Knowledge Distillation Meets Label Noise Learning: Ambiguity-Guided Mutual Label Refinery

Runqing Jiang, Yan Yan, *Senior Member, IEEE,* Jing-Hao Xue, *Senior Member, IEEE,* Si Chen, *Member, IEEE,* Nannan Wang, *Member, IEEE,* and Hanzi Wang, *Senior Member, IEEE*

***Abstract*—Knowledge Distillation (KD), which aims at transferring the knowledge from a complex network (a teacher) to a simpler and smaller network (a student), has received considerable attention in recent years. Typically, most existing KD methods work on well-labeled data. Unfortunately, real-world data often inevitably involve noisy labels, thus leading to performance deterioration of these methods. In this paper, we study a little-explored but important issue, i.e., *KD with noisy labels*. To this end, we propose a novel KD method, called Ambiguity-guided Mutual Label refinery KD (AML-KD), to train the student model in the presence of noisy labels. Specifically, based on the pretrained teacher model, a two-stage label refinery framework is innovatively introduced to refine labels gradually. In the first stage, we perform label propagation with small-loss selection guided by the teacher model, improving the learning capability of the student model. In the second stage, we perform mutual label propagation between the teacher and student models in a mutual-benefit way. During the label refinery, an Ambiguity-aware Weight Estimation (AWE) module is developed to address the problem of ambiguous samples, avoiding overfitting these samples. One distinct advantage of AML-KD is that it is capable of learning a high-accuracy and low-cost student model with label noise. Experimental results on synthetic and real-world noisy datasets show the effectiveness of our AML-KD against state-of-the-art KD methods and label noise learning methods. Code is available at https://github.com/Runqing-forMost/AML-KD.**

***Index Terms*—Knowledge distillation, Label noise learning, Label refinery, Label propagation**

I. Introduction

OVER the past few years, a large number of deep model compression methods [1] have been developed to reduce the size of deep Convolutional Neural Networks (CNN) without greatly affecting the accuracy. Among these methods, one line of research works on Knowledge Distillation (KD), whose goal is to transfer the knowledge from a larger teacher model to a smaller student model with similar accuracy. Recent efforts on KD mainly target at exploiting more dedicated knowledge, such as minimizing the discrepancy of intermediate representations between the teacher and student models [4], distilling relationships between samples [5], and adopting auxiliary tasks as extra knowledge [6].

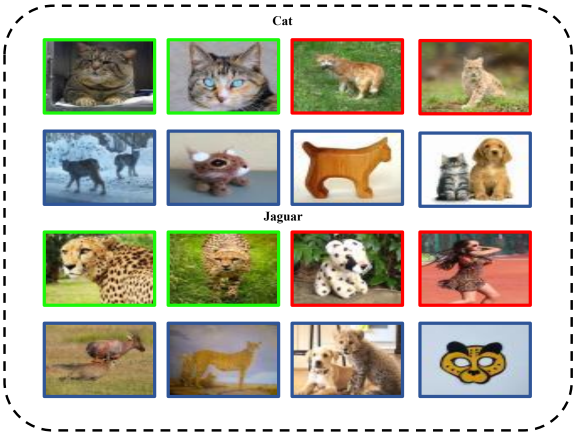


Fig. 1: Images from the Animal-10N dataset [7] with the label “cat” and the label “Jaguar”. Images in green boxes, red boxes, and blue boxes represent the clean samples, noisy labeled samples, and ambiguous samples, respectively.

Most KD methods require well-labeled data during the distillation procedure. However, collecting large-scale data with fully accurate annotations is labor-intensive and time-consuming, which inevitably introduces noisy labels. As shown in Fig. 1, large-scale real-world image datasets are usually contaminated with noisy labels. In particular, there exists a considerable portion of ambiguous samples in these datasets. As a consequence, when trained on noisy datasets, KD methods are prone to overfit these samples, leading to significant performance deterioration. Therefore, it is still an open question *how to learn an effective and compact student model in the presence of noisy labels.*

Recently, Label Noise Learning (LNL) methods [8]–[10], [12], [13], which learn robust deep CNN models from noisy data, have made remarkable progress. Some methods design robust loss functions (such as SCE [8]) to alleviate the overfitting problem, while other methods (such as DivideMix [9] and JoCoR [10]) explicitly select potentially clean samples from all the samples to achieve the robustness. Unfortunately, when small CNN models are individually served as backbones, the performance of these methods on large-scale datasets is seriously degraded. This is mainly because of the difficulty of identifying and refining noisy labels by small CNN models

R. Jiang, Y. Yan, H. Wang are with the Fujian Key Laboratory of Sensing and Computing for Smart City, School of Informatics, Xiamen University, Xiamen 361005, China and the State Key Laboratory of Integrated Services Networks (Xidian University), Xi’an 710071, China (e-mail: jiangrunqing@stu.xmu.edu.cn; yanyan@xmu.edu.cn; hanzi.wang@xmu.edu.cn).

J.-H. Xue is with the Department of Statistical Science, University College London, London WC1E 6BT, UK (e-mail: jinghao.xue@ucl.ac.uk).

S. Chen is with the School of Computer and Information Engineering, Xiamen University of Technology, Xiamen 361024, China (e-mail: chensi@xmut.edu.cn).

N. Wang is with the State Key Laboratory of Integrated Services Networks (Xidian University), Xi’an 710071, China (e-mail: nnwang@xidian.edu.cn).

whose learning capability is inferior.

To address the above problems, we leverage a stronger-performing pretrained teacher model to guide the label refinery process of a smaller student model. Specifically, we propose a novel KD method, called Ambiguity-guided Mutual Label refinery KD (AML-KD), for learning with noisy labels. Based on the pretrained teacher model, AML-KD involves a two-stage label refinery framework (i.e., label propagation with small-loss selection in the first stage and mutual label propagation in the second stage) to progressively improve the label quality for training the student model. During the label refinery, we elaborately design an Ambiguity-aware Weight Estimation (AWE) module to alleviate the negative influence of ambiguous samples. Based on the above elaborate designs, our method is able to learn a high-performance and low-cost student model with noisy labels.

In summary, the main contributions of this paper are:

- We propose a novel AML-KD to successfully transfer the knowledge from a teacher model to a student model under label noise. In AML-KD, we develop a two-stage label refinery framework, which takes advantage of feature representations from both the teacher and student models to refine noisy labels gradually. Therefore, a high-accuracy student model with small memory consumption can be learned from noisy data.
- We design an AWE module to alleviate the problem of ambiguous samples by exploiting the feature distribution scores and the refined annotations, largely preventing the model from overfitting these samples and effectively improving the label accuracy.
- Extensive experiments on synthetic and real-world noisy datasets validate the superiority of our AML-KD method against several state-of-the-art KD methods and label noise learning methods.

The remainder of this paper is organized as follows. First, we briefly introduce the related work in Section II. Then, we present the details of our proposed AML-KD method in Section III. Next, we compare our AML-KD with several state-of-the-art methods in Section IV. Finally, we draw the conclusion in Section V.

II. Related Work

In this section, we mainly review some related work, including Knowledge Distillation (KD) and Label Noise Learning (LNL).

A. Knowledge Distillation (KD)

In KD, a student model is often trained by using the supervision signals from both the ground-truth and a teacher model. Hinton *et al.* [17] propose to leverage the Kullback-Leibler (KL) divergence to minimize the probability distributions between the teacher and student models. Early KD methods obtain the student model based on the final outputs of the teacher model, while recent works attempt to exploit the rich information from different layers of the teacher model. For instance, FitNet [4] suggests that the performance of a student model can be improved by imitating the intermediate feature representations of a teacher model via a Mean Square Error (MSE) loss. Later, Sun *et al.* [18] generalize FitNet by reducing the MSE loss between each individual layer of the student and teacher models. Zagoruyko *et al.* [19] propose Attention Transfer (AT) to transfer spatial attention from the teacher model to the student model. Liu *et al.* [14] propose to distill the knowledge hidden in the inter-channel correlations of the teacher model. Such a way is helpful for aligning features between the teacher and student models. Lin *et al.* [15] introduce a one-to-all spatial matching knowledge distillation method to combat the semantic information inconsistency caused by architecture differences. Li *et al.* [16] distill the neural architecture knowledge from the teacher model to facilitate the neural architecture search of the student model.

The above methods distill the knowledge according to feature representations extracted from the teacher model. However, they might ignore the underlying relations between samples. To overcome this problem, similarity-preserving KD [20] is developed to transfer the knowledge modeled by pair-wise similarity. Some work [6] combines KD with Self-Supervised Learning (SSL) due to the powerful representation capability of SSL. Contrastive Representation Distillation (CRD) [21] introduces contrastive learning into KD, and it maximizes the mutual information between the teacher and student representations.

To date, very few KD methods are developed to address the label noise problem. The work most relevant to ours is FNKD [22], where a sample adaptive feature normalization method is proposed to specifically alleviate the negative impact of label encoding noise. Unfortunately, FNKD cannot deal with other types of label noise. In this paper, we develop a robust KD method, which achieves good accuracy even at a high noise rate and effectively handles different types of label noise, including real-world label noise.

B. Label Noise Learning (LNL)

Recently, LNL has attracted considerable attention since labels are often noisy and imperfect in real-world scenarios. In this subsection, we review representative LNL methods, which are mostly related to our work.

1) Sample Weighting: Sample weighting-based methods aim to assign small and large weights to the samples with noisy and clean labels, respectively. For example, Guo *et al.* [23] introduce a curriculum learning-based method, which assigns a weight to each sample according to the unsupervised estimation of data complexity. Harutyunyan *et al.* [24] compute the weights based on the gradients of final layers without relying on the label information. Ma *et al.* [25] develop a self-reweighting strategy by assigning sample weights based on the similarities between the samples and the learned class centroids.

2) Sample Selection: Sample selection-based methods select potentially clean samples from noisy datasets. Arpit *et al.* [26] reveal the memorization effect. Based on this, the small-loss criterion is proposed to select samples with small cross-entropy losses as clean ones. Han *et al.* [27] develop the Co-teaching method, which trains two models simultaneously.

