Prototypical Classifier for Robust Class-Imbalanced Learning

Tong Wei[1], Jiang-Xin Shi[1], Yu-Feng Li[1(✉)], and Min-Ling Zhang[2,3]

[1] National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China
{weit,shijx,liyf}@lamda.nju.edu.cn

[2] School of Computer Science and Engineering, Southeast University, Nanjing 210096, China
zhangml@seu.edu.cn

[3] Key Laboratory of Computer Network and Information Integration (Southeast University), Ministry of Education, Nanjing, China

Abstract. Deep neural networks have been shown to be very powerful methods for many supervised learning tasks. However, they can also easily overfit to training set biases, i.e., label noise and class imbalance. While both learning with noisy labels and class-imbalanced learning have received tremendous attention, existing works mainly focus on one of these two training set biases. To fill the gap, we propose *Prototypical Classifier*, which does not require fitting additional parameters given the embedding network. Unlike conventional classifiers that are biased towards head classes, Prototypical Classifier produces balanced and comparable predictions for all classes even though the training set is class-imbalanced. By leveraging this appealing property, we can easily detect noisy labels by thresholding the confidence scores predicted by Prototypical Classifier, where the threshold is dynamically adjusted through the iteration. A sample reweighting strategy is then applied to mitigate the influence of noisy labels. We test our method on both benchmark and real-world datasets, observing that Prototypical Classifier obtains substaintial improvements compared with state of the arts.

Keywords: Noisy labels · Class imbalance · Contrastive learning

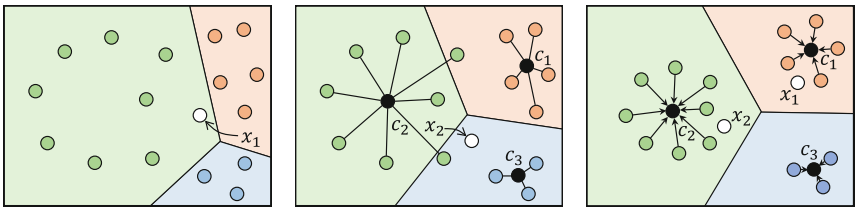
1 Introduction

Deep neural networks (DNNs) have been widely used for machine learning applications. Despite of their success, it has been shown that the training of DNNs requires large-scale labeled and *unbiased* data. However, in many real-world

T. Wei and J.-X. Shi—Co-first authors. This work was done when Tong Wei was a student at Nanjing University.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-05936-0_4.

J. Gama et al. (Eds.): PAKDD 2022, LNAI 13281, pp. 44–57, 2022.
https://doi.org/10.1007/978-3-031-05936-0_4



(a) Normal Classifier (b) Prototypical 1-NN (c) Prototypical Classifier

Fig. 1. Illustration of normal classifier and Prototypical Classifier.

applications, training set biases are prevalent [9,21,27,28], which typically have two types: i) class-imbalanced data distribution; and ii) noisy labels. For example, in autonomous driving, the vast majority of the training data is composed of standard vehicles but models also need to recognize rarely seen classes such as emergency vehicles or animals with very high accuracy. This will sometime lead to biased training models that do not perform well in practice. Moreover, large-scale high-quality data annotations are expensive and time-consuming to obtain. Although coarse labels are cheap and of high availability, the presence of noise will hurt the model performance. Therefore, it is desirable to develop machine learning algorithms that can accommodate not only class-imbalanced training set, but also the presence of label noise.

Both learning with noisy labels and class-imbalanced learning (a.k.a. long-tailed learning) have been studied for many years. When dealing with label noise, the most popular approach is sample selection where correctly-labeled examples are identified by capturing the training dynamics of DNNs [11,29]. When dealing with class imbalance, many existing works propose to reweight examples or design unbiased loss functions by taking into account the class distribution of training set [3,8,26]. However, most existing methods focus on only one of these two training set biases.

In this paper, we address both training set biases simultaneously. As shown in Fig. 1a, it is known that the classifier directly learned on class-imbalanced data is biased towards head classes [8,32] which results in poor generalization on tail classes. Moreover, using sample loss/confidence produced by biased classifiers fails to detect label noise, because both clean and noisy samples of tail classes have large loss and low confidence. To solve this problem, we propose to use *Prototypical Classifier* which is demonstrated to produce balanced predictions even through the training set is class-imbalanced. Our basic idea is that there exists an embedding in which examples cluster around a single prototype representation for each class. In order to do this, we learn a non-linear mapping of the input into an embedding space using a neural network and take a class's prototype to be the normalized mean vector of examples in the embedding space. Classification is then performed for an embedded test example by simply finding the nearest class prototype. Notably, Prototypical Classifier does not need additional learnable parameters given embedding of examples. Unfortunately, it is easy to observe that simply using prototypes for classification may lead to many

wrong predictions for samples of head classes as shown in Fig. 1b. The reason is that the representations are supposed to be modified when the classification boundaries of tail classes expand. We therefore train the neural networks to pull together embedding of examples and the prototype of their class, while pushing apart examples from prototypes of other classes. By doing this, it can avoid many mis-classifications for samples of head classes, as shown in Fig. 1c. Subsequently, we find that the confidence scores produced by Prototypical Classifier is balanced and comparable across classes. By leveraging this property, we can simply detect noisy labels via thresholding where the threshold is dynamically adjusted, followed by a sample re-weighting strategy.

