive networks to be robust to noisy labels, (vii) Co-teaching⁺ [9], which improves two networks using disagreement- and cross-update steps, (viii) JoCoR [8], which employs a joint-loss to select small-loss samples, and (ix) DivideMix [6] that maintains two networks to select an unlabeled set of each other.

B. COMPARISON WITH STATE-OF-THE-ART METHODS

1) COMPARISONS USING CIFAR-10

Table 4 shows the test accuracy of the compared methods on the CIFAR-10 dataset. As shown in the table, our method exhibited state-of-the-art results for all cases. In particular, our method was significantly superior to other methods in the Asymmetric 45% case. As shown in Fig.6, our method solved over-parameterization problems and its test accuracy did not decrease even after 500 epochs. The label precision of our method was close to 100% in all three cases, which means that our selection strategy accurately selected clean sample data, and the method was robust to severely noisy labels.

2) COMPARISONS USING CIFAR-100

Table 6 shows the test accuracy on the CIFAR-100 dataset. In the Symmetric 50% case, our method and DivideMix considerably outperformed other baselines. In the Asymmetric 45%, our method significantly outperformed other methods. It should be noted that our method uses a single network and enables end-to-end training with noisy data.

3) COMPARISONS USING CLOTHING1M

Table 7 shows the test accuracy of compared methods on real-world noisy labels using the Clothing1M dataset. Our method outperformed single network-based methods in Table 7(a), whereas it is comparable to two network-based methods in Table 7(b).

4) COMPARISONS ON COMPUTATIONAL COMPLEXITY

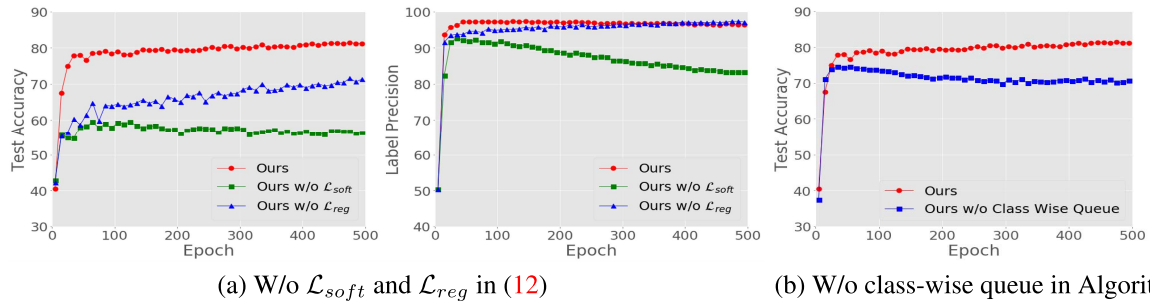
Because of Co-teaching, Co-teaching⁺, JoCoR, and DivideMix require additional networks, the number of network learning parameters is approximately twice than those of standard network, as shown in Table 8(b). Our method can effectively prevent over-parameterization only with a half of parameters compared with two network-based methods. Table 9 reports the network training time of Co-teaching, Divide Mix, and the proposed method on cifar10 datasets.

C. ABLATION STUDY

To verify the effectiveness of our method and provide an in-depth analysis, we evaluated the method in a component-wise manner (*i.e.*, w/o regularization loss for soft mixture

TABLE 7. Test accuracy (%) on the Clothing1M dataset. (a) Single network-method results. (b) Two network-method results. The best results are boldfaced, whereas blue indicates the second best method.

	Methods	Test Accuracy (%)
(a)	Standard	68.86
	GCE	69.0
	Joint Optimization [20]	72.23
	SL	71.02
(b)	Co-teaching	69.21
	Co-teaching ⁺	59.46
	JoCoR	70.30
	DivideMix	74.76
	Ours	72.59

**FIGURE 7.** Ablation study on the proposed method using the CIFAR-10 dataset.**TABLE 8.** The number of learning parameters of baselines on the CIFAR-10/100 dataset. (a) Single network based-methods. (b) Two network based-methods.

Methods	(a)	(b)	Ours
# of Parameters	$\approx 4.43(M)$	$\approx 8.86(M)$	$\approx 4.43(M)$

TABLE 9. The training time (sec) of baselines on the CIFAR-10 dataset.