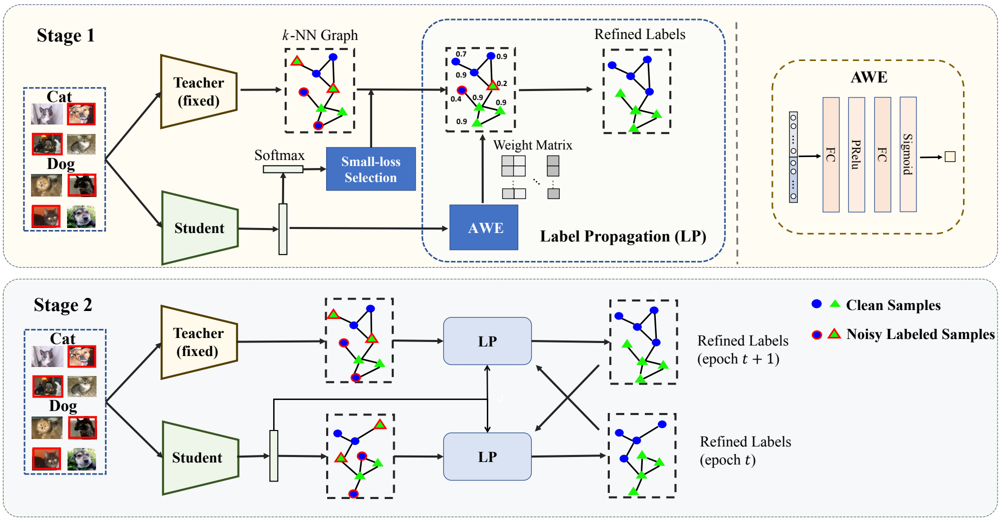


Fig. 2: Overview of our proposed two-stage label refinery framework. We take a two-class classification (cat and dog) task as an example, where the circle or triangle enclosed in red indicates a noisy labeled sample. The upper panel denotes the LP with small-loss selection stage while the lower panel shows the mutual LP stage.

Different from Co-teaching, FINE [28] introduces eigenvectors to select clean samples from the noisy dataset. Jiang *et al.* [29] propose a Two-Stream Sample Selection Network (TS^3-Net) to effectively train a sparse neural network on the noisy dataset. Note that TS^3-Net works on unstructured pruning, which removes unimportant weights. Hence, it is difficult to achieve actual acceleration in real-world implementations. Wei *et al.* [10] propose a robust learning paradigm called JoCoR to minimize the diversity of two models during training. Karim *et al.* [11] introduce a Jensen-Shannon divergence based uniform selection mechanism and contrastive learning to address high label noise.

3) Graph-Based Methods: Recently, graph-based methods have shown the effectiveness of combating label noise. Zhang *et al.* [30] propose DualGraph to capture structural relations between labels by using instance-level and distribution-level relations. Wu *et al.* [31] develop the Noise Graph Correction (NGC) method, where Label Propagation (LP) [32] is used to generate pseudo-labels by propagating labels along high-density areas.

The above LNL methods often work on complicated CNN models for their superior learning capability. Generally, the performance of these methods significantly drops on large-scale noisy datasets, when they are trained with small CNN models as backbones. Unlike these methods, we design a robust KD method that aims to learn a high-accuracy small model from the large-scale noisy dataset.

Recently, Li *et al.* [12] also propose to learn from noisy labels with distillation. However, we emphasize that the motivation, problem setting, and methodology of our method are intrinsically different from those of Li *et al.*'s method [12]. On the one hand, Li *et al.*'s method leverages the knowledge graph to obtain a deep model (rather than a small model) from the noisy dataset, while our developed method introduces a two-stage label refinery framework to obtain a compact model in the presence of noisy labels. On the other hand, Li *et al.*'s method requires extra clean samples and side information. In contrast, our method can directly work on the noisy dataset without the need of collecting clean samples.

III. Methodology

In this section, we first give the problem formulation in Section III-A. Then, we present an overview of the proposed method in Section III-B. Next, we describe the key components of our method in Sections III-C and III-D. Finally, we show the overall training loss in Section III-E.

A. Problem Formulation

Notations Given that we have a training set $\mathcal{D}_{\mathfrak{t}}$ for the teacher model (denoted as $\mathcal{M}_{\mathfrak{t}}$) and a training set $\mathcal{D}_{\mathfrak{s}} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i^{(0)})\}_{i=1}^{N}$ with noisy labels for the student model (denoted as $\mathcal{M}_{\mathfrak{s}}$). Here, $\boldsymbol{x}_i$ and $\boldsymbol{y}_i^{(0)} \in \mathbb{R}^C$ denote the i-th training sample and its corresponding initial label annotation, respectively. N and C are the number of training samples and the number of classes, respectively. Let $\boldsymbol{Y}^{(0)} = [\boldsymbol{y}_1^{(0)}, \cdots, \boldsymbol{y}_N^{(0)}] \in \mathbb{R}^{C\times N}$ denote the initial noise-corrupted label matrix. Given an input $\boldsymbol{x}_j$, we denote the feature representations extracted by $\mathcal{M}_{\mathfrak{s}}$ and $\mathcal{M}_{\mathfrak{t}}$ as $\phi_{\mathfrak{s}}(\boldsymbol{x}_j)$ and $\phi_{\mathfrak{t}}(\boldsymbol{x}_j)$, respectively. For clarity, all the vectors are normalized.

We follow the conventional learning scheme of offline distillation [17], which first pretrains a teacher model and then trains a student model under the guidance of the teacher model. Moreover, we assume that the teacher model can be pretrained with a dedicated LNL method or clean samples. Hence, we target at *obtaining a high-accuracy and low-cost student model with noisy labeled samples*. Such a problem setting is a practical setup, which investigates the ability to learn a compact model supervised by a pretrained teacher model in the case of label noise.

B. Overview

A straightforward way to address KD with label noise is to leverage a trained teacher model (via an LNL method) to relabel the annotations, based on which we can apply KD to learn a student model. However, such a way relies heavily on relabeled annotations (still containing noisy labels) by the teacher model and does not fully exploit the student model for relabeling. A more desirable way is to make use of both the teacher and student models to refine the labels. Note that the learning capability of the student model is weak at the early training stage, especially in the case of noisy labels. Therefore, we develop an innovative method to gradually perform label refinery by first enhancing the learning capability of the student model and then taking advantage of the teacher and student models in two distinguished views, alleviating error accumulations and improving the label accuracy.

Specifically, we propose an AML-KD method, which distills the knowledge from the teacher model into the student model under noisy labels. The training process of AML-KD involves two periods. In the first period, a teacher model is pretrained with $\mathcal{D}_{\mathbb{t}}$ (a dedicated LNL method can be used when the annotations of $\mathcal{D}_{\mathbb{t}}$ are noisy). In the second period, based on the pretrained teacher model, a student model is trained with $\mathcal{D}_{\mathbb{s}}$. In this period, we develop a two-stage label refinery framework to progressively purify noisy labels, as illustrated in Fig. 2.

1) Label propagation (LP) with small-loss selection: For the first stage, a k-NN graph is constructed, where each vertex denotes a sample (corresponding to the feature extracted by $\mathcal{M}_{\mathbb{t}}$) in $\mathcal{D}_{\mathbb{s}}$ and each edge represents the similarity between two vertices. Meanwhile, we relabel the initial annotations according to the small-loss criterion. Based on the k-NN graph and relabeled annotations, we perform LP to refine the labels. After the first stage, both the label quality of $\mathcal{D}_{\mathbb{s}}$ and the learning capability of $\mathcal{M}_{\mathbb{s}}$ are greatly improved. Thus, we can employ $\mathcal{M}_{\mathbb{t}}$ and $\mathcal{M}_{\mathbb{s}}$ to perform LP alternately in the next stage.

2) Mutual label propagation: For the second stage, we perform label refinery in two steps: (i) performing LP with the k-NN graph constructed by $\mathcal{M}_{\mathbb{s}}$ and the refined labels (obtained at epoch $t-1$) at epoch t; (ii) performing LP with the k-NN graph constructed by $\mathcal{M}_{\mathbb{t}}$ and the refined labels (obtained at epoch t) at epoch $t+1$. The above two steps are optimized in an alternate way. In this way, the labels are gradually refined with the guidance of both the teacher and student models.

During the label refinery, to alleviate the influence of ambiguous samples (i.e., incorrectly relabeled samples after LP), an AWE module is introduced to estimate an ambiguity weight for each sample, thereby leading to refined label quality and boosting the model training capacity.

C. Two-Stage Label Refinery Framework

High-quality labels are of great importance to ensure the performance of CNN models. To obtain high-quality labels, we perform two-stage label refinery.

Stage 1. First, an undirected graph $G = \langle V, E\rangle$ is introduced to model the relationships between samples. Here, V and E represent the sets of graph vertices and edges, respectively. In graph G, the affinity between vertices is modeled by a similarity matrix $\boldsymbol{A}_{\mathbb{t}} \in \mathbb{R}^{N\times N}$, which is

$$\boldsymbol{A}_{\mathbb{t}}[i,j] = \begin{cases} \phi_{\mathbb{t}}(\boldsymbol{x}_i)^{\mathrm{T}}\phi_{\mathbb{t}}(\boldsymbol{x}_j), & \text{if } i \neq j \text{ \& } \boldsymbol{x}_j \in \mathbf{NN}_k(\boldsymbol{x}_i) \\ 0 & \text{otherwise,} \end{cases} \tag{1}$$

where $\mathbf{NN}_k(\cdot)$ represents the k nearest neighbors. Initially, the teacher model $\mathcal{M}_{\mathbb{t}}$ has stronger learning capability than the student model $\mathcal{M}_{\mathbb{s}}$. Hence, the similarity matrix is constructed based on the features extracted by $\mathcal{M}_{\mathbb{t}}$.

Similar to [32], we obtain an $N \times N$ symmetric nonnegative adjacency matrix with zero diagonal $\boldsymbol{W}_{\mathbb{t}} = \boldsymbol{A}_{\mathbb{t}} + \boldsymbol{A}_{\mathbb{t}}^{\mathrm{T}}$. Then, $\boldsymbol{W}_{\mathbb{t}}$ is normalized to obtain an $N \times N$ matrix $\tilde{\boldsymbol{W}}_{\mathbb{t}} = \boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{W}_{\mathbb{t}}\boldsymbol{D}^{-\frac{1}{2}}$. Here, the matrix $\boldsymbol{D} \in \mathbb{R}^{N\times N}$ is the degree matrix defined as $\boldsymbol{D} = \mathrm{diag}(\boldsymbol{W}_{\mathbb{t}}\mathbf{1}_N)$, where $\mathbf{1}_N \in \mathbb{R}^N$ is an all-ones vector and $\mathrm{diag}(.)$ denotes a diagonal matrix whose diagonal is the input vector.