In summary, our key contributions of this work are:

- We propose to learn from training set with mixed biases, which is practical but has been understudied;
- Our approach, Prototype Classifier, is simple yet powerful. It produces more balanced predictions over all classes than normal classifiers even when the training set is class-imbalanced. This property further benefits the detection of label noise.
- On both simulated datasets and a real-world dataset Webvision with label noise, Prototype Classifier achieves substantial performance improvement.

2 Related Work

Class-Imbalanced Learning. Recently, many approaches have been proposed to handle class-imbalanced training set. Most extant approaches can be categorized into three types by modifying (i) the inputs to a model by re-balancing the training data [16,22,32]; (ii) the outputs of a model, for example by post-hoc adjustment of the classifier [8,17,25]; and (iii) the internals of a model by modifying the loss function [2,6,20,23]. Each of the above methods are intuitive, and have shown strong empirical performance. However, these methods assume the training examples are correctly-labeled, which is often difficult to obtain in real-world applications. Instead, we study a realistic problem to learn from class-imbalanced data with label noise.

Label Noise Detection. Plenty of methods have been proposed to detect noisy labels [4,7,10]. Many works adopt the small-loss trick, which treats samples with small training losses as correctly-labeled. In particular, MentorNet [7] reweights samples with small loss so that noisy samples contribute less to the loss. Co-teaching [4] trains two networks where each network selects small-loss samples in a mini-batch to train the other. DivideMix [10] fits a Gaussian mixture model on per-sample loss distribution to divide the training data into clean set and noisy set. In addition, AUM [19] introduces a margin statistic to identify noisy samples by measuring the average difference between the logit values for a sample's assigned class and its highest non-assigned class. The above methods only consider class-balanced training sets, thus is not directly applicable for class-imbalanced problems. Ref. [12] observes that real-world dataset with label noise also has imbalanced number of samples per-class. Nevertheless, they only inspect a particular setup of class imbalance.

3 Prototypical Classifier with Dynamic Threshold

3.1 Motivation

Consider a binary classification problem with the data generating distribution $\mathbb{P}_{XY}$ being a mixture of two Gaussians. In particular, the label Y is either positive $(+1)$ or negative (-1) with equal probability (i.e., $\frac{1}{2}$). Condition on $Y = +1, \mathbb{P}(X \mid Y = +1) \sim \mathcal{N}(\mu_1, \sigma_1)$ and similarly, $\mathbb{P}(X \mid Y = -1) \sim \mathcal{N}(\mu_2, \sigma_2)$. Without loss of generality, let $\mu_1 > \mu_2$. It is straightforward to verify that the optimal Bayes's classifier is $f(x) = sign(x - \frac{\mu_1+\mu_2}{2})$ [30], i.e., classify x as $+1$ if $x > \frac{\mu_1+\mu_2}{2}$. This reminds us the nearest neighbor classifier, whose classification boundary is at the middle of two data points (i.e., balanced classification boundary). For general multi-class tasks, this motivates us to measure the distance of samples to class prototypes, which is empirically observed to produce balanced classification boundary even though the training set is class-imbalanced, as shown in Fig. 2.

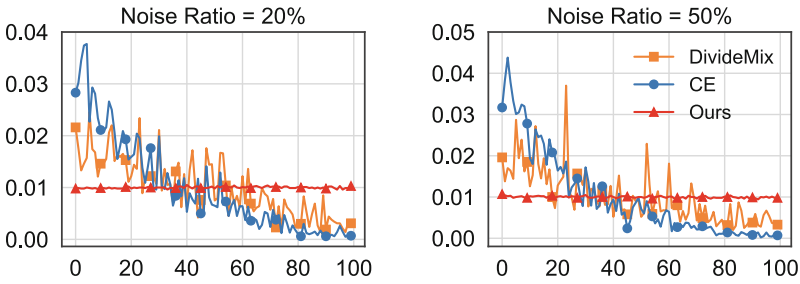


Fig. 2. Experiment on CIFAR-100-LT. x-axis is the class labels with decreasing training samples and y-axis is the marginal likelihood $p(y)$ on the test set.

