CLASS PROTOTYPE-BASED CLEANER FOR LABEL NOISE LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Semi-supervised learning based methods are current SOTA solutions to the noisy-label learning problem, which rely on learning an unsupervised label cleaner first to divide the training samples into a labeled set for clean data and an unlabeled set for noise data. Typically, the cleaner is obtained via fitting a mixture model to the distribution of per-sample training losses. However, the modeling procedure is *class agnostic* and assumes the loss distributions of clean and noise samples are the same across different classes. Unfortunately, in practice, such an assumption does not always hold due to the varying learning difficulty of different classes, thus leading to sub-optimal label noise partition criteria. In this work, we reveal this long-ignored problem and propose a simple yet effective solution, named **C**lass **P**rototype-based label noise **C**leaner (**CPC**). Unlike previous works treating all the classes equally, CPC fully considers loss distribution heterogeneity and applies class-aware modulation to partition the clean and noise data. CPC takes advantage of loss distribution modeling and intra-class consistency regularization in feature space simultaneously and thus can better distinguish clean and noise labels. We theoretically justify the effectiveness of our method by explaining it from the Expectation-Maximization (EM) framework. Extensive experiments are conducted on the noisy-label benchmarks CIFAR-10, CIFAR-100, Clothing1M and WebVision. The results show that CPC consistently brings about performance improvement across all benchmarks.

1 INTRODUCTION

Deep Neural Networks (DNNs) have brought about significant progress to the computer vision community over past few years. One key to its success is the availability of large amount of training data with proper annotations. However, label noise is very common in real-world applications. Without proper intervention, DNNs would be easily misled by the label noise and yield poor performance.

In order to improve the performance of DNNs when learning with noise labels, various methods have been developed (Liu et al., 2020; Li et al., 2020a; Reed et al., 2014; Nishi et al., 2021). Among them, semi-supervised learning based methods (Nishi et al., 2021; Li et al., 2020a) achieve the most competitive results. The semi-supervised learning methods follow a two-stage pipeline. They first model the loss distribution of training samples to construct a noise cleaner based on the "small-loss prior" (Han et al., 2020), which says in the early stage of training, samples with smaller cross-entropy losses are more likely to have clean labels. The prior is widely adopted and demonstrated to be highly effective in practice (Han et al., 2020). Given the noise cleaner, the training samples are divided into a labeled clean set and an unlabeled noise set. Then, semi-supervised learning strategies like MixMatch (Berthelot et al., 2019) are employed to train DNNs on the divided dataset.

The key to their performance lies in the accuracy of the label-noise cleaner (Cordeiro et al., 2022). Usually, a single Gaussian Mixture Model (GMM) (Li et al., 2020a) is used to model the loss distribution of all the training samples across different categories. However, this modeling procedure is class-agnostic, which assumes a DNN model has the same learning speed to fit the training samples in different categories, thus the same loss value on samples in different categories can reflect the same degree of noise likelihood.

Unfortunately, such assumption does not hold in practise. In Fig. 1, we present the cross-entropy loss distribution of training samples at the end of DNNs warm-up period. We conduct Kolmogorov-

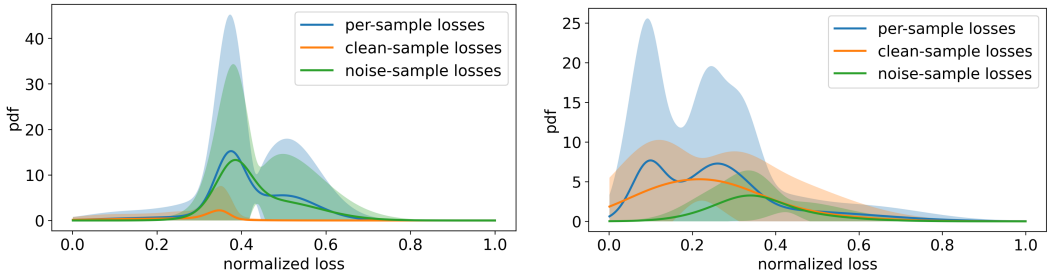

Figure 1: Loss distribution of samples in CIFAR-100 with 90% symmetric noise at epoch 30 (left) and CIFAR-10 with 40% asymmetric noise at epoch 10 (right), where the curves indicate mean probability density over all the categories while the shadow indicates the 95% confidence interval. The loss distribution for each class deviates significantly from the average loss distribution.

Smirnov test (Massey Jr, 1951) to quantify the loss distribution difference between the samples in each class and samples in the whole dataset. The results show that for 54% categories in CIFAR-100 under 90% symmetric noise, the p-value is lower than 0.05[1] for the hypothesis test that the probability distribution of clean samples in the class is the same with the probability distribution of clean samples in the whole dataset, while the number in the case of noise samples is 53%. Therefore, the class-agnostic label noise cleaner, which establishes a overly rigid criterion shared by all the classes, would introduce more noise samples to the clean set while reject clean samples, and consequently get the model perform poorly. A straightforward remedy to the problem is to fit distinct GMMs to losses of samples in different classes respectively, yielding a class-aware GMM cleaner. Nevertheless, this class-aware modeling strategy implicitly assumes that label noise is existed in every class. In the case of asymmetric noise *e.g.,* CIFAR10-asym40%, where samples in parts of classes are clean, such a naive strategy would classify most of hard samples in the clean classes as noise, and results in negative affect on model training.

Considering that images in the same category should share similar visual representations, the similarity between a sample and the cluster center (*e.g.*, class prototype) of its labeled class is helpful for recognizing label noise. In this paper, we propose a simple **C**lass **P**rototype-based label noise **C**leaner (**CPC**) to apply class-aware modulation to the partitioning of clean and noise data, which takes advantage of intra-class consistency regularization in feature space and loss distribution modeling, simultaneously. CPC learns embedding for each class, *i.e.,* class prototypes, via intra-class consistency regularization, which urges samples in the same class to gather around the corresponding class prototype while pushes samples not belonging to the class away. Unlike the aforementioned naive class-aware GMM cleaner, CPC apply class-aware modulation to label noise partitioning via representation similarity measuring without assuming that label noise is existed in every class, which is more general for different label noise scenarios. Meanwhile, CPC leverages the "small-loss prior" to provide stronger and more robust supervision signals to facilitate the learning of prototypes.

We plug CPC to the popular DivideMix(Li et al., 2020a) framework, which iterates between label noise partitioning and DNNs optimization. With the stronger label noise cleaner in the first stage, DNNs can be trained better in the second stage, which would further improve the learning of class prototypes. We theoretically justify the procedure from Expectation-Maximization algorithm perspective, which guarantees the efficacy of the method. We conduct extensive experiments on multiple noisy-label benchmarks, including CIFAR-10, CIFAR-100, Clothing1M and WebVision. The results clearly show that CPC effectively improves accuracy of label-noise partition, and brings about consistently performance improvement across all noise levels and benchmarks.

The contribution of our work lie in three folds: (1) We reveal the long-ignored problem of *class-agnostic* loss distribution modeling that widely existed in label noise learning, and propose a simple yet effective solution, named Class Prototype-based label noise Cleaner (CPC); (2) CPC takes advantage of loss distribution modeling and intra-class consistency regularization in feature space si-

[1]A p-value < 0.05 suggests the probability that the class-wise loss distribution are the same with the global loss distribution is lower than 5%.