Meanwhile, based on the small-loss criterion (i.e., samples with small training losses are likely to be clean samples according to the memorization effect) [26], the labels of clean samples remain unchanged while those of noisy labeled samples are relabeled according to the outputs of $\mathcal{M}_{\mathbb{s}}$. Mathematically, we relabel the sample $\boldsymbol{x}_i$ at epoch t by

$$\boldsymbol{z}_i^{(t)}[c] = \begin{cases} \boldsymbol{y}_i^{(0)}[c], & \text{if } \boldsymbol{x}_i \in \mathcal{S}_{clean}^{(t)} \\ \mathbb{1}\{c = \arg\max\limits_j(\boldsymbol{p}_{\mathbb{s}}^{(t-1)}(\boldsymbol{x}_i)[j])\} & \text{otherwise,} \end{cases} \tag{2}$$

where $\boldsymbol{z}_i^{(t)}[c]$ denotes the c-th element of $\boldsymbol{z}_i^{(t)} \in \mathbb{R}^C$, which is the refined label corresponding to $\boldsymbol{x}_i$ at epoch t. $\mathcal{S}_{clean}^{(t)} = \arg\min_{\mathcal{S}':|\mathcal{S}'|\geq(1-r)|\mathcal{D}_{\mathbb{s}}|} \mathcal{L}_{ce}(\mathcal{M}_{\mathbb{s}}, \mathcal{S}')$ denotes the clean set selected by $\mathcal{M}_{\mathbb{s}}$ with the Cross-Entropy (CE) loss at epoch t. $|\mathcal{S}'|$ and $|\mathcal{D}_{\mathbb{s}}|$ denote the sizes of $\mathcal{S}'$ and $\mathcal{D}_{\mathbb{s}}$, respectively. r is the noise rate. $\boldsymbol{p}_{\mathbb{s}}^{(t-1)}(\boldsymbol{x}_i)[j]$ represents the j-th element of the predicted probability $\boldsymbol{p}_{\mathbb{s}}^{(t-1)}(\boldsymbol{x}_i)$ given by $\mathcal{M}_{\mathbb{s}}$ at epoch $t-1$. The indicator function $\mathbb{1}$ takes on the value 1 if its argument is true, and 0 otherwise.

The above relabeling process only relies on the prediction of the student model $\mathcal{M}_{\mathbb{s}}$, which may limit the relabeling accuracy. To deal with this problem, we further exploit the intrinsic neighborhood structure of training samples from the perspective of the teacher model $\mathcal{M}_{\mathbb{t}}$. To this end, motivated by the manifold assumption that similar samples should give the same predictions [34], we propose to leverage LP to refine the label matrix on graph G. Specifically, LP can be formulated as

$$\boldsymbol{Y}^{(t)} = \mathrm{LP}(\boldsymbol{Z}^{(t)}, \tilde{\boldsymbol{W}}_{\mathbb{t}}, \boldsymbol{Q}_{\mathbb{s}}^{(t-1)}), \tag{3}$$

where $\boldsymbol{Y}^{(t)} = [\boldsymbol{y}_1^{(t)}, \cdots, \boldsymbol{y}_N^{(t)}] \in \mathbb{R}^{C\times N}$ and $\boldsymbol{Z}^{(t)} = [\boldsymbol{z}_1^{(t)}, \cdots, \boldsymbol{z}_N^{(t)}] \in \mathbb{R}^{C\times N}$ represent the refined label matrices given by LP and small-loss selection at epoch t, respectively. $\mathrm{LP}(\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}) = \arg\min_{\boldsymbol{K}} \beta/2 \sum_{i,j=1}^{N} \boldsymbol{B}_{ij}||\boldsymbol{k}_i/\sqrt{d_{ii}} - \boldsymbol{k}_j/\sqrt{d_{jj}}||^2 + (1-\beta)||\boldsymbol{A} \odot \boldsymbol{C} - \boldsymbol{K}||_F$, where $\boldsymbol{K} = [\boldsymbol{k}_1, \ldots, \boldsymbol{k}_N] \in \mathbb{R}^{C\times N}$ is the refined soft labels. Note that, after obtaining the refined soft labels from LP, we transform them into hard labels, as done in previous methods [31], [32].

$\boldsymbol{Q}_{\mathbb{s}}^{(t-1)}$ denotes the weight matrix (see Section 3.4) at epoch $t-1$. d_{ii} is the i-th diagonal diagonal element of $\boldsymbol{D}$ mentioned previously. $||.||_F$ is the Frobenius norm and $\odot$ represents the element-wise product between two matrices. β is a regularized parameter.

Stage 2. After the first stage, the learning capability of $\mathcal{M}_{\mathbb{s}}$ is greatly improved. To fully exploit the feature information from both $\mathcal{M}_{\mathbb{s}}$ and $\mathcal{M}_{\mathbb{t}}$, we further take advantage of mutual learning to refine labels. During this stage, the student model and the teacher model perform mutual LP. Technically, based on the k-NN graph constructed by $\mathcal{M}_{\mathbb{s}}$ and the refined labels (obtained at epoch $t-1$), we perform LP to improve the label quality at epoch t. Then, we construct the k-NN graph by $\mathcal{M}_{\mathbb{t}}$ and the refined labels (obtained at epoch t) to perform LP at epoch $t+1$. The above procedure is formulated as

$$\boldsymbol{Y}^{(t)} = \mathrm{LP}(\boldsymbol{Y}^{(t-1)}, \tilde{\boldsymbol{W}}_{\mathbb{s}}, \boldsymbol{Q}_{\mathbb{s}}^{(t-1)}), \tag{4}$$

$$\boldsymbol{Y}^{(t+1)} = \mathrm{LP}(\boldsymbol{Y}^{(t)}, \tilde{\boldsymbol{W}}_{\mathbb{t}}, \boldsymbol{Q}_{\mathbb{s}}^{(t)}), \tag{5}$$

where $\boldsymbol{Y}^{(t)}$ represents the refined label matrix by LP at epoch t. $\tilde{\boldsymbol{W}}_{\mathbb{s}}$ and $\tilde{\boldsymbol{W}}_{\mathbb{t}}$ denote the normalized matrices based on $\mathcal{M}_{\mathbb{s}}$ and $\mathcal{M}_{\mathbb{t}}$, respectively.

Mutual LP shares some similarities to Co-teaching [27], Co-teaching+ [35], and JoCoR [10]. For Co-teaching and Co-teaching+, two networks iteratively teach each other to improve performance. For JoCoR, two networks are collaboratively trained. However, there are some differences between mutual LP and these methods. First, Co-teaching, Co-teaching+, and JoCoR focus on noisy label learning without considering model complexity while mutual LP works on the task of KD with noisy labels. Second, Co-teaching, Co-teaching+, and JoCoR usually jointly optimize the two network parameters at a batch level. On the contrary, our mutual LP improves the performance of the teacher and student models at an epoch level, mitigating the conformation bias to some extent. Third, Co-teaching, Co-teaching+, and JoCoR identify noisy labeled samples via the logit information of the output. In contrast, mutual LP leverages the feature distribution information to identify and correct noisy labeled samples.

It is worth pointing out that both our method and NGC [31] leverage LP to refine noisy labels. However, the differences between our method and NGC are significant. First, we perform mutual LP from the perspective of the teacher and student models, while NGC performs LP by using a single model. Second, NGC treats all the samples equally, and thus it ignores the negative effect of ambiguous samples. In contrast, we introduce the AWE module into LP to mitigate the influence of ambiguous samples during the label refinery, leading to better label accuracy. Third, NGC does not consider the model size, while our method focuses on learning a compact model under the label noise condition.

D. Ambiguity-Aware Weight Estimation (AWE)

During LP in label refinery, it is difficult to assign accurate labels to ambiguous samples (see Fig. 1). Learning from these samples inevitably leads to the overfitting of CNN models, degrading the performance. Hence, we develop an AWE module to mitigate the negative impact of ambiguous samples. Note that the feature representations obtained by $\mathcal{M}_{\mathbb{t}}$ are fixed during the distillation process. To improve the performance of label refinery, we propose to estimate the ambiguity weight for each sample by considering both the feature distribution score and the refined annotations from the perspective of $\mathcal{M}_{\mathbb{s}}$. In detail, we first calculate the prototypes according to feature representations extracted by $\mathcal{M}_{\mathbb{s}}$ as

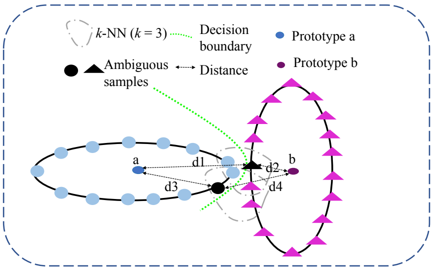


Fig. 3: Illustration of ambiguous samples (the black triangle and the black circle). These ambiguous samples correspond to incorrectly relabeled samples after LP.

$$\boldsymbol{u}_c^{(t)} = \frac{1}{N_c}\sum_{i=1}^{N} \phi_{\mathbb{s}}^{(t)}(\boldsymbol{x}_i)\mathbb{1}\left\{\boldsymbol{y}_i^{(t)}[c]=1\right\}, \tag{6}$$

where $\boldsymbol{u}_c^{(t)} \in \mathbb{R}^m$ is the prototype of class c at epoch t and N_c is the number of samples in class c. $\phi_{\mathbb{s}}^{(t)}(\boldsymbol{x}_i) \in \mathbb{R}^m$ is the feature extracted by $\mathcal{M}_{\mathbb{s}}$ at epoch t.

Then, we calculate the feature distribution score $\boldsymbol{S}_i^{(t)} \in \mathbb{R}^C$ at epoch t for the sample $\boldsymbol{x}_i$ as

$$\boldsymbol{S}_i^{(t)} = [\boldsymbol{u}_1^{(t)\mathrm{T}}\phi_{\mathbb{s}}^{(t)}(\boldsymbol{x}_i), \cdots, \boldsymbol{u}_C^{(t)\mathrm{T}}\phi_{\mathbb{s}}^{(t)}(\boldsymbol{x}_i)]^{\mathrm{T}}. \tag{7}$$

Here, each element of $\boldsymbol{S}_i^{(t)}$ denotes the similarity score between the sample $\boldsymbol{x}_i$ and a prototype at epoch t.

After that, $\boldsymbol{S}_i^{(t)}$ is concatenated with $\boldsymbol{y}_i^{(t)}$ to form the ambiguity feature $\boldsymbol{T}_i^{(t)} \in \mathbb{R}^{2C}$, which reveals the ambiguity in both feature distribution and annotation aspects:

$$\boldsymbol{T}_i^{(t)} = \mathrm{concat}(\boldsymbol{S}_i^{(t)}, \boldsymbol{y}_i^{(t)}), \tag{8}$$

where $\mathrm{concat}(\cdot)$ denotes the concatenation operation.