In order to do this, we learn a non-linear mapping of the input into an embedding space using a neural network f_θ parameterized by θ using training set $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^N$. The class prototype is taken as the normalized mean vector of the embedded examples belonging to its class. For example, the prototype for class $k \in \{1, \ldots, K\}$ is computed as:

$$\boldsymbol{c}_k = \text{Normalize}\left(\frac{1}{|\mathcal{D}_k|}\sum_{i \in \mathcal{D}_k} f_\theta(\boldsymbol{x}_i)\right), \mathcal{D}_k = \{i \mid y_i = k\}\,. \tag{1}$$

Prototypical Classifier produces a distribution over classes for sample $\boldsymbol{x}$ based on a softmax over distances to the prototypes in the embedding space. In particular, when use cosine similarity as distance measure, we have:

$$\mathbb{P}_\theta(Y = k \mid \boldsymbol{x}) = \frac{\exp\left(f_\theta(\boldsymbol{x})^\top \mathbf{c}_k\right)}{\sum_{k'} \exp\left(f_\theta(\boldsymbol{x})^\top \mathbf{c}_{k'}\right)}. \tag{2}$$

Learning proceeds by minimizing the negative log-probability $J(\theta) = -\log \mathbb{P}_\theta(Y = k \mid \mathbf{x})$ of the true class label k via SGD. Notably, the model in Eq. (2) is equivalent to a linear model with a particular parameterization [18]. To see this, expand the term in the exponent:

$$\mathbf{c}_k^\top f_\theta(\boldsymbol{x}) = \mathbf{w}_k^\top f_\theta(\boldsymbol{x}) + b_k, \text{ where } \mathbf{w}_k = \mathbf{c}_k \text{ and } b_k = 0. \tag{3}$$

Our results indicate that Prototypical Classifier is effective despite the equivalence to a linear model. We hypothesize this is because all of the required non-linearity can be learned within the embedding function [24]. Indeed, this is the approach that modern neural network classification systems currently use.

3.2 Dynamic Thresholding for Label Noise Detection

However, the existence of label noise may hurt the representation learning of the network. To tackle this issue, it is a common practice to correct noisy labels. Let $\hat{\boldsymbol{y}} = [\hat{y}_1, \cdots, \hat{y}_K] = \mathbb{P}_\theta(Y \mid \boldsymbol{x})$ be the prediction of Prototypical Classifier, the labels are refined as stated by the following rule:

$$\tilde{y} = \begin{cases} y_i & \text{if } \hat{y}_{y_i} > \tau_t \\ \arg\max_j \hat{y}_j & \text{otherwise.} \end{cases} \tag{4}$$

In words, we deem samples as clean if the confidence scores on their original labels is greater than a threshold τ_t. It is notably that using normal classifiers cannot achieve this goal due to its biased predictions, while predictions of Prototypical Classifier are balanced and comparable. We illustrate this finding in Fig. 3. We then need to construct τ_t. Intuitively, with the increase of the optimization iteration t, the predictive confidence also increases in general, so that τ_t is also required to increase. Mathematically, we set the dynamic threshold τ_t as an increasing function of t, which is given by:

$$\tau_t = \gamma^t \tau_0. \tag{5}$$

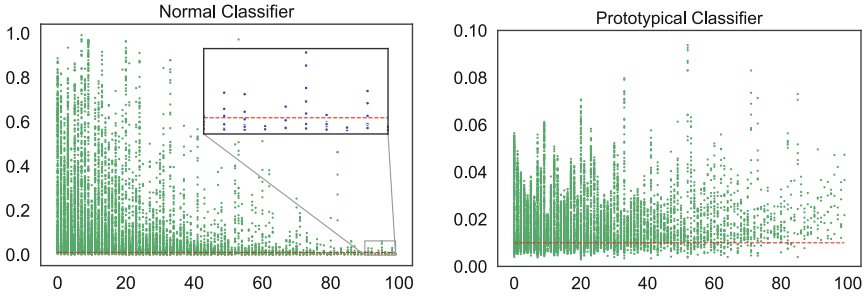


Fig. 3. Experiment on CIFAR-100-LT. x-axis is the class labels with decreasing training samples and y-axis is the confidence scores of classifiers on training set.

Here, τ_0 is the initial threshold and γ is set to 1.005 in our experiments. We provide more analysis about τ_t in supplementary materials. Lemma 1 summarizes the performance bound of the label noise detection method.