Method	Standard	Co-teaching	DivideMix	Ours
Time(sec)	11.4	16.02	35.1	17.2

probability, $\mathcal{L}_{soft}$, in (12), w/o regularization loss for augmented samples, $\mathcal{L}_{reg}$, in (12), and moving weighted moment queue in Algorithm 1). For this experiment, we used the CIFAR-10 dataset with 50% symmetric noise. To analyze the contribution of $\mathcal{L}_{soft}$, we trained our network with $\omega = 0$ in (12). As shown in the green curves in Fig.7(a), without $\mathcal{L}_{soft}$, the test accuracy and label precision decreased as the epochs increased, which indicates that regularization for soft mixture probability is crucial for selecting small-loss samples. As shown in the blue curves in Fig.7(a), owing to the network memorization effect, the test accuracy gradually increased even without $\mathcal{L}_{reg}$. However, the accuracy was considerably lower than that of our method with $\mathcal{L}_{reg}$. Fig.7(b) presents the contribution of the class-wise queues in Algorithm 1. As shown in the figure, by computing Gaussian statistics using queues (red), the proposed method produced accurate results, whereas our method without queues induced over-parameterization (blue).

VI. DISCUSSION

The small loss selection is a widely adopted strategy for noisy labels. Specifically, it assumes that the sample with a small loss value potentially has a clean label. So, it trains the network only with these small loss samples while discarding samples with large loss values. In this case, the critical point is how to select small loss samples. If we select too small samples, the network cannot learn enough data representation because we discard many essential representations. In contrast, if we select too large samples, the network is affected by noisy label samples, so the generalization performance is degraded. The previous method's pre-designed selection strategy cannot flexibly adapt to the network's learning dynamics.

To remedy the above issues, we propose two crucial concepts "self-augmentation from robust GMM" and "data-adaptive selection strategy." First, we assume that data samples in the pre-softmax layer follow a class-conditional Gaussian distribution, which is a usual assumption. Then, we obtain class-wise Gaussian parameters by "Moving weighted moment queue" from the sample we can expect as clean samples with high probability. After we collect enough samples in this queue, we construct GMM from this class-conditional distribution. Then, we sample the augmented sample Z from GMM. However, as training proceeds, Gaussian parameters are contaminated by noise labels if we do not use a carefully designed selection strategy. So, we propose the "data-driven selection strategy" derived from the second moment of Gaussian (covariance matrix).

VII. CONCLUSION

In this study, we induced accurate small-loss samples based on the proposed class-conditional GMM without additional classifiers and parameters. Because small-loss samples inevitably contain inconsistent (noisy) label information, we regularized Gaussian distributions with moving weighted Gaussian moments. In addition, we proposed a novel data-adaptive strategy to enhance the quality of small-loss samples. Our self-augmentation using the noisy-robust GMM effectively prevented over-parameterization of the network. The experimental results indicate that our method outperforms state-of-the-art methods on several datasets. Since our method relies on mixture distributions, generated samples from GMM cannot be directly used for back-propagation. It restricts our method to merely regularization. To be more powerful, the reparameterization trick for mixture sampling might be helpful for directly back-propagating through Gaussian parameters. Moreover, for future work, theoretical analysis of the proposed method on sub-Gaussianity may bring an in-depth analysis of generalization bound in the noisy label situation.