$\boldsymbol{T}_i^{(t)}$ contains feature distribution information and annotation information, which can be understood in the following two ways (see Fig. 3 for an illustration): (1) For a noisy labeled sample $(\boldsymbol{x}_i, \boldsymbol{y}_i^{(t)})$ with the noisy label $\boldsymbol{y}_i^{(t)}[a] = 1$ while the ground-truth label $\boldsymbol{y}_i^{(t)}[b] = 1$ (see the black triangle in Fig. 3), LP fails to assign the correct label to this sample. However, the feature similarity between $\boldsymbol{x}_i$ and the b-th prototype is high, although the annotation in $\boldsymbol{y}_i^{(t)}$ given by LP indicates that $\boldsymbol{x}_i$ belongs to the a-th class. (2) For a clean but ambiguous sample that is near the decision boundary (see the black circle in Fig. 3), LP assigns an incorrect label to this sample. However, its distances to the two prototypes might be close. Therefore, it is difficult to indicate the label of this sample. Both two cases reveal the ambiguity of these samples. Notice that ambiguous samples and hard training samples with correct labels are intrinsically different since the labels of ambiguous samples are still noisy after LP.

The AWE module takes $\boldsymbol{T}_i^{(t)}$ as the input and outputs a weight $q_i \in (0,1)$ for $\boldsymbol{x}_i$. Specifically, the AWE module consists of two Fully-Connected (FC) layers with a PReLU non-linear function and a sigmoid function (see Fig. 2),

$$q_{\mathfrak{s},i}^{(t)} = \mathrm{Sigmoid}(\boldsymbol{H}_2^{\mathrm{T}}\sigma(\boldsymbol{H}_1^{\mathrm{T}}(\boldsymbol{T}_i^{(t)}))), \tag{9}$$

where $q_{\mathfrak{s},i}^{(t)}$ denotes the weight with respect to $\boldsymbol{x}_i$. Generally, the more ambiguous a sample is, the lower the corresponding weight is. $\boldsymbol{H}_1 \in \mathbb{R}^{2C\times C}$ and $\boldsymbol{H}_2 \in \mathbb{R}^{C\times 1}$ are two FC layers. $\sigma(\cdot)$ and $\mathrm{Sigmoid}(\cdot)$ denote the PReLU non-linear function and the Sigmoid function, respectively. $\boldsymbol{Q}_{\mathfrak{s}}^{(t)} = [\boldsymbol{q}_{\mathfrak{s}}^{(t)}(1), \cdots, \boldsymbol{q}_{\mathfrak{s}}^{(t)}(N)]$ denotes the weight matrix, where $\boldsymbol{q}_{\mathfrak{s}}^{(t)}(i) = q_{\mathfrak{s},i}^{(t)}\boldsymbol{1}_C$ and $\boldsymbol{1}_C \in \mathbb{R}^C$ is an all-ones vector.

E. Overall Training Loss

We use the weighted CE loss as the classification loss:

$$\mathcal{L}_{cls} = -\frac{1}{N}\sum_{i=1}^{N} q_{\mathfrak{s},i}^{(t)}(\boldsymbol{y}_i^{(t)})^{\mathrm{T}}\log(\boldsymbol{p}_{\mathfrak{s}}^{(t)}(\boldsymbol{x}_i)). \tag{10}$$

In this paper, we use Mixup [36] to generate virtual samples by applying convex combinations of pairs of samples and their corresponding labels:

$$\begin{aligned} \boldsymbol{x}_m^v &= \lambda\boldsymbol{x}_i + (1-\lambda)\boldsymbol{x}_j, \\ \boldsymbol{y}_m^{v(t)} &= \lambda\boldsymbol{y}_i^{(t)} + (1-\lambda)\boldsymbol{y}_j^{(t)}, \end{aligned} \tag{11}$$

where λ is sampled from the Beta distribution $Beta(\alpha,\alpha)$. $\boldsymbol{x}_m^v$ and $\boldsymbol{y}_m^{v(t)}$ denote the virtual sample and its corresponding label, respectively.

Based on the virtual samples, the knowledge distillation loss measured by the KL-divergence can be formulated as

$$\mathcal{L}_{kd} = -\tau^2\frac{1}{N}\sum_{m=1}^{N}\tilde{\boldsymbol{p}}_{\mathfrak{t}}^{(t)}(\boldsymbol{x}_m^v;\tau)^{\mathrm{T}}\log(\tilde{\boldsymbol{p}}_{\mathfrak{s}}^{(t)}(\boldsymbol{x}_m^v;\tau)), \tag{12}$$

where $\tilde{\boldsymbol{p}}_{\mathfrak{t}}^{(t)}(\boldsymbol{x}_m^v;\tau) = \mathrm{Softmax}(\boldsymbol{p}_{\mathfrak{t}}^{(t)}(\boldsymbol{x}_m^v);\ \tau)$ and $\tilde{\boldsymbol{p}}_{\mathfrak{s}}^{(t)}(\boldsymbol{x}_m^v;\tau) = \mathrm{Softmax}(\boldsymbol{p}_{\mathfrak{s}}^{(t)}(\boldsymbol{x}_m^v);\ \tau)$. $\mathrm{Softmax}(\cdot)$ is the softmax function. τ is the temperature (we fix it to 4 as in [6]).

Moreover, we calculate the CE loss for virtual samples:

$$\mathcal{L}_{cls\text{-}mix} = -\frac{1}{N}\sum_{m=1}^{N}(\boldsymbol{y}_m^{v(t)})^{\mathrm{T}}\log(\boldsymbol{p}_{\mathfrak{s}}^{(t)}(\boldsymbol{x}_m^v)). \tag{13}$$

To achieve better feature representation capability, we also enforce $\mathcal{M}_{\mathfrak{s}}$ to mimic $\mathcal{M}_{\mathfrak{t}}$ in terms of pairwise similarities. The similarity loss is given as

$$\begin{aligned} \mathcal{L}_{sl} = \frac{2}{N(N-1)}\sum_{i=1}^{N}\sum_{j=1,i\neq j}^{N} &|\mathrm{Sim}(\phi_{\mathfrak{s}}^{(t)}(\boldsymbol{x}_i), \phi_{\mathfrak{s}}^{(t)}(\boldsymbol{x}_j)) - \\ &\mathrm{Sim}(\phi_{\mathfrak{t}}(\boldsymbol{x}_i), \phi_{\mathfrak{t}}(\boldsymbol{x}_j))|^2, \end{aligned} \tag{14}$$

where $\mathrm{Sim}(\mathbf{a},\mathbf{b})$ is the cosine similarity between $\mathbf{a}$ and $\mathbf{b}$.

Therefore, the overall training loss is

$$\mathcal{L}_{all} = \mathcal{L}_{cls} + \lambda_1\mathcal{L}_{kd} + \lambda_2\mathcal{L}_{cls\text{-}mix} + \lambda_3\mathcal{L}_{sl}, \tag{15}$$

where λ_1, λ_2, and λ_3 are the balanced parameters.

The detailed training process of the two-stage label refinery framework is given in Algorithm 1. Our framework is easy-to-implement and effective to generate high-quality labels during the distillation procedure.

Algorithm 1: Two-Stage Label Refinery Framework

```
Input: A pretrained teacher model $\mathcal{M}_{\mathfrak{t}}$; an initial
       student model $\mathcal{M}_{\mathfrak{s}}$; the AWE module $\Psi$; the
       total training epochs $E$; the training epochs for
       the first stage $E_1$.
for $t \leftarrow 1$ to $E$ do
    // Stage 1: LP with small-loss selection
    if $t \leq E_1$ then
        Construct a $k$-NN graph with the features
         extracted by $\mathcal{M}_{\mathfrak{t}}$ via Eq. (1) and get $\tilde{\boldsymbol{W}}_{\mathfrak{t}}$;
        Relabel samples with predictions given by $\mathcal{M}_{\mathfrak{s}}$
         via Eq. (2) and obtain $\boldsymbol{Z}^{(t)}$;
        Calculate the weight matrix $\boldsymbol{Q}_{\mathfrak{s}}^{(t-1)}$ obtained
         by $\Psi$ via Eq. (9);
        Get the refined label matrix $\boldsymbol{Y}^{(t)}$ via Eq. (3);
    end
    // Stage 2: Mutual LP
    else if $(t - E_1)\%2! = 0$ then
        Construct a $k$-NN graph with the features
         extracted by $\mathcal{M}_{\mathfrak{s}}$ via Eq. (1) and get $\tilde{\boldsymbol{W}}_{\mathfrak{s}}$;
        Calculate the weight matrix $\boldsymbol{Q}_{\mathfrak{s}}^{(t-1)}$ obtained
         by $\Psi$ via Eq. (9);
        Get the refined label matrix $\boldsymbol{Y}^{(t)}$ via Eq. (4);
    end
    else
        Construct a $k$-NN graph with the features
         extracted by $\mathcal{M}_{\mathfrak{t}}$ via Eq. (1) and get $\tilde{\boldsymbol{W}}_{\mathfrak{t}}$;
        Calculate the weight matrix $\boldsymbol{Q}_{\mathfrak{s}}^{(t-1)}$ obtained
         by $\Psi$ via Eq. (9);
        Get the refined label matrix $\boldsymbol{Y}^{(t)}$ via Eq. (5);
    end
    Calculate the overall training loss $\mathcal{L}_{all}$ with $\boldsymbol{Y}^{(t)}$
     via Eq. (15);
    Update $\mathcal{M}_{\mathfrak{s}}$ and $\Psi$;
end
Output: A well-trained student model $\mathcal{M}_{\mathfrak{s}}$.
```

IV. Experiments

In this section, we evaluate our AML-KD method on synthetic and real-world noisy datasets. First, we introduce datasets and implementation details in Section IV-A. Then, we give ablation studies in Section IV-B. Finally, we compare our proposed method with several state-of-the-art methods in Section IV-C.

A. Datasets and Implementation Details

1) Datasets: In this paper, we adopt CIFAR-100 [37], Animal-10N [7], Clothing1M [38], and WebVision [39] for performance evaluation in our experiments.

CIFAR-100 contains 100 object categories with 60,000 images (50,000 images for training and 10,000 images for testing). Following previous works [10], [27], we corrupt the training set of CIFAR-100 with two types of label noise: symmetric noise and asymmetric noise. Animal-10N is a real-world noisy dataset containing ten confusing animals with 55,000 images (50,000 images for training and 5,000 images

TABLE I: Test accuracy (%) obtained by different variants of our method for three teacher-student pairs on CIFAR-100.