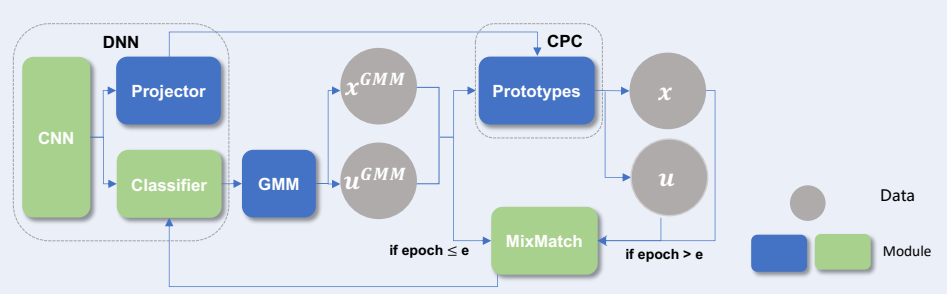

Figure 2: Illustration of the training pipeline in a single epoch. Blue modules are utilized in the first stage, where we update the prototypes in CPC and partition the training data. Green modules are utilized in the second stage, where the DNN model is trained based on the partitioned data.

multaneously, which can better distinguish clean and noise labels; (3) Extensive experimental results show that our method achieves competitive performance compared to current SOTAs.

2 Related Work

Recent advances in robust learning with noisy labels can be roughly divided into three groups. (a) **Label correction methods** aim to translate wrong labels into correct ones. Early studies rely on an auxiliary set with clean samples for clean label inference (Xiao et al., 2015a; Vahdat, 2017; Li et al., 2017b; Lee et al., 2018). Recent efforts focus on performing label correction procedures without supervision regarding clean or noise labels. (Yi & Wu, 2019a; Tanaka et al., 2018) propose to jointly optimize labels during learning model parameters. Li et al. (2020b) propose to correct corrupted labels via learning class prototypes and utilize the pseudo-label generated by measuring the similarity between prototypes and samples to train model. Wu et al. (2021) and Li et al. (2021) introduce neighbouring information in feature space to correct noise label, and propose a graph-based method and a class prototype-based method, respectively. (b) **Sample selection methods** select potential clean samples for training to eliminate the effect of noise labels on learning the true data distribution. (Han et al., 2018; Jiang et al., 2018; 2020; Yu et al., 2019) involve training two DNNs simultaneously and focus on the samples that are probably to be correctly labeled. (c) **Semi-supervised learning methods** conceal noise labels and treat these samples as unlabeled data (Ding et al., 2018). DivideMix (Li et al., 2020a) is a typical algorithm among these works, which compromises an unsupervised label noise cleaner that divides the training data to a labeled clean set and an unlabeled noise set, followed by semi-supervised learning that minimize the empirical vicinal risk of the model. Inspired by DivideMix, a series of methods (Cordeiro et al., 2022; Nishi et al., 2021; Cordeiro et al., 2021) are proposed, which achieve SOTA performance. However, all these methods rely on the class-agnostic loss distribution modeling to achieve the label noise cleaner, which hinders the performance of the model. The class-agnostic loss distribution modeling implicitly assumes a DNN model has the same learning speed to memory training samples in different categories. However, in reality, the memorization speed are actually different and will cause the the problem of under learning in hard classes as revealed by Wang et al. (2019). In this paper, we focuses on another problem, *i.e.,* class agnostic loss distribution modeling problem caused by the issue in the context of label noise cleaner. In our method, we propose the simple yet effective class prototype-based label noise cleaner to solve the problem. Besides, compared to previous prototype-based label noise learning methods (Li et al., 2020b; 2021), our method are different from them in two folds: (1) we utilize prototypes as label noise cleaner to effectively improve the semi-supervised learning based methods; (2) CPC takes advantage of both loss distribution modeling and intra-class consistency regularization in feature space simultaneously which learns better prototypes.

3 Preliminary

In label noise learning, given a training set $D = (X, Y) = \{(x_i, y_i)\}_{i=1}^{N}$, where x_i is an image and $y_i \in \{1, 2, ..., K\}$ is the annotated label over K classes, the label y_i could differ from the unknown true label $\hat{y}_i$. In this paper, we follow the popular label noise learning framework DivideMix (Li

et al., 2020a), which first warms up the model for a few epochs by training on all the data using the standard cross-entropy loss, and then trains the model by iterating a two-stage pipeline. The pipeline comprises an unsupervised label cleaner Q to divide training samples into a labeled set for clean data $\mathcal{X}$ and an unlabeled set for noise data $\mathcal{U}$, followed by a semi-supervised learning stage that trains the model to minimise the empirical vicinal risk (EVR) (Zhang et al., 2017):

$$\ell_{EVR} = \frac{1}{|\mathcal{X}'|} \sum_{\mathcal{X}'} \ell_{\mathcal{X}'}(p(\tilde{y_i'}|x_i'), y_i') + \frac{\lambda}{|\mathcal{U}'|} \sum_{\mathcal{U}'} \ell_{\mathcal{U}'}(p(\tilde{y_i'}|x_i'), y_i'), \tag{1}$$

where $\mathcal{X}'$ and $\mathcal{U}'$ indicate MixMatch (Berthelot et al., 2019) augmented clean and noise set. $l_{\mathcal{X}'}$ and $l_{\mathcal{U}'}$ denote the losses for samples in set $\mathcal{X}'$ and $\mathcal{U}'$, which are weighted by λ. $p(\tilde{y_i'}|x_i')$ is the softmax output of DNNs, where $\tilde{y_i'}$ is the predicted label. For more details about EVR, please refer to the appendix A.1.

In Li et al. (2020a), the unsupervised label cleaner is operated under the "small-loss prior", which is widely adopted and demonstrated to be highly effective (Han et al., 2020). The prior assumes that in the early stage of training, samples with smaller cross-entropy losses are more likely to have clean labels. The well known insight behind the "small-loss prior" is that DNNs tend to learn simple patterns first before fitting label noise (Arpit et al., 2017). Given a training sample x_i and the softmax output $p(\tilde{y}_i|x_i)$ of DNNs, where $\tilde{y}_i$ is the predicted label, the cross-entropy loss $l(p(\tilde{y}_i|x_i), y_i)$ reflects how well the model fits the training sample.

To achieve the unsupervised label cleaner Q, a two-component Gaussian Mixture Model (GMM) is employed to fit the loss distribution of all training samples, *i.e.,* $\ell(p(\tilde{y}_i|x_i), y_i) \sim \phi_0\mathcal{N}(\mu_0, \sigma_0) + \phi_1\mathcal{N}(\mu_1, \sigma_1)$, where $\mu_0 < \mu_1$, and ϕ is a mixing coefficient. The component with smaller mean represents the distribution of clean samples and the other one is for noise samples. We use $z_i \in \{0, 1\}$ indicates the data is clean or not. Then, $q(z_i = 0)$ represents the clean probability of x_i, which is the posterior probability of its loss belonging to the clean component. The label cleaner is shared by training samples across different classes, which is actually *class-agnostic*. A hypothesis implicitly accompanying this loss distribution modeling method is ignored by current works, which assumes the loss distributions of clean and noise samples are consistent across different categories. Unfortunately, as illustrated in Fig.1, the hypothesis dose not hold in practise. In this paper, we propose the class prototype-based label noise cleaner which applies class-aware modulation to the partitioning of clean and noise data and improves label noise learning.

4 METHODOLOGY

4.1 OVERVIEW

Our method follows the two-stage label noise learning framework DivideMix (Li et al., 2020a) and improves the framework with the proposed CPC. CPC comprises class prototypes $C = \{c_k \in \mathbb{R}^{1\times d} | k = 1, 2, ..., K\}$, where c_k indicates the prototype of k-th class and d is the dimension of prototype embedding. Our DNN model consists of a CNN backbone, a classifier head and a projection layer. The backbone maps an image input x_i to a feature vector $v_i \in \mathbb{R}^{1\times D}$. The classifier takes v_i as input and outputs class prediction $p(\tilde{y}_i|x_i)$. The projection layer serves to project the high dimension feature v_i to a low-dimensional embedding $v_i' \in \mathbb{R}^{1\times d}$, where $d < D$.