REFERENCES

- [1] B. Han, Q. Yao, X. Yu, G. Niu, M. Xu, W. Hu, I. W. Tsang, and M. Sugiyama, "Co-teaching: Robust training of deep neural networks with extremely noisy labels," in *Proc. NIPS*, 2018, pp. 1–11.
- [2] L. Van der Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, no. 11, pp. 1–27, 2008.
- [3] X. Yu, T. Liu, M. Gong, and D. Tao, "Learning with biased complementary labels," in *Proc. ECCV*, 2018, pp. 68–83.
- [4] A. Blum, A. Kalai, and H. Wasserman, "Noise-tolerant learning, the parity problem, and the statistical query model," *J. ACM*, vol. 50, no. 4, pp. 506–519, Jul. 2003.
- [5] L. Jiang, Z. Zhou, T. Leung, L.-J. Li, and L. Fei-Fei, "MentorNet: Regularizing very deep neural networks on corrupted labels," in *Proc. ICML*, 2018, pp. 2304–2313.
- [6] J. Li, R. Socher, and S. C. Hoi, "DivideMix: Learning with noisy labels as semi-supervised learning," in *Proc. ICLR*, 2020, pp. 1–14.
- [7] H. Song, M. Kim, and J.-G. Lee, "Selfie: Refurbishing unclean samples for robust deep learning," in *Proc. ICML*, 2019, pp. 5907–5915.
- [8] H. Wei, L. Feng, X. Chen, and B. An, "Combating noisy labels by agreement: A joint training method with co-regularization," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 13726–13735.
- [9] X. Yu, B. Han, J. Yao, G. Niu, I. Tsang, and M. Sugiyama, "How does disagreement help generalization against label corruption?" in *Proc. ICML*, 2019, pp. 7164–7173.
- [10] D. Arpit, S. Jastrzebski, N. Ballas, D. Krueger, E. Bengio, M. S. Kanwal, T. Maharaj, A. Fischer, and A. Courville, "A closer look at memorization in deep networks," in *Proc. ICML*, 2017, pp. 233–242.
- [11] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning requires rethinking generalization," in *Proc. ICLR*, 2017, pp. 1–15.
- [12] Q. Yao, H. Yang, B. Han, G. Niu, and J. T. Kwok, "Searching to exploit memorization effect in learning with noisy labels," in *Proc. ICML*, 2020, pp. 10789–10798.
- [13] K. Lee, S. Yun, K. Lee, H. Lee, B. Li, and J. Shin, "Robust inference via generative classifiers for handling noisy labels," in *Proc. ICML*, 2019, pp. 3763–3772.
- [14] G. Patrini, A. Rozza, A. K. Menon, R. Nock, and L. Qu, "Making deep neural networks robust to label noise: A loss correction approach," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1944–1952.
- [15] D. Hendrycks, M. Mazeika, D. Wilson, and K. Gimpel, "Using trusted data to train deep networks on labels corrupted by severe noise," in *Proc. NIPS*, 2018, pp. 1–10.
- [16] M. Ren, W. Zeng, B. Yang, and R. Urtasun, "Learning to reweight examples for robust deep learning," in *Proc. ICML*, 2018, pp. 4334–4343.
- [17] S. Reed, H. Lee, D. Anguelov, C. Szegedy, D. Erhan, and A. Rabinovich, "Training deep neural networks on noisy labels with bootstrapping," in *Proc. NIPS*, 2015, pp. 1–11.
- [18] E. Arazo, D. Ortego, P. Albert, N. E. O'Connor, and K. McGuinness, "Unsupervised label noise modeling and loss correction," in *Proc. ICML*, 2019, p. 312.
- [19] Y. Wang, X. Ma, Z. Chen, Y. Luo, J. Yi, and J. Bailey, "Symmetric cross entropy for robust learning with noisy labels," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 322–330.
- [20] D. Tanaka, D. Ikami, T. Yamasaki, and K. Aizawa, "Joint optimization framework for learning with noisy labels," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 5552–5560.
- [21] B. Zong, Q. Song, M. R. Min, W. Cheng, C. Lumezanu, D. Cho, and H. Chen, "Deep autoencoding Gaussian mixture model for unsupervised anomaly detection," in *Proc. ICLR*, 2018, pp. 1–19.
- [22] K. Yi and J. Wu, "Probabilistic end-to-end noise correction for learning with noisy labels," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 7017–7025.
- [23] T. Xiao, T. Xia, Y. Yang, C. Huang, and X. Wang, "Learning from massive noisy labeled data for image classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 2691–2699.
- [24] J. Han, P. Luo, and X. Wang, "Deep self-learning from noisy labels," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 5138–5147.
- [25] Y. Sun, Y. Tian, Y. Xu, and J. Li, "Limited gradient descent: Learning with noisy labels," 2018, *arXiv:1811.08117*.
- [26] B. V. Rooyen, A. Menon, and R. C. Williamson, "Learning with symmetric label noise: The importance of being unhinged," in *Proc. NIPS*, 2015, pp. 10–18.
- [27] Z. Zhang and M. Sabuncu, "Generalized cross entropy loss for training deep neural networks with noisy labels," in *Proc. NIPS*, 2018, pp. 1–11.
- [28] A. Ghosh, H. Kumar, and P. S. Sastry, "Robust loss functions under label noise for deep neural networks," in *Proc. AAAI*, 2017, pp. 1–7.
- [29] Y. Lyu and I. W. Tsang, "Curriculum loss: Robust learning and generalization against label corruption," in *Proc. ICLR*, 2020, pp. 1–22.



BYOUNG WOO PARK received the B.S. degree in statistics from Inha University, Seoul, South Korea, in 2021. He is currently pursuing the M.S. degree in artificial intelligence with Chung-Ang University. His research interests include generative model, time series model, and stochastic differential equation based deep neural networks.



SUNG WOO PARK received the B.S. degree in electrical engineering and the M.S. degree in computer science from Chung-Ang University (CAU), Seoul, South Korea, in 2018 and 2020, respectively, where he is currently pursuing the Ph.D. degree in computer science. His research interests include generative model, graph model, and reliability of neural networks.



JUNSEOK KWON (Member, IEEE) received the B.Sc. and M.Sc. degrees in the topic of object tracking (supervised by Prof. Kyoung Mu Lee) and the Ph.D. degree in electrical engineering and computer science from the Seoul National University, South Korea, in 2006, 2008, and 2013, respectively. He was a Postdoctoral Researcher under Prof. Luc Van Gool at the Computer Vision Laboratory, ETH Zurich, from 2013 to 2014. He is an Associate Professor with the School of Computer Science and Engineering, Chung-Ang University, Seoul, South Korea. He is working in the field of object tracking to capture the dynamics of objects. His research interests include visual tracking, visual surveillance, and Monte Carlo sampling method and its variants.

...