S1	S2	AWE	WRN_40_2 WRN_16_2	WRN_40_2 WRN_40_1	ResNet56 ResNet20
×	×	×	65.21	64.37	63.01
✓	×	×	69.84	68.36	64.08
×	✓	×	69.22	67.88	63.71
×	×	✓	66.09	64.38	63.22
✓	×	✓	70.01	68.54	64.98
×	✓	✓	68.78	68.09	64.27
✓	✓	×	70.08	68.88	65.01
✓	✓	✓	**70.54**	**69.48**	**65.33**

for testing). Clothing1M and WebVision are two large-scale real-world noisy datasets. Clothing1M contains about one million images that are collected from online shopping websites. WebVision includes around 2.4 million training images obtained from the web using 1,000 concepts in ImageNet ILSVRC12. Following [9], we adopt the first 50 classes of the Google image subset for training and testing in WebVision (termed WebVision (mini)).

2) Implementation Details: For all the experiments, the SGD optimizer is used, where the weight decay and momentum are set to 0.0001 and 0.90, respectively. For KD methods (including KD [17], KD (logit only), RKD [40], SSKD [6], CRD [21], ICKD-C [14], TaT [15], and AML-KD), we use a representative LNL method (i.e., DivideMix [9]) to train the teacher model unless explicitly mentioned otherwise. We train the student model with the original annotations in the first 10 epochs for warm-up before the two-stage label refinery. The teacher and student models are trained with the same noisy training set.

For CIFAR-100 and Animal-10N, the batch size is set to 64. We run 240 epochs in total. The initial learning rate is set to 0.05 and is decayed by a factor of 10 at 150, 180, and 210 epochs. Moreover, we perform LP with small-loss selection for 140 epochs and mutual LP for 90 epochs. For Clothing1M, the batch size is set to 32. We run 100 epochs in total, and the initial learning rate is set to 0.05 and is decayed by a factor of 10 at 60 and 80 epochs. We perform LP with small-loss selection at the first 60 epochs and mutual LP at the rest epochs. For WebVision, we set the batch size to 32 and totally run 80 epochs. The initial learning rate is set to 0.02 and decayed to 0.002 after 60 epochs. We perform LP with small-loss selection at the first 60 epochs and mutual LP in the rest epochs. In all our experiments, we empirically set $k = 30$ in Eq. (1) and $\beta = 0.80$ in Eq. (3). Besides, we set $\lambda_1 = \lambda_2 = 1.00$ and $\lambda_3 = 0.50$ in Eq. (15). Following Co-teaching [27], we assume that the noise rate r is given before model training. For ANIMAL-10N, Clothing1M, and WebVision, we set the noise rate to 8.0%, 38.5%, and 20.0%, respectively, as suggested by [41].

For CIFAR-100, we evaluate two types of label noise (i.e., symmetric noise and asymmetric noise) with two noise rates (i.e., 20% and 50%). Moreover, we evaluate the setting that the whole training dataset is divided into two subsets (a clean subset and a noisy subset) with the same numbers of samples, where the teacher model is pretrained by the clean subset while the student model is trained by the noisy subset. We denote this setting as C-N.

For Sym-20% (C-N) and Sym-50% (C-N), the teacher models used in the KD methods (including KD, SSKD, RKD, CRD, and our AML-KD) are trained by using the standard CE loss. For other settings, the teacher models are trained by DivideMix. The Student (CE) method refers to the method that the student model is trained by the standard CE loss. For the LNL methods (including Co-teaching and DivideMix), only the student models are used for training.

B. Ablation Studies

In this subsection, three representative teacher-student pairs (including WRN_40_2-WRN_16_2, WRN_40_2-WRN_40_1, and ResNet56-ResNet20) [42], [43] are selected to investigate the generalization ability of the method.

1) Influence of Key Components in AML-KD: To validate the superiority of key components in our method, we conduct ablation studies under the Sym-50% (C-N) label noise on CIFAR-100. In our AML-KD, a two-stage label refinery framework (consisting of the LP with small-loss selection stage and the mutual LP stage) and an AWE module are developed to address noisy labels. We abbreviate the two stages as S1 and S2, respectively. Experimental results obtained by different variants of our method are shown in Table I. Note that when AWE is not used, the weight matrix is set to an all-ones matrix. For AML-KD only with AWE, the weight matrix is used to compute the weighted CE loss defined in Eq. (10).

Among all the variants, AML-KD achieves the worst performance in three teacher-student pairs when S1, S2, and AWE are not used (i.e., AML-KD is trained by the overall loss defined in Eq. (15) and the original annotations). AML-KD with only AWE gives the second-worst classification accuracy. This is mainly due to the negative influence of noisy labels, leading to overfitting. AML-KD with S1 performs better than that with S2. Note that the initial feature learning capability of the student model is poor when S1 is not used. Thus, the student model cannot effectively capture the neighboring relationships between samples, thereby limiting the label refinery accuracy in S2. By incorporating AWE into S1 or S2, the performance is improved. When S1, S2, and AWE are all used, AML-KD gives the best performance in all three teacher-student pairs. The above results show the necessity of each component in AML-KD.

2) Influence of the Balanced Parameters λ_1, λ_2, and λ_3: We evaluate the performance of AML-KD with the different values of λ_1, λ_2, and λ_3 in Eq. (15) under the Sym-50% (C-N) label noise on CIFAR-100. The results are given in Tables II(a)-II(c).

Specifically, we first fix $\lambda_2 = 1.00$ and $\lambda_3 = 0.50$, and set the value of λ_1 from 0.00 to 2.00. The results are listed in Table II(a). From Table II(a), we can observe that our method achieves the best performance when the value of λ_1 is set to 1.00. When λ_1 is set to 0.00, which indicates that AML-KD is trained without the knowledge distillation loss, leading to the worst performance. Next, we fix $\lambda_1 = 1.00$ and $\lambda_3 = 0.50$ to investigate the influence of λ_2 (see Table II(b)). Obviously,

TABLE II: Test accuracy (%) for the different values of λ_1, λ_2, λ_3, k, and β for the three teacher-student pairs on CIFAR-100. (a)-(c) Influence of λ_1, λ_2, and λ_3. (d)-(f) Influence of k. (g)-(i) Influence of β.

(a) Influence of λ_1.

λ_1	p1	p2	p3
0.00	65.54	63.81	61.08
0.50	67.29	68.41	65.13
1.00	**70.54**	**69.48**	**65.33**
1.50	69.86	69.22	64.88
2.00	69.54	69.10	64.86

(b) Influence of λ_2.

λ_2	p1	p2	p3
0.00	69.55	68.60	64.67
0.50	70.10	69.37	65.06
1.00	**70.54**	**69.48**	**65.33**
1.50	70.12	69.41	65.21
2.00	69.82	69.31	65.00

(c) Influence of λ_3.

λ_3	p1	p2	p3
0.00	70.00	69.32	64.77
0.50	**70.54**	**69.48**	**65.33**
1.00	69.44	69.31	64.72
1.50	69.98	69.06	65.01
2.00	69.69	69.19	65.20

(d) Results of p1.

k	Sym-20%	Sym-50%
20	71.17	69.98
30	**71.71**	**70.54**
50	71.50	69.61
100	69.55	69.44

(e) Results of p2.

k	Sym-20%	Sym-50%
20	69.88	68.97
30	70.02	**69.48**
50	**70.50**	68.93
100	69.55	68.81

(f) Results of p3.

k	Sym-20%	Sym-50%
20	66.12	64.91
30	**66.49**	**65.33**
50	65.50	64.99
100	65.18	64.01

(g) Results of p1.

β	Sym-20%	Sym-50%
0.70	71.16	70.08
0.80	**71.71**	**70.54**
0.90	71.46	70.28
0.99	**71.71**	70.11

(h) Results of p2.

β	Sym-20%	Sym-50%
0.70	69.77	69.18
0.80	70.02	**69.48**
0.90	**70.22**	69.31
0.99	69.91	69.09

(i) Results of p3.

β	Sym-20%	Sym-50%
0.70	66.23	65.12
0.80	**66.49**	**65.33**
0.90	66.41	65.11
0.99	66.29	64.90

"p1", "p2", "p3" denote WRN_40_2-WRN_16_2, WRN_40_2-WRN_40_1, and ResNet56-ResNet20, respectively.

our method obtains the top accuracy when $\lambda_2 = 1.00$. Finally, we fix $\lambda_1 = 1.00$ and $\lambda_2 = 1.00$ to demonstrate the influence of λ_3. As shown in Table II(c), AML-KD achieves the best performance in all three teacher-student pairs when $\lambda_3 = 0.50$. In all our experiments, we fix the values of λ_1, λ_2, and λ_3 to 1.00, 1.00, and 0.50, respectively.

3) Influence of the Number of Neighbors k and the Regularized Parameter β: To investigate the influence of k and β, we conduct ablation studies under the Sym-20% (C-N) and Sym-50% (C-N) label noise on CIFAR-100. The results are given in Tables II(d)-II(f) and Tables II(g)-II(i).

We first fix $\beta = 0.80$ and set the value of k from 20 to 100. As shown in Tables II(d)-II(f), for most cases, AML-KD achieves the best performance when $k = 30$ and obtains the worst performance when $k = 100$ under the Sym-20% (C-N) and Sym-50% (C-N) label noise. The performance differences are not significant when the value of k ranges in [20, 50]. Then, we fix $k = 30$ and set the value of β from 0.70 to 0.99 to evaluate the influence of β. From Tables II(g)-II(i), we can observe that AML-KD achieves the best performance when $\beta = 0.80$ in most cases. In all our experiments, we fix the values of k and β to 30 and 0.80, respectively.

4) Influence of the Estimated Noise Rate r: In this paper, we assume that the noise rate r is given for each dataset. However, the noise rate is often unknown in many real-world applications. Therefore, we verify the performance of our proposed AML-KD without giving an accurate noise rate. Specifically, following the strategy used in [9], we adopt Gaussian Mixture Models (GMMs) to the classification losses of training samples for selecting clean samples. We set the posterior probability of clean samples to 0.50 in our experiments. We perform experiments under the Sym-20%, Sym-50%, Sym-20% (C-N), and Sym-50% (C-N) label noise on CIFAR-100. The results are given in Fig. 4.

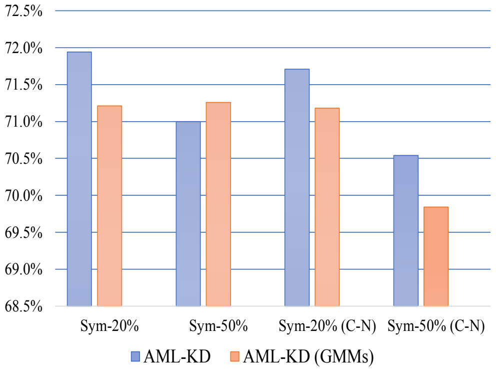


Fig. 4: Test accuracy under the Sym-20%, Sym-50%, Sym-20% (C-N) and Sym-50% (C-N) label noise on CIFAR-100. WRN_40_2-WRN_16_2 is used as the teacher-student pair.