As shown in Fig. 2, we update the DNN as well as the CPC by iterating a two-stage training pipeline in every epoch. In the first stage, we update CPC as well as the projector in DNN, and utilize the updated CPC to partition label noise. We first calculate the cross-entropy loss of every training sample and fits a GMM to the losses. We utilize the GMM as a label noise cleaner to get a labeled clean set $\mathcal{X}^{GMM}$ and a unlabeled noise set $\mathcal{U}^{GMM}$. The data partition $\mathcal{X}^{GMM}$ and $\mathcal{U}^{GMM}$ are utilized to update the prototypes in CPC and parameters in the projector. Note that we cut off the gradient back-propagation from the projector to the CNN backbone. Then, the updated CPC is employed to re-divide the training data into another two set $\mathcal{X}$ and $\mathcal{U}$. In the second stage, we train DNN model to minimise the EVR in Eq. (1) with data partitioned by the cleaner. In the first e epochs, we wait CPC to warm up, and minimise the EVR of DNNs based on training data partitioned by the GMM cleaner. After the e-th epoch, the label noise estimation results of CPC, *i.e.,* $\mathcal{X}$ and $\mathcal{U}$ are employed to train DNNs, while the estimation results of GMM cleaner are only used to update

prototypes in CPC. In inference, we utilize DNN classifier for image recognition, directly. In A.5, we further delineate the full framework.

4.2 CLASS PROTOTYPE-BASED LABEL NOISE CLEANER

In order to apply class-aware modulation to the label noise partitioning, we propose to learn an embedding space where samples from the same class are aligned with their class prototypes, and leverage the prototypes to recognize noise labels. The prototypes are typically learnt with intra-class consistency regularization, which urges samples in the same class to align with the corresponding class prototype while keeping samples not belonging to the class away. Previous methods (Wang et al., 2022; Li et al., 2020b) apply the intra-class consistency regularization to prototype learning via unsupervised contrastive objectives, *e.g.,* prototypical contrastive objective (Li et al., 2020c), where the unsupervised training labels are typically determined by the similarity between samples and prototypes. The accuracy of the training labels are highly depends on the quality of representation learnt by the CNN encoder, which would be too low to effectively update the prototypes, especially in the early stage of training. In contrast, we empirically find that the GMM cleaner, which is operated under the well evaluated "small-loss prior", are not as sensitive as the prototypes to the representation quality, and can provide more robust and accurate training labels.

Therefore, we propose to take samples in clean set $\mathcal{X}^{\text{GMM}}$ as positive samples and those in noise set $\mathcal{U}^{\text{GMM}}$ as negative samples to update prototypes. Specifically, given the feature embedding v_i' of a sample x_i from $\mathcal{X}^{\text{GMM}}$, we update prototypes C as well as the parameters of the projector to maximize the score $q(z_i = 0)$ between $c_{k=y_i}$ and v_i', and minimize the score between $c_{k\neq y_i}$ and v_i' via minimize $L_{\mathcal{X}^{\text{GMM}}}$:

$$
\begin{aligned}
L_{\mathcal{X}^{\text{GMM}}} &= -\frac{1}{|\mathcal{X}^{GMM}|}\sum_{\mathcal{X}^{\text{GMM}}}\sum_{k=1}^{K}\ell_k(v_i', y_i), \quad \text{where} \\
\ell_k(v_i', y_i) &= \begin{cases} \log(\text{sigmoid}(v_i' c_k^\top)), & k = y_i, \\ \lambda^{neg}\log(1-\text{sigmoid}(v_i' c_k^\top)), & k \neq y_i, \end{cases}
\end{aligned} \tag{2}
$$

where $\lambda^{neg} = \frac{1}{K}$ weights the losses between positive pair and negative pairs to avoid under-fitting the positive samples. Given v_i' of a sample x_i from $\mathcal{U}^{\text{GMM}}$, we update prototypes c_k as well as the parameters of the projector to minimize the score $q(z_i = 0)$ between $c_{k=y_i}$ and v_i' via minimizing $L_{\mathcal{U}^{\text{GMM}}}$:

$$
L_{\mathcal{U}^{\text{GMM}}} = -\frac{1}{|\mathcal{U}^{GMM}|}\sum_{\mathcal{U}^{\text{GMM}}}\log(1-\text{sigmoid}(v_i' c_k^\top)), \text{ where } k = y_i. \tag{3}
$$

At last, for noise samples in $\mathcal{U}^{\text{GMM}}$ with high classification confidence, the samples are more likely to belong to the class predicted by DNNs, which is potentially valuable to the update of prototypes. Therefore, we collect such training samples $\mathcal{X}^P$ from $\mathcal{U}^{\text{GMM}}$ taking the averaged classification confidence of samples in $\mathcal{X}^{\text{GMM}}$ as the threshold. Specifically, given a sample in $\mathcal{U}^{\text{GMM}}$ with the label predicted by DNNs $k = \max_k(p(\tilde{y}_i|x_i))$, the sample is collected into $\mathcal{X}^P$ if $p(\tilde{y}_i|x_i)_k > average(\{p(\tilde{y}_i|x_i)_k|(x_j, y_j|y_j = k) \in \mathcal{X}^{\text{GMM}}\})$. Then, we update the prototypes and projectors to minimize $L_{(\mathcal{X}^P)}$:

$$
L_{\mathcal{X}^P} = -\frac{1}{|\mathcal{X}^P|}\sum_{\mathcal{X}^P}\log(\text{sigmoid}(v_i' c_k^\top)), \text{where } k = \max_k(p(\tilde{y}_i|x_i)). \tag{4}
$$

The overall empirical risk L_C for prototypes and the projector is as follows:

$$
L_C = L_{\mathcal{X}^{\text{GMM}}} + L_{\mathcal{U}^{\text{GMM}}} + \alpha L_{\mathcal{X}^P}, \tag{5}
$$

where α is the weight scalar.

CPC distinguishes a clean sample (x_i, y_i) with the score $q(z_i = 0) = \text{sigmoid}(v_i' c_{k=y_i}^\top)$ and the threshold τ. Samples with $q(z_i = 0) > \tau$ are classified as clean, and otherwise as noise.

4.3 THEORETICAL JUSTIFICATION ON THE EFFICACY OF CPC

We provide theoretical justification on the efficacy of CPC from the perspective of Expectation-Maximization algorithm, which guarantees that though CPC does not follow the classical prototypical contrastive objective, it can still learn meaningful prototypes and act as an effective cleaner.

We consider training data with label noise $D = (X, Y) = (x_i, y_i)_{i=1}^N$ as the observable data, and $Z \in \{0, 1\}^N$ as the latent variable, where $z_i = 0$ *iff* (x_i, y_i) is clean (*i.e.,* $y_i = \hat{y}_i$). The prototypes C in the cleaner are taken as parameters expected to be updated. Then, the negative log likelihood for D given C is as follows:

$$NLL(D|C) = -\sum_D \log \sum_{z_i \in \{0,1\}} p(x_i, y_i, z_i|C) = -\sum_D \log \sum_{z_i \in \{0,1\}} q(z_i) \frac{p(x_i, y_i, z_i|C)}{q(z_i)}, \quad (6)$$

where $q(z_i) = p(z_i|x_i, y_i, C)$. According to the Bayes theorem and Jensen's inequality , we have

$$\begin{aligned} NLL(D|C) &= -\sum_D \log \sum_{z_i \in \{0,1\}} q(z_i) p(x_i, y_i|C), \\ &\leq -\sum_D \sum_{z_i \in \{0,1\}} q(z_i) \log p(x_i, y_i|C) \\ &= -\sum_D \sum_{z_i \in \{0,1\}} q(z_i) \log p(y_i|C, x_i) + const, \end{aligned} \quad (7)$$

where $-\sum_D \sum_{z_i \in \{0,1\}} q(z_i) \log p(y_i|C, x_i)$ is the upper bound of $NLL(D|C)$. Typically, we can adopt the EM algorithm to find the prototypes C that minimize the upper bound by iterating:

E-step: Compute a new estimate of $q(z_i)$ (*i.e.,* clean or noise) according to prototypes C^{old} from the last iteration:

$$q(z_i) = p(z_i|x_i, y_i, C^{old}). \quad (8)$$

M-step: Find the prototypes C that minimizes the bound:

$$C^{new} = \arg\min_C -\sum_D \sum_{z_i \in \{0,1\}} q(z_i) \log p(y_i|C, x_i) \quad (9)$$

In our method, in order to introduce the "small-loss prior" to provide stronger and more robust supervision signals to the learning of CPC, in the **E-step**, we estimate the distribution of clean or noise of samples, denoted as $q(z'_i)$, via the GMM cleaner instead of $q(z_i)$ in Eq. (8). And consequently, we replace the $q(z_i)$ in Eq. (9) to $q(z'_i)$ and find the prototype C minimize the bound. Next, we provide the justification that the EM algorithm still work by proving that $q(z'_i)$ can be considered as an approximation to $q(z_i)$ in our framework.