Lemma 1. *With probability at least p, the F_1-score of detecting noisy labels in $\mathcal{D}_j$ by thresholding the predictive scores of Prototypical Classifier is at least* $1 - \frac{e^{-v}\max\left(N^-, N^+\right)+\alpha}{N^-}$ *when the noise ratio is known, where* $p = \int_{-1}^{\mu^{true}-\mu^{false}-\Delta} f(t)dt$, *$f(t)$ is the probability density function of the difference of two independent beta-distributed random variables $\beta_1 - \beta_2$, where $\beta_1 \sim \text{Beta}\left(N^-, 1\right)$, $\beta_2 \sim \text{Beta}\left(\alpha + 1, N^+ - \alpha\right)$.*

Lemma 1 shows that the performance of noise detection depends on the intraclass concentration of clean samples in the embedding space (denoted by $\frac{\Delta^2}{v}$), which is optimized by the prototypical contrastive loss defined in Eq. (6). We refer the reader to Ref. [33] for the proof of Lemma 1. We further justify the effectiveness of our method in Fig. 4, which produces high F_1-score for both head and tail classes.

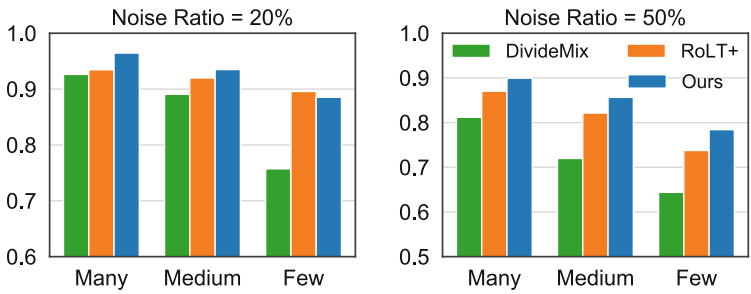


Fig. 4. Experiment on CIFAR-100-LT. We show the F_1-score of clean examples selection module for many, medium and few classes.

3.3 Example Reweighting

In standard training, we aim to minimize the expected loss for the training set, where each input example is weighted equally. Here we aim to learn a reweighting of the inputs to cope with hard mislabeled samples whose labels are not correctly refined, where we minimize a weighted loss:

$$\mathcal{L}_{\text{pc}} = \frac{-1}{\sum_{i=1}^{N} w_i} \sum_{i=1}^{N} w_i \log \frac{\exp\left(f_\theta(\boldsymbol{x}) \cdot \boldsymbol{c}_{y_i}/\tau\right)}{\sum_{k=1}^{K} \exp\left(f_\theta(\boldsymbol{x}) \cdot \boldsymbol{c}_k/\tau\right)}. \tag{6}$$

With a slight abuse of the notation, we re-define w_i to be the weight for the i-th example and τ is a temperature parameter. We expect the weights can reflect

the likelihood of examples being correctly-labeled. In that regard, we devise a weighted version for computing prototypes as:

$$\boldsymbol{c}_k = \text{Normalize}\left(\frac{1}{\sum_{i\in\mathcal{D}_k} w_i}\sum_{i\in\mathcal{D}_k} w_i f_\theta(\boldsymbol{x}_i)\right), \mathcal{D}_k = \{i \mid y_i = k\}. \tag{7}$$

Recall that, one appealing property of Prototypical Classifier is balanced predictions across all classes, as opposite to biased normal classifiers. We therefore simply set examples weights as the predicted score of Prototypical Classifier on the training label, i.e., for the i-th example, we set $w_i = \mathbb{P}_\theta(Y = y_i \mid \boldsymbol{x}_i)$ where y_i is the training label of $\boldsymbol{x}_i$. For samples whose labels are rectified, we update their weights by $w' = \frac{\tau_t - w}{2}$ to reflect the uncertainty. The modified example weights are always positive since the label is refined if and only if $w = \mathbb{P}(Y = y_i \mid \boldsymbol{x}_i) \leq \tau_t$. The optimization of $\mathcal{L}_{\text{pc}}$ is realized by contrastive learning, which has been demonstrated effective in learning representations [13]. Observing that the presence of label noise may have negative effect on representation learning, we train networks to optimize the unsupervised contrastive loss, which does not use the biased training labels. The basic idea of unsupervised contrastive learning is to pull together two embeddings of the same example, while pushing apart from other examples. Formally, let $\boldsymbol{z}_i = f_\theta(\boldsymbol{x}_i)$ and $\boldsymbol{z}'_i$ be the embedding of augmented version of $\boldsymbol{x}_i$, the unsupervised contrastive loss is computed as:

$$\mathcal{L}^i_{\text{cc}} = -\log \frac{\exp\left(\boldsymbol{z}_i \cdot \boldsymbol{z}'_i/\tau\right)}{\sum_{b=0}^{B} \exp\left(\boldsymbol{z}_i \cdot \boldsymbol{z}'_b/\tau\right)}, \tag{8}$$

where τ is a scalar temperature parameter and B is mini-batch size.