As shown in Fig. 4, AML-KD with the estimated noise rate obtains only slight performance degeneration compared with AML-KD with the accurate noise rate. This demonstrates that AML-KD works well when r is estimated with GMM.

5) Effectiveness of the Two-Stage Label Refinery Framework (TSLRF): Our two-stage label refinery framework can be easily applied to other KD-based methods when the datasets involve class-dependent label noise. Thus, we explore the effectiveness of our TSLRF on several KD-based methods (including KD, SSKD, and RKD) under the Sym-50% (C-N) label noise on CIFAR-100. Specifically, we replace the training loss $\mathcal{L}_{all}$ defined in Eq. (15) with the corresponding losses used in KD, SSKD, and RKD. In this way, TSLRF can be used to gradually refine noisy labels. The comparison

TABLE III: Accuracy (%) obtained by different methods for the four teacher-student pairs with different types of label noise on CIFAR-100. All the competing methods are re-implemented by using author-provided codes.

Teacher Student	Methods			Noise Type		
		Sym-20%	Sym-50%	Asym-50%	Sym-20% (C-N)	Sym-50% (C-N)
WRN_40_2 WRN_16_2	Teacher	73.99	71.29	71.07	69.99	69.99
	Student (CE)	55.71	37.97	40.18	54.70	33.38
	KD [17]	68.81	67.82	68.28	65.85	63.47
	KD (logit only)	64.31	63.39	63.19	62.87	62.87
	RKD [40]	70.43	66.71	62.15	66.13	61.10
	CRD [21]	70.80	68.46	69.31	69.12	67.88
	SSKD [6]	69.01	67.56	68.06	68.78	68.97
	ICKD-C [14]	71.63	70.29	69.98	71.68	69.84
	TaT [15]	71.69	70.18	69.65	71.27	69.63
	UNICON [11]	70.26	67.58	67.01	69.00	67.37
	DivideMix [9]	69.99	66.86	65.55	68.74	65.32
	Co-teaching [27]	56.18	46.37	42.87	54.13	47.76
	SELC [44]	69.01	66.48	66.03	66.74	65.99
	AML-KD	**71.94**	**71.70**	**70.28**	**71.71**	**70.54**
WRN_40_2 WRN_40_1	Teacher	73.99	71.29	71.07	69.99	69.99
	Student (CE)	53.89	36.43	37.64	53.01	34.90
	KD [17]	70.20	67.84	67.86	67.73	60.72
	KD (logit only)	67.71	66.03	65.29	64.35	64.35
	RKD [40]	70.01	66.04	62.51	66.31	61.18
	CRD [21]	69.38	68.10	67.90	68.34	66.68
	SSKD [6]	67.46	67.07	67.77	67.51	66.88
	ICKD-C [14]	71.01	70.03	69.58	69.99	69.00
	TaT [15]	70.59	69.46	69.46	69.88	68.93
	UNICON [11]	69.46	69.01	68.64	68.33	67.54
	DivideMix [9]	69.01	66.71	65.33	67.91	66.04
	Co-teaching [27]	58.97	49.91	52.68	56.54	47.11
	SELC [44]	68.65	64.26	64.11	65.21	64.18
	AML-KD	**71.49**	**71.05**	**70.00**	**70.02**	**69.48**
ResNet56 ResNet20	Teacher	69.71	66.09	65.65	66.89	66.89
	Student (CE)	53.31	34.76	35.88	53.06	35.70
	KD [17]	67.70	65.67	66.57	64.63	62.53
	KD (logit only)	63.28	62.21	62.37	61.11	61.11
	RKD [40]	67.02	65.41	65.89	64.64	64.56
	CRD [21]	67.50	66.41	65.64	65.55	64.50
	SSKD [6]	64.16	63.87	63.52	64.99	63.42
	ICKD-C [14]	67.82	66.16	66.09	66.41	65.03
	TaT [15]	67.28	67.04	66.72	66.38	64.43
	UNICON [11]	66.28	65.03	64.87	66.26	64.53
	DivideMix [9]	63.99	61.75	61.64	63.38	61.07
	Co-teaching [27]	57.96	48.46	49.71	57.23	48.19
	SELC [44]	67.77	65.53	65.09	65.28	64.21
	AML-KD	**67.91**	**67.40**	**66.82**	**66.49**	**65.33**
WRN_40_2 ShuffleNetV1	Teacher	73.99	71.29	71.07	69.99	69.99
	Student (CE)	55.81	31.99	37.87	55.29	38.86
	KD [17]	68.78	66.67	66.05	65.73	65.27
	KD (logit only)	63.29	62.18	62.51	62.11	62.11
	RKD [40]	67.99	66.13	65.79	66.78	65.47
	CRD [21]	69.80	68.01	67.27	69.18	67.29
	SSKD [6]	68.25	67.00	66.04	66.52	64.39
	ICKD-C [14]	70.28	69.18	69.01	70.37	68.99
	TaT [15]	69.98	68.57	68.03	69.82	69.08
	UNICON [11]	69.18	66.79	66.31	67.99	67.38
	DivideMix [9]	69.48	66.79	66.41	67.52	66.01
	Co-teaching [27]	59.88	55.26	54.21	58.39	53.46
	SELC [44]	67.98	65.44	65.10	65.38	64.19
	AML-KD	**70.91**	**70.45**	**70.26**	**71.19**	**70.66**

results are given in Fig. 5.

From Fig. 5, we see that when applying our TSLRF to these KD-based methods, the performance can be effectively boosted. This shows that our label refinery framework can be successfully combined with the existing KD-based methods to improve their performance, even when the dataset contains massive noisy labeled samples.

In the first stage of TSLRF, we directly relabel the noisy labeled samples according to the prediction of the student model (see Eq. (2)). Note that many existing label-noise learning methods (such as [29]) rely on a manually-selected threshold for label correction. Hence, we also compare the threshold-based label correction (the values of the threshold (t) are set to 0.2 and 0.5, respectively) and our adopted label correction mechanism. The results are given in Fig. 6.

Experimental results show that our method with threshold-

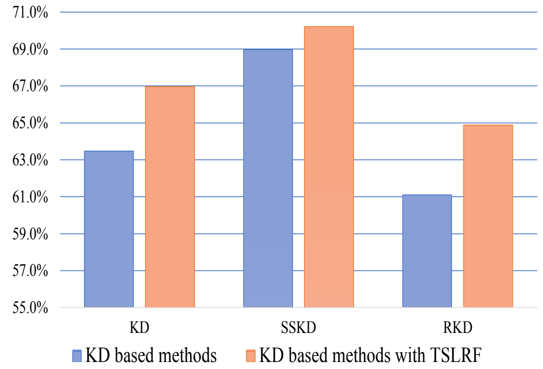


Fig. 5: Test accuracy under the Sym-50% (C-N) label noise on CIFAR-100. WRN_40_2-WRN_16_2 is used as the teacher-student pair.

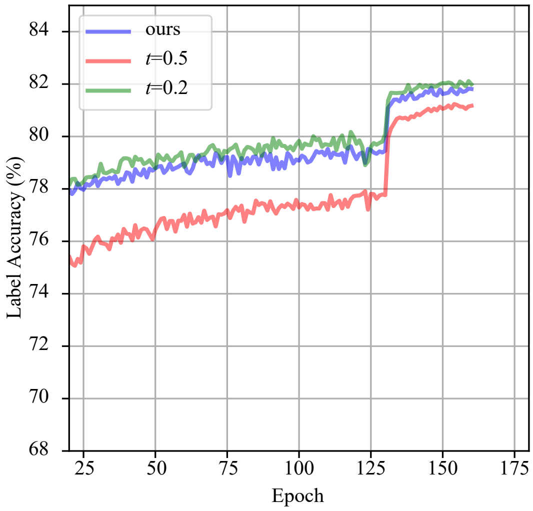


Fig. 6: Label accuracy obtained by the threshold-based label correction and our adopted mechanism under Sym-50% (C-N) label noise on CIFAR-100.

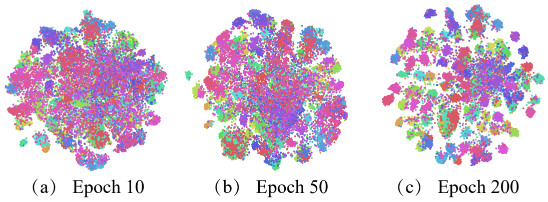
based label correction achieves similar performance in terms of label accuracy in comparison with our method with the adopted mechanism. However, threshold-based label correction requires manually choosing a threshold for each dataset, hindering its practical applications. On the contrary, our adopted label correction mechanism performs label correction without using the threshold. Moreover, the label correction in Eq. (2) is only used in the first stage. Even if some samples are not correctly relabeled in this stage, the labels of these samples can still be corrected in the second stage.

6) Influence of Different Teacher Models: We show the influence of different teacher models on the final performance. The results are shown in Table IV.

Experimental results show that AML-KD using UNICON as the teacher model can achieve better performance than that using DivideMix (the default teacher model in this paper) as the teacher model under the Sym-50% and Asym-50% label noise. This clearly validates that the performance of AML-KD can be further improved when a superior teacher model is used.

TABLE IV: Comparison of AML-KD with different teacher models trained via different label noise learning methods on CIFAR-100. ResNet56-ResNet20 is selected as the teacher-student pair. Teacher (DivideMix) and Teacher (UNICON) represent the teacher models trained with DivideMix and UNICON, respectively. AML-KD (DivideMix) and AML-KD (UNICON) denote the AML-KD methods guided by Teacher (DivideMix) and Teacher (UNICON), respectively.

Methods	Sym-50%	Asym-50%
Teacher (DivideMix)	66.09	65.65
Teacher (UNICON)	66.78	66.71
AML-KD (DivideMix)	67.40	66.82
AML-KD (UNICON)	**67.68**	**67.01**



(a) Epoch 10 (b) Epoch 50 (c) Epoch 200

Fig. 7: t-SNE visualization of low-dimensional embeddings (color represents the ground-truth label) at (a) epoch 10, (b) epoch 50, and (c) epoch 200 under the Sym-50% label noise on CIFAR-100.