In our method, $q(z'_i) = p(z'_i|l(p(\tilde{y}_i|x_i), y_i))$, where $\tilde{y}_i \sim p(\tilde{y}_i|x_i, \theta)$, which is the label predicted by the DNN parameterized by θ. As introduced in section 4.1, in the first stage of each epoch, the CPC's estimation results $z_i \sim q(z_i)$ are utilized to divide training samples into a labeled set for clean data $\mathcal{X} = \{(x_i, y_i)|z_i = 0\}$ and an unlabeled set for noise data $\mathcal{U} = \{(x_i, y_i)|z_i = 1\}$. Then the parameters of DNNs, which we denote as θ, are optimized using Eq. (1) in the second stage. There exists an optimal θ^* with respected to z_i, with which the softmax output $p(\tilde{y}_i|x_i)$ of DNNs satisfies:

$$\ell(p(\tilde{y}_i|x_i), y_i) = 0, \text{ if } z_i = 0, \text{otherwise } 1, \quad (10)$$

where $\ell(p(\tilde{y}_i|x_i), y_i)$ is the cross-entropy loss between the network prediction and the annotated label. With these loss values, the subsequent GMM cleaner can easily distinguish samples of $\mathcal{X}$ from samples of $\mathcal{U}$. In other words, under the optimal θ^*, the estimation of the GMM cleaner would be consistent with the partition of CPC, *i.e.,* $z'_i = z_i$. In practice, in each epoch, we takes the θ optimized to minimize Eq. (1) as an approximation to the optimal θ^* with respect to z_i, and consequently we can get $q(z'_i)$ as an approximation to $q(z_i)$. Therefore, we can see that with the "small loss prior" introduced into the prototype learning, the EM optimization procedure would still work, which guarantees CPC can learn meaningful prototypes and act as an effective cleaner. In appendix A.4, we further present more details and empirical results to demonstrate the approximation is hold in practice.

5 Experiments

5.1 Datasets and Implementation Details

Datasets. We evaluate our method on the following popular LNL benchmarks. For CIFAR-10 and CIFAR-100 (Krizhevsky et al., 2009), we experiment with two types of synthetic noise: symmetric

and asymmetric, which are injected into the datasets following the standard setup in (Li et al., 2020a). Clothing1M (Xiao et al., 2015b) and WebVision1.0 (Li et al., 2017a) are two large-scale real-world label noise benchmarks. Clothing1M contains 1 million images in 14 categories acquired from online shopping websites, which is heavily imbalanced and most of the noise is asymmetric (Yi & Wu, 2019b). WebVision1.0 contains 2.4 million images crawled from the web using the concepts in ImageNet-ILSVRC12 (ILSVRC12). Following convention, we compare with SOTAs on the first 50 classes of WebVision, as well as the performance after transferring to ILSVRC12.

Implementation details. We plug the proposed CPC to the DivideMix (Li et al., 2020a) framework. For Clothing1M and CIFAR-10 with asymmetric noise, we employ a single class-agnostic GMM for loss-distribution modeling. For other cases, we find that class-aware GMMs would further improve the performance of CPC. Following DivideMix, we employ ResNet18 (He et al., 2016) for CIFAR-10 and CIFAR-100, and utilize ImageNet pre-trained ResNet-50 for Clothing1M. Since previous works chose different backbones, *e.g.*, Inception-resnet v2 (Szegedy et al., 2017) and ResNet-50, we adopt the weaker one, *i.e.,* ResNet-50 according to (Zheltonozhskii et al., 2021), and train it from scratch for fair comparison. The threshold of CPC τ is set 0.5 by default for all the datasets except for the extremely imbalanced Clothing1M where it is set to 0.3. For CIFAR-10 and CIFAR-100, we train the models for 450 epochs. For the large-scale dataset Clothing1M and WebVision1.0, we train the model for 80 and 100 epochs, respectively. The warm-up periods of prototypes for all the datasets is set to the first 5% epochs after network warm-up, except in CIFAR-100 with noise ratios larger than 80% when set to 10% of total epochs. For the other settings, we simply follow the standard set-up as in DivideMix. For more implementation details, please refer to the appendix A.2 and codes in supplementary materials.

Table 1: Comparison with SOTAs on Real-world Benchmarks. Following GJS(Englesson & Azizpour, 2021), we run our method three times with different random seeds and report the mean and standard deviation of classification accuracy. $\diamondsuit$ indicates methods utilize ResNet50 for WebVision, while others utilize Inception-resnet v2. The best results are indicated with boldface.

	WebVision		WebVision → ILSVRC12		Clothing1M
	top1	top5	top1	top5	
ELR+	77.78	91.64	70.29	89.76	74.8
DivideMix	77.32	91.64	75.2	90.84	74.76
DivideMix$^{\diamondsuit}$	$76.3_{\pm 0.36}$	$90.65_{\pm 0.16}$	$74.42_{\pm 0.29}$	$91.21_{\pm 0.12}$	74.76
LongReMix	78.92	92.32	-	-	74.38
NGC	79.16	91.84	74.44	91.04	-
AugDMix	-	-	-	-	75.11
NCR$^{\diamondsuit}$	**80.5**	-	-	-	74.6
GJS$^{\diamondsuit}$	$79.28_{\pm 0.24}$	$91.22_{\pm 0.3}$	$75.5_{\pm 0.17}$	$91.27_{\pm 0.26}$	-
Baseline$^{\diamondsuit}$	$76.3_{\pm 0.36}$	$90.65_{\pm 0.16}$	$74.42_{\pm 0.29}$	$91.21_{\pm 0.12}$	$74.73_{\pm 0.02}$
Ours$^{\diamondsuit}$	$79.63_{\pm 0.08}$	$\mathbf{93.46}_{\pm 0.10}$	$\mathbf{75.75}_{\pm 0.14}$	$\mathbf{93.49}_{\pm 0.25}$	$\mathbf{75.40}_{\pm 0.10}$

5.2 Comparison with State-of-the-art methods

Real-world noise benchmarks. We evaluate our method on real-world large scale data sets, and compare our method with latest SOTA label noise learning methods, including DivideMix(Li et al., 2020a), LongReMix(Cordeiro et al., 2022), NGC(Wu et al., 2021), GJS(Englesson & Azizpour, 2021), ELR+(Liu et al., 2020), AugDMix(Nishi et al., 2021) and NCR(Huang et al., 2021). For WebVision, we measure the top1 and top5 accuracy on WebVision validation set and ImageNet ILSVRC12 validation set. We take ResNet50-based DivideMix (Zheltonozhskii et al., 2021) as baseline. As shown in Table 1, our CPC improves top1 and top5 accuracy over baseline model on WebVision by 3.33% and 2.81%, respectively. Our method achieves competitive performance on WebVision, and shows stronger transferable capability, outperforming other competitors on the ILSVRC12 validation set significantly. For Clothing1M, we apply the strong augmentation strategy (Nishi et al., 2021) to DivideMix as our baseline, and rerun the method three times. Our method achieves 75.4% accuracy on this challenging benchmark, outperforming all the other SOTAs. We also notice that though NCR achieves SOTA result on WebVision, it shows moderate performance compared to ELR+, DivideMix and AugDMix on Clothing1M containing asymmetric noise with imbalanced data distribution. It reveals that our method could be more robust across different label noise scenarios.