Given the above definitions and denoting $\mathcal{L}^{\text{ce}}$ as conventional cross-entropy loss, the overall training objective is written as:

$$\mathcal{L} = \mathcal{L}^{\text{ce}} + \lambda_1 \mathcal{L}^{\text{cc}} + \lambda_2 \mathcal{L}^{\text{pc}}, \tag{9}$$

where hyperparameters λ_1 and λ_2 are trade-off parameters. We adopt DNNs as feature extractor and a linear layer as projector to generate latent feature representation $\boldsymbol{z}_i$. Another linear layer following the feature extractor is used as classifier. When minimizing $\mathcal{L}_{\text{pc}}$, we apply mixup [31] to improve the generalization which has been shown to be effective for learning with noisy labels [29].

4 Experiments

We perform experiments on CIFAR-10 and CIFAR-100 datasets by controlling label noise ratio and imbalance factor of the training set. Additionally, we perform experiments on a commonly used dataset Webvision with real-world label noise.

4.1 Results on Simulated Datasets

Class-Imbalanced Dataset Generation. Formally, for a dataset with K classes and N training examples for each class, by assuming the imbalance factor is ρ, the number of examples for the k-th class is set to $N_k = N/\rho^{\frac{k-1}{K-1}}$.

Label Noise Injection. Let Y denote the variable for the clean label, $\bar{Y}$ the noisy label, and X the instance/feature, the transition matrix $T(X = x)$ is defined as $T_{ij}(X) = \mathbb{P}(\bar{Y} = j \mid Y = i, X = x)$. In this work, we follow the setup in RoLT+ [28] by setting $T(X = x)$ according to the estimated class priors $\mathbb{P}(y)$, e.g., the empirical class frequencies in the training dataset. Formally, given the noise proportion $\gamma \in [0, 1]$, we define:

$$T_{ij}(X) = \mathbb{P}(\bar{Y} = j \mid Y = i, X = x) = \begin{cases} 1 - \gamma & i = j \\ \frac{N_j}{N - N_i}\gamma & \text{otherwise.} \end{cases} \tag{10}$$

Here, N is the size of training set and N_j is frequency of class j.

Table 1. Test accuracy (%) on CIFAR-10. * denotes ensemble models.

Noise ratio		0.2			0.5		
Imbalance factor		10	50	100	10	50	100
(1) CE	Best	77.86	64.38	61.79	60.72	46.50	38.43
	Last	74.00	61.38	55.69	44.29	32.69	27.78
(2) LDAM	Best	83.48	72.01	66.41	63.57	38.92	34.08
	Last	82.91	71.23	66.22	62.13	37.97	32.56
(3) LDAM-DRW	Best	84.98	76.77	73.24	69.53	49.90	42.60
	Last	84.71	75.98	72.46	68.76	47.71	40.47
(4) DivideMix*	Best	88.79	75.34	66.90	87.54	67.92	61.81
	Last	88.10	73.48	63.76	86.88	65.22	59.65
(5) RoLT+*	Best	87.95	77.26	72.31	**88.17**	**75.11**	64.42
	Last	87.54	75.90	69.12	**87.45**	**73.92**	61.15
(6) Prototypical Classifier	Best	**90.92**	**84.12**	**79.54**	84.04	71.44	**66.33**
	Last	**90.81**	**83.71**	**78.34**	83.51	71.44	**64.69**

Result. We train the PreAct ResNet-18 network using SGD optimizer with momentum 0.9 for all methods. We set $\lambda_1 = 1$ and $\lambda_2 = 5$. We use $\tau_0 = 0.1$ for CIFAR-10 and $\tau = 0.01$ for CIFAR-100. Tables 1 and 2 respectively summarize the results for CIFAR-10 and CIFAR-100 datasets. We compare our methods with several commonly used baselines for long-tailed learning (1–3) and learning with noisy labels (4–5). As shown in the results, previous methods dreadfully degrade their performance as the noise ratio and imbalance factor increase, while our methods retain robust performance. In particular, compared with CE, Prototypical Classifier improves the test accuracy by 9% on average. It can be observed that the improvement becomes more significant when the noise ratio is high, benefiting from proposed noise detection method.

As DivideMix [10] and RoLT+ [28] are two strong baselines in this task, (4) and (5) obtain much higher performance than (1–3), particularly when noise ratio is high. Although (4) and (5) use an ensemble of two networks, our method (6) outperforms them in most cases. On CIFAR-100, Prototypical Classifier achieves the best results among all the approaches and outperforms others by a large margin for both head and tail classes in Fig. 5.

Table 2. Test accuracy (%) on CIFAR-100. * denotes ensemble models.