C. Comparison with State-of-the-Art Methods

1) Results on CIFAR-100: We compare AML-KD with representative LNL methods (including Co-teaching [27], DivideMix [9], SELC [44], and UNICON [11]) and KD methods (including KD [17], KD (logit only), SSKD [6], RKD [40], CRD [21], ICKD-C [14], and TaT [15]) in Table III. For a fair comparison, we report the results of all the competing methods without model ensemble.

From Table III, our proposed AML-KD consistently outperforms all the competing methods, including state-of-the-art KD and LNL methods. The Student (CE) method obtains the worst performance when the noise rate is high (i.e., 50%). This is because of the memorization of noisy labeled samples. Note that CRD achieves much worse performance than our method under Sym-50% (C-N) since it does not consider the influence of noisy labeled samples. Besides, ICKD-C shows better performance than KD, since ICKD-C uses inter-channel correlations as the knowledge to guide the training of the student model. TaT outperforms SSKD by a large margin. This is mainly due to that TaT uses a one-to-all spatial matching knowledge distillation strategy, which can learn fine-grained knowledge hidden in the teacher model. In general, the above results validate the superiority of our method. This can be ascribed to the effectiveness of the proposed two-stage label refinery framework and the AWE module.

TABLE V: Comparison of training time (hours) under the Sym-50% label noise on CIFAR-100 (a single NIVDIA RTX 2080Ti GPU is used). For DivideMix and Co-teaching, WRN_16_2 is used as the backbone. For AML-KD and SSKD, WRN_40_2-WRN_16_2 is used as the teacher-student pair.

Co-teaching	DivideMix	SSKD	AML-KD
5.1	6.9	6.4	6.0

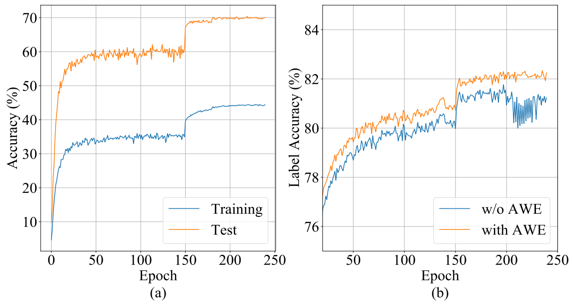


Fig. 8: (a) Training and test accuracy under the Sym-50% (C-N) label noise on CIFAR-100. (b) Label accuracy under the Sym-50% (C-N) label noise on CIFAR-100. WRN_40_2-WRN_16_2 is used as the teacher-student pair.

Moreover, we adopt t-SNE [48] to visualize low-dimensional embeddings given by the student model for all training samples of CIFAR-100, as shown in Fig. 7. As training proceeds, our method learns the features with enhanced intra-class compactness and inter-class separation. Besides, we also compare the training time obtained by several representative methods under the Sym-50% (C-N) label noise on CIFAR-100 in Table V. Our method outperforms the competing methods with fewer or comparable training time. This demonstrates the training efficiency of our AML-KD.

Finally, we visualize the training and test accuracy vs. training epochs in Fig. 8(a). The training accuracy of our method is lower than 50%. This reveals that AML-KD is able to reduce overfitting (since the noise rate is 50%). In addition, we also visualize the label accuracy vs. training epochs in Fig. 8(b). The noisy labels of most samples are gradually refined after two stages. Compared with AML-KD without AWE, AML-KD achieves consistently higher label accuracy, showing the importance of AWE.

2) Results on Animal-10N: We use WRN_40_2-WRN_16_2 as the teacher-student pair on Animal-10N. The results are listed in Table VI. AML-KD gives the top accuracy (82.84%) while Co-learning obtains slightly worse accuracy (82.18%) at the last epoch. But Co-learning requires much more network parameters (∼30 times larger) than AML-KD. SSKD outperforms KD by using self-supervised tasks. In addition, the GFLOPs obtained by AML-KD are comparable to most of the competing methods. These results show the superiority of AML-KD in terms of accuracy and model size.

TABLE VI: Classification accuracy (%), the number of parameters (Params), and the giga-floating point operations (GFLOPs) on the Animal-10N dataset.

Methods	Best	Last	Backbone	Params (M)	GFLOPs
Teacher	79.90	78.61	WRN_40_2	≈ 2.2	≈ 1.3
Noisy Teacher	78.61	76.18	WRN_40_2	≈ 2.2	≈ 1.3
Student (CE)	78.28	77.36	WRN_16_2	≈ 0.7	≈ 0.4
KD [17]	80.70	78.91	WRN_16_2	≈ 0.7	≈ 0.4
SSKD [6]	81.91	79.12	WRN_16_2	≈ 0.7	≈ 0.4
CRD [21]	82.17	79.29	WRN_16_2	≈ 0.7	≈ 0.4
Co-learning [45]	82.95	82.18	ResNet34	≈ 21.3	≈ 4.7
Co-teaching [27]	81.34	80.56	WRN_16_2	≈ 0.7	≈ 0.4
TS^3-Net [29]	-	81.36	VGG-19	≈ 18.0	≈ 0.2
AML-KD (NT)	82.04	81.87	WRN_16_2	≈ 0.7	≈ 0.4
AML-KD	**82.98**	**82.84**	WRN_16_2	≈ 0.7	≈ 0.4

TABLE VII: Top-1 test accuracy (%), the number of parameters (Params), and the giga-floating point operations (GFLOPs) on the Clothing1M dataset.

Method	Accuracy	Backbone	Params (M)	GFLOPs
Teacher	73.79	ResNet50	≈ 23.5	≈ 4.1
Noisy Teacher	73.79	ResNet50	≈ 23.5	≈ 4.1
Student (CE)	65.60	ResNet18	≈ 11.7	≈ 1.8
KD [17]	71.98	ResNet18	≈ 11.7	≈ 1.8
SSKD [6]	71.28	ResNet18	≈ 11.7	≈ 1.8
DivideMix [9]	72.12	ResNet18	≈ 11.7	≈ 1.8
DAT [46]	73.00	ResNet50	≈ 23.5	≈ 4.1
Co-teaching [27]	69.79	ResNet18	≈ 11.7	≈ 1.8
FINE [28]	72.91	ResNet50	≈ 23.5	≈ 4.1
TS^3-Net [29]	72.09	ResNet18	≈ 2.5	≈ 0.4
BARE [47]	72.28	ResNet50	≈ 23.5	≈ 4.1
SELC [44]	74.01	ResNet50	≈ 23.5	≈ 4.1
SRCC [25]	73.99	ResNet50	≈ 23.5	≈ 4.1
UNICON [11]	73.16	ResNet18	≈ 11.7	≈ 1.8
AML-KD (NT)	73.04	ResNet18	≈ 11.7	≈ 1.8
AML-KD	**74.12**	ResNet18	≈ 11.7	≈ 1.8

In some scenarios, the teacher model may be difficult to be trained with dedicated LNL methods. To evaluate the effectiveness of AML-KD in such a case, we conduct additional experiments, where the teacher model (we call it the noisy teacher) is trained by only using the standard CE loss. The results are given in Table VI, where our AML-KD method that is trained based on the noisy teacher is denoted as AML-KD (NT).

From Table VI, we can easily observe that our AML-KD (NT) outperforms its corresponding noisy teacher model. This can be ascribed to the effectiveness of our proposed two-stage label refinery framework, which refines massive noisy labels during the training process. Therefore, our AML-KD is still effective even when the noisy teacher model is used on the real-world dataset.

3) Results on Clothing1M: We adopt ResNet50-ResNet18 as the teacher-student pair on Clothing1M. The comparison results are given in Table VII. AML-KD obtains much better performance than the Student (CE) method with a large margin (8.38% improvements). In particular, our AML-KD achieves slightly better performance than the teacher model. This can be ascribed to the involvement of the mutual LP stage, which gradually improves the label accuracy supervised by the two models. These results show the superiority of AML-KD in the real-world noisy dataset.

We also perform experiments when the noisy teacher is used

TABLE VIII: The top-1, top-5 accuracy (%), the number of parameters (Params), and the giga-floating point operations (GFLOPs) obtained by different methods on WebVision (mini). The accuracy is reported on both the WebVision validation set and the ImageNet ILSVRC12 validation set.

Test dataset	WebVision		ILSVRC12		-	-	-
Accuracy (%)	top-1	top-5	top-1	top-5	Backbone	Params (M)	GFLOPs
Teacher	75.31	90.65	73.14	89.64	Inception-ResNet-v2	≈ 54.4	≈ 13.2
Noisy Teacher	69.36	88.20	67.98	87.03	Inception-ResNet-v2	≈ 54.4	≈ 13.2
Student (CE)	63.25	82.17	61.36	80.41	ResNet18	≈ 11.7	≈ 1.8
KD [17]	70.21	88.37	68.29	87.32	ResNet18	≈ 11.7	≈ 1.8
F-correction [49]	61.12	82.68	57.36	82.36	Inception-ResNet-v2	≈ 54.4	≈ 13.2
Decoupling [50]	62.54	84.74	58.26	82.26	Inception-ResNet-v2	≈ 54.4	≈ 13.2
D2L [51]	62.68	84.00	57.80	81.36	Inception-ResNet-v2	≈ 54.4	≈ 13.2
MentorNet [52]	63.00	81.40	57.80	79.92	Inception-ResNet-v2	≈ 54.4	≈ 13.2
Co-teaching [27]	63.58	85.20	61.48	84.70	Inception-ResNet-v2	≈ 54.4	≈ 13.2
Iterative-CV [53]	65.24	85.34	61.60	84.98	Inception-ResNet-v2	≈ 54.4	≈ 13.2
TS^3-Net [29]	73.48	90.92	70.27	90.46	Inception-ResNet-v2	≈ 11.8	≈ 2.6
DivideMix [9]	73.58	90.27	71.64	90.09	ResNet18	≈ 11.7	≈ 1.8
UNICON [11]	73.88	**91.28**	71.54	90.66	ResNet18	≈ 11.7	≈ 1.8
AML-KD (NT)	73.28	90.32	71.43	90.28	ResNet18	≈ 11.7	≈ 1.8
AML-KD	**74.11**	90.98	**72.09**	**90.78**	ResNet18	≈ 11.7	≈ 1.8

for training. The results are given in Table VII. We can see that, compared with the results obtained by AML-KD (the LNL method (DivideMix) is used to train the teacher model), the performance of AML-KD (NT) does not greatly drop. Moreover, AML-KD (NT) outperforms most of the competing methods. This shows the robustness of AML-KD when the noisy teacher model is adopted as the teacher.