Table 2: Comparison with SOTAs on CIFAR-10 and CIFAR-100. Following previous work (Wu et al., 2021), we run our method three times with different random seeds and report the mean and standard deviation of classification accuracy. † indicates our baseline. * indicates semi-supervised learning based label noise learning methods. SOTA results are indicated with boldface.

	CIFAR-10 / CIFAR-100 (Sym)				CIFAR-10 (Asym)
	20%	50%	80%	90%	40%
ELR+	94.9 / 76.3	93.9. / 72.0	90.9. / 57.2	74.5 / 30.9	88.9
NCR	95.2 / 76.6	94.3 / 72.5	91.6 / 58.0	75.1 / 30.8	90.7
ProtoMix	96.4 / **80.3**	95.3 / 76.0	93.3 / 61.1	77.4 / 33.1	92.6
DivideMix*	96.1 / 77.3	94.6 / 74.5	93.2 / 60.2	76.0 / 31.5	93.4
LongReMix*	96.2 / 77.8	95.0 / 75.6	93.9 / 62.9	82.0 / 33.8	94.7
AugDMix*†	96.3 / 79.5	95.6 / 77.2	93.6 / 66.4	91.9 / 41.2	94.6
NGC	95.88 / 79.31 $\pm$0.13 / $\pm$0.35	94.54 / 75.91 $\pm$0.35 / $\pm$0.39	91.59 /62.7 $\pm$0.31 / $\pm$0.37	80.46 / 29.76 $\pm$1.97 / $\pm$0.85	90.55 $\pm$0.29
GJS	95.33 / 75.71 $\pm$0.18 / $\pm$0.25	- -	79.11 / 44.49 $\pm$0.31 / $\pm$0.53	- -	89.65 $\pm$0.37
Ours*	**96.50** / 80.22 $\pm$0.10 / $\pm$0.21	**95.64 / 79.31** $\pm$0.01 / $\pm$0.13	**94.78 / 69.56** $\pm$0.01 / $\pm$0.34	**92.55 / 54.60** $\pm$0.58 / $\pm$0.24	**94.73** $\pm$0.04

Synthetic noise benchmarks. We evaluate the performance of CPC on CIFAR-10 and CIFAR-100 datasets with symmetric label noise level ranging from 20% to 90% and asymmetric noise of rate 40%. We take AugDMix as the baseline, and compare our method with latest SOTA methods, where DivideMix, LongReMix and Aug-DMix are semi-supervised learning based methods. Following NGC and GJS, we run our method three times with different random seeds and report the mean and standard deviation. For other methods, *e.g.,* ProtoMix (Li et al., 2021), we report the best results reported in their papers. As shown in Table 2, though with a baseline method as strong as AugDMix, our method brings about performance improvement across all noise levels as well as noise types consistently, and establishes new SOTAs on CIFAR-10 and CIFAR-100. Additionally, we notice that, under asymmetric noise set-up, semi-supervised learning based methods consistently outperform other methods that achieve SOTA results on WebVision benchmark,including NGC, GJS and NCR. The results reveal that semi-supervised learning based method could be more robust to asymmetric noise, while our method achieves SOTA performance among them.

5.3 Analysis

Is CPC a better label noise cleaner? We evaluate the performance of label noise cleaner under both symmetric and asymmetric label noise set-ups. For symmetric noise, we use CIFAR-100 with 90% noise as benchmark to reveal the relationship between CPC and the significant performance improvement under this set-up. For asymmetric noise, we employ the most commonly adopted CIFAR-10-asym40% as benchmark. The AUC of clean/noise binary classification results of a cleaner is calculated as the evaluation metric. We take the original class-agnostic GMM cleaner (GMM_{agn}) proposed in DivideMix as baseline, and compare it to our CPC and the aforementioned naive class-aware GMM cleaner (GMM_{awr}). Furthermore, we also implement another version of CPC that trained based on the class-aware GMM cleaner. To distinguish these two CPC, we denote the regular one trained based on conventional class-agnostic GMM cleaner as CPC_{agn}, and the other one as CPC_{awr}. As shown in Figure 3, in both cases, the regular CPC_{agn} outperforms the baseline GMM_{agn} as well as GMM_{awr}, which demonstrates our class prototype-based method is the better label noise cleaner. As for the comparison between GMM_{agn} and GMM_{awr}, we find that in the situation of high symmetric noise, though GMM_{agn} shows better performance in the early stage of training, GMM_{awr} outperforms it in the second half stage of training. In the case of asymmetric noise, GMM_{awr}, which tend to classify hard clean samples in clean categories as noise wrongly, consistently underperforms GMM_{agn} across the whole training period. The results further prove that our class prototype-based method is the better choice for applying class-aware modulation to label noise cleaning, which is more robust across different noise types. Moreover, we find that in the case of asymmetric noise, CPC_{agn} achieves higher AUC compared to GMM_{agn}, which shows our method can partially make up for the shortcomings of GMM_{agn}. In the case of symmetric noise, we find that GMM_{agn} can further improve the performance of CPC, where CPC_{awr} achieves the best performance among the four cleaners.

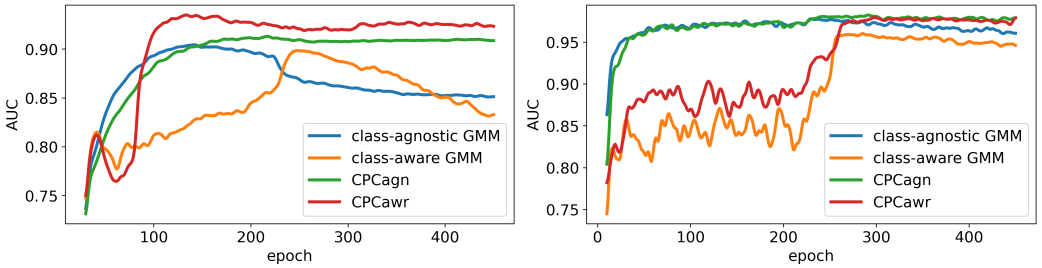

Figure 3: AUC of different label noise cleaners with respect to the training period. Left is the results on CIFAR-100 under high symmetric noise ratio 0.9. Right is the results on CIFAR-10 under medium asymetric noise ratio 0.4.

Table 3: The affect of different label noise cleaner to the final classification accuracy. The best results are indicated with boldface.

Model	CIFAR-100 sym-90%	WebVision top1	CIFAR-10 asym-40%	Clothing1M
w/ GMM$_{agn}$	41.2	76.32	94.6	74.73
w/ GMM$_{awr}$	45.6	78.66	94.18	74.18
w/ CPC$_{agn}$	48.88	79.4	**94.73**	**75.4**
w/ CPC$_{awr}$	**54.6**	**79.63**	94.29	74.36

How do different label noise cleaners affect label noise learning? We plug different cleaners to DivideMix framework, and keep all the other training settings the same as described in the implementation details. As shown in Table 3, the final performance of the model is consistent with the performance of the cleaner used. On CIFAR-100 with 90% symmetric noise, performance improvement bought about by CPC$_{agn}$ are 7.68%, while model with CPC$_{awr}$ outperforms the baseline method by 13.4%. We also report the comparison results on large-scale WebVision dataset, where the performance of different models show the same trend of change as in CIFAR-100-sym90%. As for the asymmetric noise situation, *i.e.,* CIFAR-10-asym40% and Clothing1M, model with CPC$_{agn}$, which has superior label noise partitioning capability as shown in Fig.3, achieves best performance while CPC$_{awr}$ beat GMM$_{awr}$ in both cases. The results demonstrate that CPC is helpful to train a better model in label noise learning.