Noise ratio		0.2			0.5		
Imbalance factor		10	50	100	10	50	100
(1) CE	Best	45.97	33.41	29.85	28.70	18.49	16.24
	Last	45.75	33.12	29.58	23.70	16.56	14.19
(2) LDAM	Best	47.30	35.70	32.67	27.86	17.62	15.68
	Last	47.12	35.50	32.60	24.20	17.50	14.73
(3) LDAM-DRW	Best	47.85	36.29	33.38	27.86	17.91	15.68
	Last	47.68	36.01	32.99	24.45	17.81	15.07
(4) DivideMix*	Best	63.79	49.64	43.91	49.35	36.52	31.82
	Last	63.17	48.37	42.59	48.87	35.72	31.05
(5) RoLT+*	Best	64.22	51.01	45.35	53.31	39.78	35.29
	Last	63.31	49.40	43.16	52.44	39.27	34.43
(6) Prototypical Classifier	Best	**65.23**	**51.73**	**47.38**	**57.65**	**42.51**	**38.42**
	Last	**65.14**	**51.46**	**47.12**	**57.65**	**42.51**	**38.36**

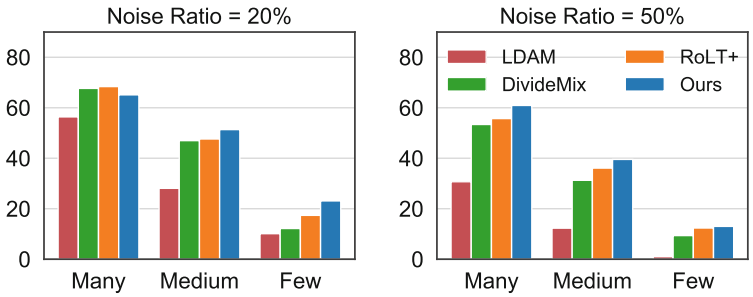


Fig. 5. Experiment on CIFAR-100-LT. We show the accuracy for many (#inst >100), medium (#inst $\in [20, 100]$) and few (#inst < 20) classes.

4.2 Results on Real-World Dataset

We test the performance of our method on a real-world dataset. WebVision [14] contains 2.4 million images collected from Flickr and Google with real noisy and class-imbalanced data. Following previous literature, we train on a subset, mini WebVision, which contains the first 50 classes. In Table 3, we report results comparing against state-of-the-art approaches, including MentorNet [7], Co-teaching [4], ELR [15], HAR [1], and DivideMix [10]. We use InceptionResNet-v2 for all methods. We set $\tau_0 = 0.05$, $\lambda_1 = 1$ and $\lambda_2 = 2$ in all experiments. From the results, we can see that, by using a single model, the proposed method achieves competitive performance with DivideMix and outperforms other baselines.

4.3 Ablation Studies

We examine the effectiveness of the each module of our method by removing it and comparing its performance with the full framework. The results are reported

Table 3. Accuracy (%) on WebVision and ImageNet. * denotes ensemble models.

		MentorNet	Co-teaching	ELR	HAR	DivideMix*	**Ours**
Webvision	top1	63.00	63.58	76.26	75.5	**77.32**	**77.32**
	top5	81.40	85.20	91.26	90.7	91.64	**92.60**
ImageNet	top1	57.80	61.48	68.71	70.3	**75.20**	75.12
	top5	79.92	84.70	87.84	90.0	90.84	**91.92**

in Table 4. Generally, it is easy to see that removing any part of the method significantly drops the performance or even fails in some cases. The performance of re-weighting and dynamic threshold shows their great effectiveness for dealing with label noise. Though we do not use the normal classifier trained via $\mathcal{L}_{ce}$, it is observed to help improve the representation learning. We have a similar observation for the unsupervised contrastive loss $\mathcal{L}_{ce}$. The strong augmentation method AugMix [5] also provides substaintial improvement.

Additionally, we also test our method on class-balanced training sets with label noise in Table 5. Prototypical Classifier outperforms other methods in most cases, even though both DivideMix and RoLT+ uses an ensemble of two networks, which shows the generality of Prototypical Classifier.

Table 4. Ablation studies. $\rho = 0.5$ and $\gamma = 100$. ▼ (▲) indicate performance loss (gain) compared with Prototypical Classifier.

Method		CIFAR-10	CIFAR-100
w/o re-weighting	Best	61.69 (▼4.64)	–
	Last	58.57 (▼6.12)	–
w/o dynamic threshold	Best	63.85 (▼2.48)	39.04 (▲0.62)
	Last	56.01 (▼8.68)	38.67 (▲0.25)
w/o mixup	Best	52.79 (▼13.54)	33.09 (▼5.33)
	Last	51.43 (▼13.26)	32.57 (▼5.79)
w/o AugMix	Best	62.51 (▼3.82)	36.11 (▼2.31)
	Last	55.21 (▼9.48)	35.68 (▼2.68)
w/o $\mathcal{L}_{\text{cc}}$	Best	55.34 (▼9.35)	32.65 (▼5.71)
	Last	53.17 (▼11.52)	32.39(▼5.97)
w/o $\mathcal{L}_{\text{ce}}$	Best	57.61 (▼7.08)	35.25 (▼3.11)
	Last	53.24 (▼11.45)	35.02 (▼3.34)

Table 5. Accuracy (%) on class-balanced datasets. * denotes ensemble models.