4) Results on WebVision: We show the evaluation results on WebVision in Table VIII. Following the same settings as [9], we use the Inception-ResNet-v2 (which is trained by DivideMix) as the teacher model, and select ResNet18 as the student model. Among all the competing methods, our AML-KD achieves the best performance in terms of top-1 and top-5 accuracy on the ILSVRC12 test dataset with only 1.8 GFLOPs. For the WebVision test dataset, UNICON achieves the best top-5 accuracy (91.28%) while our method obtains comparable accuracy (90.98%).

We also evaluate AML-KD (NT) on WebVision. As shown in Table VIII, AML-KD (NT) significantly outperforms the noisy teacher and achieves comparable performance with AML-KD. This further demonstrates the scalability of our method on the large-scale real-world noisy dataset.

V. Conclusion and Future Work

In this paper, we develop a novel AML-KD method, which is based on a proposed two-stage label refinery framework, to effectively obtain a compact and high-accuracy student model with label noise. To alleviate the overfitting of ambiguous samples during the label refinery, an AWE module is introduced to assign low weights to these samples by exploiting both the feature distribution information and the annotation information. Extensive experiments on synthetic and real-world noisy datasets demonstrate that AML-KD consistently outperforms several state-of-the-art KD methods and LNL methods in dealing with different types of label noise.

In this paper, we study KD with noisy labels, where the datasets involve class-dependent label noise. However, in some real-world applications, out-of-distribution label noise may also exist. Therefore, our method is not applicable in such a case. To perform KD under out-of-distribution label noise, we can exploit statistical information from the teacher model and the student model to identify out-of-distribution noisy labeled samples in future work. Moreover, we believe that our method can be extended to other supervised learning tasks. For example, AML-KD can be used to obtain a lightweight text emotion classifier for emotion analysis when the training set contains noisy tags.

References

[1] Y. Cheng, D. Wang, P. Zhou, and T. Zhang, "A survey of model compression and acceleration for deep neural networks," 2017, *arXiv:1710.09282*.

[2] Y.-J. Zheng, S.-B. Chen, C. H. Ding, and B. Luo, "Model compression based on differentiable network channel pruning," *IEEE Trans. Neural Netw. Learn. Syst.*, pp. 1–10, 2022.

[3] S. M. Shah and V. K. Lau, "Model compression for communication efficient federated learning," *IEEE Trans. Neural Netw. Learn. Syst.*, pp. 1–15, 2021.

[4] A. Romero, N. Ballas, S. E. Kahou, A. Chassang, C. Gatta, and Y. Bengio, "FitNets: Hints for thin deep nets," 2014, *arXiv:1412.6550*.

[5] Y. Liu, J. Cao, B. Li, C. Yuan, W. Hu, Y. Li, and Y. Duan, "Knowledge distillation via instance relationship graph," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2019, pp. 7096–7104.

[6] G. Xu, Z. Liu, X. Li, and C. C. Loy, "Knowledge distillation meets self-supervision," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2020, pp. 588–604.

[7] H. Song, M. Kim, and J.-G. Lee, "SELFIE: Refurbishing unclean samples for robust deep learning," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2019, pp. 5907–5915.

[8] 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 Int. Conf. Comput. Vis. (ICCV)*, 2019, pp. 322–330.

[9] J. Li, R. Socher, and S. C. Hoi, "DivideMix: Learning with noisy labels as semi-supervised learning," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2019, pp. 1–14.

[10] 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)*, 2020, pp. 13 726–13 735.

[11] N. Karim, M. N. Rizve, N. Rahnavard, A. Mian, and M. Shah, "UNICON: Combating label noise through uniform selection and contrastive learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2022, pp. 9676–9686.

[12] Y. Li, J. Yang, Y. Song, L. Cao, J. Luo, and L.-J. Li, "Learning from noisy labels with distillation," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, 2017, pp. 1910–1918.

[13] Y. Chen, S. X. Hu, X. Shen, C. Ai, and J. A. Suykens "Compressing features for learning with noisy labels," *IEEE Trans. Neural Netw. Learn. Syst.*, pp. 1–15, 2021.

[14] L. Liu, Q. Huang, S. Lin, H. Xie, B. Wang, X. Chang, and X. Liang, "Exploring inter-channel correlation for diversity-preserved knowledge distillation," in *Proc. IEEE Int.Conf. Comput. Vis. (ICCV)*, 2021, pp. 8251–8260.

[15] S. Lin, H. Xie, B. Wang, K. Yu, X. Chang, X. Liang, and G. Wang, "Knowledge distillation via the target-aware transformer," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2022, pp.10 905–10 914.

[16] C. Li, J. Peng. Rizve, L. Yuan, A. Mian, G. Wang, X. Liang, L. Lin, and X. Chang, "Block-wisely supervised neural architecture search with knowledge distillation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2020, pp. 1989–1998.

[17] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," 2015,*arXiv:1503.02531*.

[18] S. Sun, Y. Cheng, Z. Gan, and J. Liu, "Patient knowledge distillation for bert model compression," in *Proc. Conf. Empir. Methods Nat. Lang. Process. Int. Jt. Conf. Nat. Lang. Process. (EMNLP-IJCNLP)*, 2019, pp. 4323–4332.

[19] S. Zagoruyko and N. Komodakis, "Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer," 2016, *arXiv:1612.03928*.

[20] F. Tung and G. Mori, "Similarity-preserving knowledge distillation," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, 2019, pp. 1365–1374.

[21] Y. Tian, D. Krishnan, and P. Isola, "Contrastive representation distillation," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2019, pp. 1–19.

[22] K. Xu, L. Rui, Y. Li, and L. Gu, "Feature normalized knowledge distillation for image classification," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2020, pp. 664–680.

[23] S. Guo, W. Huang, H. Zhang, C. Zhuang, D. Dong, M. R. Scott, and D. Huang, "CurriculumNet: Weakly supervised learning from large-scale web images," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 135–150.

[24] H. Harutyunyan, K. Reing, G. Ver Steeg, and A. Galstyan, "Improving generalization by controlling label-noise information in neural network weights," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2020, pp. 4071–4081.

[25] F. Ma, Y. Wu. Y. Xin, and Y. Yang, "Learning with noisy labels via self-reweighting from class centroids," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 11, pp. 6275–6285, 2022.

[26] D. Arpit, S. Jastrzebski, N. Ballas, D. Krueger, E. Bengio, M. S. Kanwal, T. Maharaj, A. Fischer, A. C. Courville, Y. Bengio, and S. Lacoste-Julien, "A closer look at memorization in deep networks," in *Proc. Int. Conf. Mach. Learn. (ICML)*, vol. 70, 2017, pp. 233–242.

[27] 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. Adv. Neural Inf. Process. Syst. (NIPS)*, 2018, pp. 8536–8546.

[28] T. Kim, J. Ko, J. Choi, S.-Y. Yun *et al.*, "FINE samples for learning with noisy labels," *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, vol. 34, pp. 24 137–24 149, 2021.

[29] R. Jiang, Y. Yan, J.-H. Xue, B. Wang, and H. Wang, "When sparse neural network meets label noise learning: A multistage learning framework," *IEEE Trans. Neural Netw. Learn. Syst.*, pp. 1–15, 2022.

[30] H. Zhang, X. Xing, and L. Liu, "DualGraph: A graph-based method for reasoning about label noise," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2021, pp. 9654–9663.

[31] Z.-F. Wu, T. Wei, J. Jiang, C. Mao, M. Tang, and Y.-F. Li, "NGC: A unified framework for learning with open-world noisy data," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, 2021, pp. 62–71.

[32] A. Iscen, G. Tolias, Y. Avrithis, and O. Chum, "Label propagation for deep semi-supervised learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2019, pp. 5070–5079.

[33] X. Yu, T. Liu, M. Gong, K. Batmanghelich, and D. Tao, "An efficient and provable approach for mixture proportion estimation using linear independence assumption," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2018, pp. 4480–4489.

[34] A. Iscen, G. Tolias, Y. Avrithis, and O. Chum, "Mining on manifolds: Metric learning without labels," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2018, pp. 7642–7651.

[35] X. Yu, B. Han, J. Yao, G. Niu, I. Tsang, and M. Sugiyama, "How does disagreement help generalization against label corruption?" in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2019, pp. 7164–7173.

[36] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "Mixup: Beyond empirical risk minimization," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018, pp.1–13.

[37] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," Univ. Toronto, Toronto, ON, Canada, Tech. Rep., 2009, pp. 1–60.

[38] T. Xiao, T. Xia, Y. Yang, C. Huang, and X. Wang, "Learning from massive noisy labeled data for image classification," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2015, pp. 2691–2699.

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

[40] W. Park, D. Kim, Y. Lu, and M. Cho, "Relational knowledge distillation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2019, pp. 3967–3976.

[41] H. Song, M. Kim, D. Park, Y. Shin, and J.-G. Lee, "Learning from noisy labels with deep neural networks: A survey," *IEEE Trans. Neural Netw. Learn. Syst.*, pp. 1–19, 2022.

[42] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*.

[43] S. Zagoruyko and N. Komodakis, "Wide residual networks," in *Proc. Brit. Mach. Vis. Conf. (BMVC)*, 2016.

[44] Y. Lu and W. He, "SELC: Self-ensemble label correction improves learning with noisy labels," 2022, *arXiv:2205.01156*.

[45] C. Tan, J. Xia, L. Wu, and S. Z. Li, "Co-learning: Learning from noisy labels with self-supervision," in *Proc. ACM Int. Conf. Multimedia. (ACM MM)*, 2021, pp. 1405–1413.

[46] Y. Qu, S. Mo, and J. Niu, "DAT: Training deep networks robust to label-noise by matching the feature distributions," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2021, pp. 6821–6829.

[47] D. Patel and P. Sastry, "Adaptive sample selection for robust learning under label noise," *Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis. (WACV)*, 2023, pp. 3932–3942.

[48] L. Van der Maaten and G. Hinton, "Visualizing data using T-SNE." *J. Mach. Learn. Res*, vol. 9, pp. 2579–2605, 2008.

[49] 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 *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2017, pp. 1944–1952.

[50] E. Malach and S. Shalev-Shwartz, "Decoupling" when to update" from" how to update"," *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp.960–970.

[51] X. Ma, Y. Wang, M. E. Houle, S. Zhou, S. Erfani, S. Xia, S. Wijewickrema, and J. Bailey, "Dimensionality-driven learning with noisy labels," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2018, pp. 3355–3364.

[52] 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 *Proc. Int. Conf. Mach. Learn. (ICML)*, 2018, pp. 2304–2313.

[53] Y. Shen and S. Sanghavi, "Learning with bad training data via iterative trimmed loss minimization," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2019, pp. 5739–5748.