Is the GMM cleaner beneficial to the learning of prototypes? In our method, we propose to leverage the GMM cleaner to facilitate the learning of prototypes via the "small loss prior". To validate the effectiveness of our method, we first compare the quality of prototypes learnt in CPC with prototypes learnt in another prototype-based label noise learning method MoPro (Li et al., 2020b). We take WebVision as benchmark and utilize prototypes to classify test samples via measuring the similarity between samples and prototypes. The results show that, on the first 50 classes of WebVision, our prototype achieves a top1 accuracy of 78.44%, while MoPro's accuracy is 72.23%, which demonstrates that our method is able to learn better prototypes. To further verify the contribution of the GMM cleaner, we remove the GMM cleaner and learn class prototypes in CPC via the typical prototypical contrastive objective as in MoPro. In experiments, we find that without the help of the GMM cleaner, the learnt prototypes generate less accurate data partition that further drawing back the overall training framework for DNNs, which proves the benefits of the GMM cleaner to our method. For more details and discussion, please refer to A.3.

6 Conclusion

In this paper, we reveal the long-ignored problem of *class-agnostic* loss distribution modeling that widely existed in label noise learning, and propose a simple yet effective solution, named Class Prototype-based label noise Cleaner (CPC). CPC takes advantage of loss distribution modeling and intra-class consistency regularization in feature space simultaneously, which can better distinguish clean and noise labels. We justify the effectiveness of our method by explaining it from the EM algorithm perspective theoretically and providing extensive empirical proves. The experimental results show that our method achieves competitive performance compared to current SOTAs.

REFERENCES

Devansh Arpit, Stanisław Jastrzebski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, et al. A closer look at memorization in deep networks. In *International conference on machine learning*, pp. 233–242. PMLR, 2017.

David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. Mixmatch: A holistic approach to semi-supervised learning. *Advances in neural information processing systems*, 32, 2019.

Filipe R. Cordeiro, Vasileios Belagiannis, Ian D. Reid, and G. Carneiro. Propmix: Hard sample filtering and proportional mixup for learning with noisy labels. *BMVC*, 2021.

Filipe R Cordeiro, Ragav Sachdeva, Vasileios Belagiannis, Ian Reid, and Gustavo Carneiro. Longremix: Robust learning with high confidence samples in a noisy label environment. *Pattern Recognition*, pp. 109013, 2022.

Yifan Ding, Liqiang Wang, Deliang Fan, and Boqing Gong. A semi-supervised two-stage approach to learning from noisy labels. *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 1215–1224, 2018.

Erik Englesson and Hossein Azizpour. Generalized jensen-shannon divergence loss for learning with noisy labels. *Advances in Neural Information Processing Systems*, 34:30284–30297, 2021.

Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Wai-Hung Tsang, and Masashi Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *NeurIPS*, 2018.

Bo Han, Quanming Yao, Tongliang Liu, Gang Niu, Ivor W Tsang, James T Kwok, and Masashi Sugiyama. A survey of label-noise representation learning: Past, present and future. *arXiv preprint arXiv:2011.04406*, 2020.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European conference on computer vision*, pp. 630–645. Springer, 2016.

Zhenyu Huang, Guocheng Niu, Xiao Liu, Wenbiao Ding, Xinyan Xiao, Hua Wu, and Xi Peng. Learning with noisy correspondence for cross-modal matching. *Advances in Neural Information Processing Systems*, 34:29406–29419, 2021.

Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *ICML*, 2018.

Lu Jiang, Di Huang, Mason Liu, and Weilong Yang. Beyond synthetic noise: Deep learning on controlled noisy labels. In *ICML*, 2020.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Kuang-Huei Lee, Xiaodong He, Lei Zhang, and Linjun Yang. Cleannet: Transfer learning for scalable image classifier training with label noise. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5447–5456, 2018.

Junnan Li, Richard Socher, and Steven C. H. Hoi. Dividemix: Learning with noisy labels as semi-supervised learning. *ICLR*, 2020a.

Junnan Li, Caiming Xiong, and Steven CH Hoi. Mopro: Webly supervised learning with momentum prototypes. *arXiv preprint arXiv:2009.07995*, 2020b.

Junnan Li, Pan Zhou, Caiming Xiong, and Steven CH Hoi. Prototypical contrastive learning of unsupervised representations. *arXiv preprint arXiv:2005.04966*, 2020c.

Junnan Li, Caiming Xiong, and Steven CH Hoi. Learning from noisy data with robust representation learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9485–9494, 2021.

Wen Li, Limin Wang, Wei Li, Eirikur Agustsson, and Luc Van Gool. Webvision database: Visual learning and understanding from web data. *arXiv preprint arXiv:1708.02862*, 2017a.

Yuncheng Li, Jianchao Yang, Yale Song, Liangliang Cao, Jiebo Luo, and Li-Jia Li. Learning from noisy labels with distillation. *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 1928–1936, 2017b.

Sheng Liu, Jonathan Niles-Weed, Narges Razavian, and Carlos Fernandez-Granda. Early-learning regularization prevents memorization of noisy labels. *Advances in neural information processing systems*, 33:20331–20342, 2020.

Frank J Massey Jr. The kolmogorov-smirnov test for goodness of fit. *Journal of the American statistical Association*, 46(253):68–78, 1951.

Kento Nishi, Yi Ding, Alex Rich, and Tobias Hollerer. Augmentation strategies for learning with noisy labels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8022–8031, 2021.

Scott Reed, Honglak Lee, Dragomir Anguelov, Christian Szegedy, Dumitru Erhan, and Andrew Rabinovich. Training deep neural networks on noisy labels with bootstrapping. *arXiv preprint arXiv:1412.6596*, 2014.

Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Thirty-first AAAI conference on artificial intelligence*, 2017.

Daiki Tanaka, Daiki Ikami, T. Yamasaki, and Kiyoharu Aizawa. Joint optimization framework for learning with noisy labels. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5552–5560, 2018.

Arash Vahdat. Toward robustness against label noise in training deep discriminative neural networks. In *NIPS*, 2017.

Haobo Wang, Ruixuan Xiao, Yixuan Li, Lei Feng, Gang Niu, Gang Chen, and Junbo Zhao. Pico: Contrastive label disambiguation for partial label learning. *arXiv preprint arXiv:2201.08984*, 2022.

Yisen Wang, Xingjun Ma, Zaiyi Chen, Yuan Luo, Jinfeng Yi, and James Bailey. Symmetric cross entropy for robust learning with noisy labels. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 322–330, 2019.

Zhi-Fan Wu, Tong Wei, Jianwen Jiang, Chaojie Mao, Mingqian Tang, and Yu-Feng Li. Ngc: a unified framework for learning with open-world noisy data. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 62–71, 2021.

Tong Xiao, Tian Xia, Yi Yang, Chang Huang, and Xiaogang Wang. Learning from massive noisy labeled data for image classification. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2691–2699, 2015a.

Tong Xiao, Tian Xia, Yi Yang, Chang Huang, and Xiaogang Wang. Learning from massive noisy labeled data for image classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2691–2699, 2015b.

Kun Yi and Jianxin Wu. Probabilistic end-to-end noise correction for learning with noisy labels. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7010–7018, 2019a.

Kun Yi and Jianxin Wu. Probabilistic end-to-end noise correction for learning with noisy labels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7017–7025, 2019b.

Xingrui Yu, Bo Han, Jiangchao Yao, Gang Niu, Ivor Wai-Hung Tsang, and Masashi Sugiyama. How does disagreement help generalization against label corruption? In *ICML*, 2019.

Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

Evgenii Zheltonozhskii, Chaim Baskin, Avi Mendelson, Alex M. Bronstein, and Or Litany. Contrast to divide: Self-supervised pre-training for learning with noisy labels. *CoRR*, abs/2103.13646, 2021. URL https://arxiv.org/abs/2103.13646.

A Appendix

A.1 Empirical Vicinal Risk

We introduce the Empirical Vicinal Risk following Cordeiro et al. (2022). In the semi-supervised learning based label noise learning framework, with the labeled set $\mathcal{X}$ and unlabeled set $\mathcal{U}$ from a cleaner, the DNNs are trained to minimise the empirical vicinal risk (EVR) (Zhang et al., 2017):

$$\ell_{EVR} = \frac{1}{|\mathcal{X}'|}\sum_{\mathcal{X}'} \ell_{\mathcal{X}'}(p(\tilde{y_i'}|x_i'), y_i') + \frac{\lambda^{(\mathcal{U}')}}{|\mathcal{U}'|}\sum_{\mathcal{U}'} \ell_{\mathcal{U}'}(p(\tilde{y_i'}|x_i'), y_i'), \tag{11}$$

where $l_{\mathcal{X}'}$ and $l_{\mathcal{U}'}$ denote the losses for set $\mathcal{X}'$ and $\mathcal{U}'$, which are weighted by $\lambda^{(\mathcal{U}')}$. $\mathcal{X}'$ and $\mathcal{U}'$ indicate MixMatch (Berthelot et al., 2019) augmented clean and noise set:

$$\begin{aligned} \mathcal{X}' &= (x_i', y_i') : (x_i', y_i') \sim f(x_i', y_i'|x_i, y_i), (x_i, y_i) \in \mathcal{X}, \\ \mathcal{U}' &= (x_i', y_i') : (x_i', y_i') \sim f(x_i', y_i'|x_i, y_i), (x_i, y_i) \in \mathcal{U}, \end{aligned} \tag{12}$$

with

$$f(x_i', y_i'|x_i, y_i) = \frac{1}{|\mathcal{X} \cup \mathcal{U}|}\sum_{\mathcal{X}\cup\mathcal{U}} \mathbb{E}_\lambda[\delta(x_i' = \lambda x_i + (1-\lambda)x_j, y_i' = \lambda y_i + (1-\lambda)y_j)], \tag{13}$$

where δ is a Dirac mass centered at (x', y'), $\lambda \sim Beta(a, a)$, and $a \in (0, +\text{inf})$.

A.2 Other Training Details

A.2.1 Training configurations

In our method, we follow most of training set-up of DivideMix(Li et al., 2020a). We present the detailed training configures as follows:

- **CIFAR-10 and CIFAR-100**. For all the experiments on CIFAR, we train our DNN model as well as class prototypes in CPC via SGD with a momentum of 0.9, a weight decay of 0.0005, and a batch size of 128. The network is trained for 450 epochs. We set the initial learning rate as 0.02, and reduce it by a factor of 10 after 225 epochs. The warm up period for the DNN is 10 epochs. The weight $\lambda^{(\mathcal{U}')}$ is set to {0,25,50,150} as in DivideMix.
- **Clthing1M**. We train our DNN model as well as class prototypes in CPC via SGD with a momentum of 0.9, a weight decay of 0.001, and a batch size of 32. The model is trained for 80 epochs. The warm up period for the DNN is 1 epoch. The initial learning rate is set as 0.002 and reduced by a factor of 10 after 40 epochs. For each epoch, we sample 1000 mini-batches from the training data. The weight $\lambda^{(\mathcal{U}')}$ is set to 0.
- **WebVision**. We train our DNN model as well as class prototypes in CPC via SGD with a momentum of 0.9, a weight decay of 0.001, and a batch size of 32. The model is trained for 100 epochs. The warm up period for the DNN is 1 epoch. The initial learning rate is set as 0.01 and reduced by a factor of 10 after 50 epochs. For each epoch, we sample 1000 mini-batches from the training data. The weight $\lambda^{(\mathcal{U}')}$ is set to 0.

A.2.2 Hyper-parameter Study

In this paper, we mainly follow the tuning procedure as in DivideMix to determine the newly introduced hyper-parameters. First of all, we initialize the hyper-parameters to $e = 5\%, \tau = 0.5, \alpha = 1$.

Table 4: The variation of performance of CPC with respect to the change of hyper-parameters. The classification accuracy of DNNs is reported. The best results are indicated with boldface.

	baseline	CPC Warm-up epochs (e)			CPC threshold (τ)			Prototypical loss weight (α)		
		5%	10%	15%	0.5	0.6	0.7	0	0.5	1
CIFAR-100(sym90%)	41.2	52.32	**54.60**	53.7	**54.60**	54.33	54.05	**54.60**	54.48	54.51
WebVision	76.3	**79.63**	79.32	79.04	**79.63**	79.52	79.36	79.16	79.44	**79.63**
CIFAR-10(asym40)	94.60	**94.73**	94.68	94.59	**94.73**	94.71	94.65	**94.73**	94.68	94.72
Clothing1M	74.73	**75.40**	75.04	74.89	75.08	75.15	**75.40**	75.35	75.28	**75.40**

Table 5: Ablation study on the contribution of GMM cleaner.The classification accuracy of DNNs is reported. The best results are indicated with boldface.

method	CIFAR-100(sym90%)	WebVision	CIFAR-10(asym40%)	Clothing1M
Baseline	41.2	76.3	94.6	74.73
CPC w/o GMM Cleaner	42.9	26.8	93.92	74.09
CPC	**54.6**	**79.63**	**94.73**	**75.4**

Then, for the large scale real world benchmark Clothing1M and WebVision, the hyper-parameter tuning is done on the validation set of Clothing1M and transferred to WebVision. For CIFAR, a small validation set with clean data is split from training data for hyper-parameter tuning. Due to the diversity of experimental set-ups, it would be an irritating task to tune hyper-parameters for each experimental set-up, respectively. Therefore, we only tune the hyper-parameters under CIFAR-100(sym80%) and CIFAR-100(sym50%), and transfer the hyper-parameters obtained under CIFAR-100(sym80%) to the noisier set-up *i.e.,* CIFAR-100(sym90%), and those obtained under CIFAR-100(sym50%) to the less challenge set-ups *i.e.,* noise ratio lower than 50% and all noise ratio on CIFAR-10.

In practical, when a clean validation set is inaccessible, it would be the difficult to tune the hyper-parameters. To shed some light to the hyper-parameter set-up in these cases, we try to conclude some empirical solutions via studying the variation of performance of CPC with respect to the newly introduced hyper-parameters on different benchmarks. According to experimental results, we find that CPC is robust in the choice of hyperparameters in the range listed in Tab.4. Generally, $e = 5\%/10\%, \tau = 0.5, \alpha = 0/1$ can be a good choice in most cases.

A.3 Discussion on the contribution of GMM cleaner to CPC

In typical prototypical contrastive objective, the unsupervised training labels are determined by similarity between samples and prototypes. Compared to it, we empirically find that GMM cleaner provides more accurate training labels for prototypes, especially in the early stage of training. For example, in CIFAR-10(asym-40%), the averaged accuracy of training labels from GMM cleaner is 9.7% higher during the CPC warming up period.

To evaluate the contribution of GMM cleaner in our framework, we further present ablation study results in Tab. 5. For *CPC w/o GMM Cleaner*, we remove the GMM cleaner and learn class prototypes in CPC with prototypical contrastive objective as in MoPro (Li et al., 2020b). In experiments, we find that without the help of the GMM cleaner, the learnt prototypes generate less accurate data partition that further drawing back the overall training framework for DNNs as shwon in Tab. 5. The situation is especially severe on the challenging benchmark with more diverse data, *e.g.,* WebVision. The results demonstrate the benefits of the GMM cleaner in our method.