Noise ratio		CIFAR-10		CIFAR-100	
		0.2	0.5	0.2	0.5
DivideMix*	Best	92.79	**95.03**	77.25	73.84
	Last	92.41	**94.63**	77.03	73.42
RoLT+*	Best	92.46	94.59	78.60	74.11
	Last	92.01	94.41	78.14	73.35
Prototypical Classifier	Best	**95.93**	92.55	**79.41**	**75.50**
	Last	**95.80**	92.40	**79.41**	**75.10**

5 Conclusion

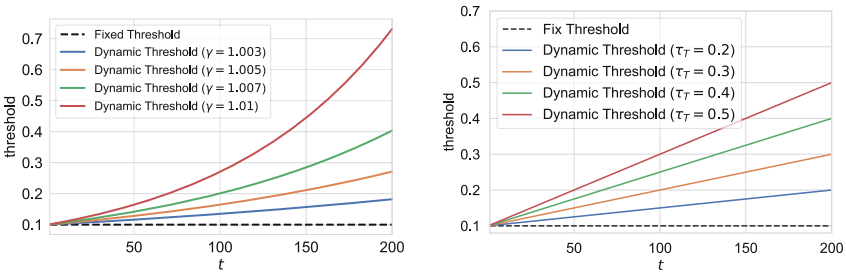
We propose Prototypical Classifier for learning with training set biases. Prototypical Classifier is shown to produce balanced predictions for all classes even when learned on class-imbalanced training set. This appealing property provides a way of detecting label noise by thresholding the predicted scores of examples. Experiments demonstrate the superiority of the proposed method. We believe Prototypical Classifier can motivate solutions to more problems with class-imbalanced training sets, for instance semi-supervised learning and self-supervised learning.

Acknowledgments. The authors wish to thank the anonymous reviewers for their helpful comments and suggestions. This research was supported by the NSFC (62176118).

A Ablations on Dynamic Threshold

Figure 6 shows a comparison of fixed threshold and the dynamic threshold τ_t with $\tau_0 = 0.1$. We consider both exponential scheduler controlled by γ and linear scheduler controlled by the threshold of last iteration τ_T.

We test the performance of different choice of parameters and the results are reported in Table 6. From the results, we have two observations: i) when using fixed threshold or the dynamic threshold grows too slow, performance drops in the last iterations because many noisy labels are incorrectly flagged as clean; and ii) when dynamic threshold grows too fast, the network cannot achieve best performance, because many clean labels are incorrectly flagged as noisy.



(a) Exponential Scheduler (b) Linear Scheduler

Fig. 6. Comparison of fixed threshold and dynamic threshold. Fix threshold $\tau = 0.1$, exponential dynamic threshold $\tau_t = 0.1\gamma^t$ and linear dynamic threshold $\tau_t = 0.1 + \frac{\tau_T - 0.1}{T}t$.

Table 6. Test accuracy (%) on CIFAR-10-LT with imbalance factor 100 and noise ratio 50%.

	Ours ($\gamma = 1.005$)	Fix	Exponential			Linear			
			1.003	1.007	1.01	0.2	0.3	0.4	0.5
Best	66.33	66.01	66.27	63.47	56.81	65.18	66.09	61.78	59.41
Last	64.69	61.37	63.57	58.93	35.84	63.40	65.11	57.84	55.12

B Results on Clean Datasets

Although our method is particularly designed learning with noisy labels, it is interesting to study its performance on clean but class-imbalanced datasets. In this experiment, we do not use sample re-weighting and label noise correction. We report the results in Table 7. For fair comparison, we do not apply AugMix in this experiment. Intriguingly, Prototypical Classifier consistently outperforms all baselines by a large margin, showing the superiority of our proposed representation learning method.

Table 7. Test accuracy (%) on clean datasets with different imbalanced factor.

	CIFAR-10			CIFAR-100		
Imbalance factor	10	50	100	10	50	100
CE	88.42	79.56	73.43	60.14	45.79	41.87
LDAM	87.43	80.32	74.50	59.84	47.61	42.59
LDAM-DRW	88.15	83.18	79.43	60.40	48.90	43.63
cRT	88.26	79.22	73.61	60.69	46.67	42.26
NCM	89.45	83.06	79.36	61.46	49.36	45.49
Prototypical Classifier	92.78	86.03	83.11	68.71	56.60	50.94