To prove the superiority of our method, we also compare the quality of prototypes learnt in our method with prototypes learnt in MoPro (Li et al., 2020b) on the first 50 classes of WebVision. To evaluate the quality of prototypes learnt in CPC, we utilize the prototypes to classify test samples via measuring the similarity between samples and prototypes. We implement the experiment with the official code released by the MoPro team. The results show that our prototype achieves a top1 accuracy of 78.44%, while MoPro's accuracy is 72.23%. The result demonstrates that our method is able to learn better prototypes.

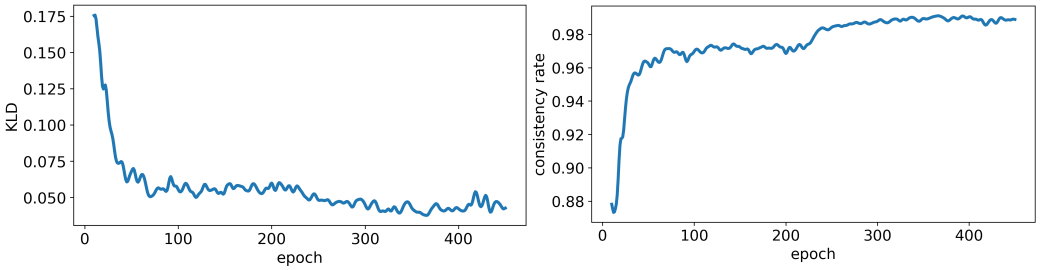

Figure 4: The left figure shows the KLD between $q(z'_i)$ and $q(z_i)$. The right figure presents the consistency rate between z'_i and z_i. Results are collected from CIFAR-10-aysm40%.

A.4 Supplementary discussion on the theoretical justification

A.4.1 Is $q(z'_i)$ a proper approximation to $q(z_i)$ in practical?

In Section 4.3, we replace the estimation of CPC $q(z_i)$ in Eq. (9) with the estimation of GMM cleaner $q(z'_i)$ and justify $q(z'_i)$ can be considered as an approximate to $q(z_i)$. To investigate if the approximation holds in practical, we calculate the K-L Divergence as well as classification consistency between $q(z'_i)$ and $q(z_i)$. As shown in Figure 4, as the training going on, the KLD between $q(z'_i)$ and $q(z_i)$ is converged and the classification consistency increases.

A.4.2 Training prototypes with L_C is an approximation to the M-step in EM

As illustrated in Section 4.3, in order to introduce the "small-loss prior" to provide stronger and more robust supervision signals to the learning of CPC, in the **E-step**, we estimate the probability distribution of clean or unclean of samples, denoted as $q(z'_i)$, via the GMM cleaner, which is an approximation to the $q(z_i)$ in Eq. (8). And consequently, we replace the $q(z_i)$ in Eq. (9) with $q(z'_i)$ and find the prototype C to minimize the bound, which makes the loss function L_C in Eq. (5) an approximation to Eq. (9). The detailed analysis on the relationship between Eq. (5) and Eq. (9) is as follows.

Firstly, we replace the estimation of CPC $q(z_i)$ in Eq. (9) with the estimation of GMM cleaner $q(z'_i)$ which is a justified approximate to $q(z_i)$:

$$\begin{aligned} C^{new} &= \arg\min_{C} -\sum_{D}\sum_{z_i\in\{0,1\}} q(z_i)\log p(y_i|C,x_i) \\ &\approx \arg\min_{C} -\sum_{D}\sum_{z'_i\in\{0,1\}} q(z'_i)\log p(y_i|C,x_i) \\ &= \arg\min_{C} -\sum_{D}[q(z'_i=0)\log p(y_i|C,x_i) + q(z'_i=1)\log p(y_i|C,x_i)] \end{aligned} \tag{14}$$

In Eq. (5), $q(z'_i)$ is quantified to 1 and 0 by the threshold τ, which makes it a "hard" version to Eq. (14). Specifically, the first term in Eq. (14) updates the prototypes C to better align the samples, that classified as clean, with labeled class prototypes. It is equivalent with the effect of Eq. (5) to positive samples, where:

$$l = \log(\text{sigmoid}(v'_i c_k^\top)), k = y_i, z'_i = 0 \tag{15}$$

where v'_i is the embedding of sample x_i. The second term in Eq. (14) updates C to prevent the samples, that classified as noise, aligning with labeled class prototypes so as to better recognize the sample as noise (*i.e.,* $z'_i = 1$), which is equivalent with the effect of Eq. (5) reducing the probability of negative samples to be recognized as clean:

$$l = \log(1 - \text{sigmoid}(v'_i c_k^\top)), k = y_i, z'_i = 1 \tag{16}$$

A.5 Illustration to the overall framework

In this paper, we plug CPC to the popular DivideMix framework. We delineate the overall training framework in Alg.1.

Algorithm 1 CPC based DivideMix

Input: Dataset $D = (X, Y)$, DNNs $\theta^{(1)}$, $\theta^{(2)}$, CPC with class prototypes $C^{(1)}$, $C^{(2)}$, clean probability τ, CPC warm-up period e.
$\theta^{(1)}, \theta^{(2)} =$ WarmUp$(X, Y, \theta^{(1)})$, WarmUp$(X, Y, \theta^{(2)})$ *//standard training to warm-up DNNs*
while $epoch <$ MaxEpoch **do**
  // get GMM cleaners by loss distribution modeling and calculate clean/noise probability distribution
  $Q^{(2)}(Z') =$GMM$(X, Y, \theta^{(1)})$
  $Q^{(1)}(Z') =$GMM$(X, Y, \theta^{(2)})$
  // calculate clean/noise probability distribution via CPC
  $Q^{(2)}(Z) =$CPC$(X, Y, \theta^{(1)}, C^{(1)})$
  $Q^{(1)}(Z) =$CPC$(X, Y, \theta^{(2)}, C^{(2)})$
  for r $\in \{1, 2\}$ **do**
    // stage1 begin
    $\mathcal{X}^{GMM(r)} = \{(x_i, y_i, w_i)|w_i = q^{(r)}(z'_i = 0), q^{(r)}(z'_i = 0) > \tau, (x_i, y_i) \in D, q^{(r)}(z'_i = 0) \in Q^{(r)}(Z' = 0)\}$
    $\mathcal{U}^{GMM(r)} = \{x_i|q^{(r)}(z'_i = 0) \leq \tau, x_i \in X, q^{(r)}(z'_i = 0) \in Q^{(r)}(Z' = 0)\}$
    Get noise labels $\{y_i|(x_i, y_i) \in D, x_i \in \mathcal{U}^{GMM(r)}\}$
    Update C^k based on Eq.5
    // stage1 end
    // stage2 begin
    if $epoch < e$ **then**
      $\mathcal{X}^{(r)} =$X$^{GMM(r)}$,$\mathcal{U}^{(r)} = \mathcal{U}^{GMM(r)}$ *//use data partition from GMM cleaner to update DNNs during the CPC warm-up period*
    else
      $\mathcal{X}^{(r)} = \{(x_i, y_i, w_i)|w_i = q^{(r)}(z_i = 0), q^{(r)}(z_i = 0) > \tau, (x_i, y_i) \in D, q^{(r)}(z_i = 0) \in Q^{(r)}(Z = 0)\}$
      $\mathcal{U}^{(r)} = \{x_i|q^{(r)}(z_i = 0) \leq \tau, x_i \in X, q^{(r)}(z_i = 0) \in Q^{(r)}(Z = 0)\}$
    end if
    Update θ^r based on Eq.11 as in standard DivideMix
    // stage2 end
  end for
  $epoch \leftarrow epoch + 1$
end while
Output: DNNs $\theta^{(1)}$, $\theta^{(2)}$