References

1. Cao, K., Chen, Y., Lu, J., Arechiga, N., Gaidon, A., Ma, T.: Heteroskedastic and imbalanced deep learning with adaptive regularization. In: ICLR (2021)
2. Cao, K., Wei, C., Gaidon, A., Aréchiga, N., Ma, T.: Learning imbalanced datasets with label-distribution-aware margin loss. In: NeurIPS, pp. 1565–1576 (2019)
3. Cui, Y., Jia, M., Lin, T., Song, Y., Belongie, S.J.: Class-balanced loss based on effective number of samples. In: CVPR, pp. 9268–9277 (2019)
4. Han, B., et al.: Co-teaching: robust training of deep neural networks with extremely noisy labels. In: NeurIPS, pp. 8536–8546 (2018)
5. Hendrycks, D., Mu, N., Cubuk, E.D., Zoph, B., Gilmer, J., Lakshminarayanan, B.: AugMix: a simple data processing method to improve robustness and uncertainty. In: ICLR (2020)
6. Jamal, M.A., Brown, M., Yang, M.H., Wang, L., Gong, B.: Rethinking class-balanced methods for long-tailed visual recognition from a domain adaptation perspective. In: CVPR, pp. 7610–7619 (2020)
7. Jiang, L., Zhou, Z., Leung, T., Li, L.J., Fei-Fei, L.: MentorNet: learning data-driven curriculum for very deep neural networks on corrupted labels. In: ICML, pp. 2304–2313 (2018)
8. Kang, B., et al.: Decoupling representation and classifier for long-tailed recognition. In: ICLR (2020)
9. Karthik, S., Revaud, J., Boris, C.: Learning from long-tailed data with noisy labels. CoRR abs/2108.11096 (2021)
10. Li, J., Socher, R., Hi, S.C.: DivideMix: learning with noisy labels as semi-supervised learning. In: ICLR (2020)
11. Li, J., Xiong, C., Hoi, S.C.: Learning from noisy data with robust representation learning. In: ICCV, pp. 9485–9494 (2021)
12. Li, J., Xiong, C., Hoi, S.C.: MOPRO: webly supervised learning with momentum prototypes. In: ICLR (2021)
13. Li, J., Zhou, P., Xiong, C., Hoi, S.C.H.: Prototypical contrastive learning of unsupervised representations. In: ICLR (2021)
14. Li, W., Wang, L., Li, W., Agustsson, E., Gool, L.V.: Webvision database: visual learning and understanding from web data. CoRR abs/1708.02862 (2017)
15. Liu, S., Niles-Weed, J., Razavian, N., Fernandez-Granda, C.: Early-learning regularization prevents memorization of noisy labels. In: NeurIPS, pp. 20331–20342 (2020)
16. Liu, Z., Miao, Z., Zhan, X., Wang, J., Gong, B., Yu, S.X.: Large-scale long-tailed recognition in an open world. In: CVPR, pp. 2537–2546 (2019)
17. Menon, A.K., Jayasumana, S., Rawat, A.S., Jain, H., Veit, A., Kumar, S.: Long-tail learning via logit adjustment. In: ICLR (2021)
18. Mensink, T., Verbeek, J.J., Perronnin, F., Csurka, G.: Distance-based image classification: generalizing to new classes at near-zero cost. IEEE Trans. Pattern Anal. Mach. Intell. **35**(11), 2624–2637 (2013)
19. Pleiss, G., Zhang, T., Elenberg, E.R., Weinberger, K.Q.: Identifying mislabeled data using the area under the margin ranking. In: NeurIPS, pp. 17044–17056 (2020)
20. Ren, J., et al.: Balanced meta-softmax for long-tailed visual recognition. In: NeurIPS, pp. 4175–4186 (2020)
21. Ren, M., Zeng, W., Yang, B., Urtasun, R.: Learning to reweight examples for robust deep learning. In: ICML, pp. 4331–4340 (2018)

22. Shen, L., Lin, Z., Huang, Q.: Relay backpropagation for effective learning of deep convolutional neural networks. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9911, pp. 467–482. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46478-7_29
23. Shu, J., et al.: Meta-weight-net: learning an explicit mapping for sample weighting. In: NeurIPS, pp. 1917–1928 (2019)
24. Snell, J., Swersky, K., Zemel, R.S.: Prototypical networks for few-shot learning. In: NeurIPS, pp. 4077–4087 (2017)
25. Tang, K., Huang, J., Zhang, H.: Long-tailed classification by keeping the good and removing the bad momentum causal effect. In: NeurIPS, pp. 1513–1524 (2020)
26. Wang, Y., Ramanan, D., Hebert, M.: Learning to model the tail. In: NeurIPS, pp. 7029–7039 (2017)
27. Wei, T., Li, Y.F.: Does tail label help for large-scale multi-label learning? IEEE Trans. Neural Netw. Learn. Syst. **31**(7), 2315–2324 (2020)
28. Wei, T., Shi, J., Tu, W., Li, Y.: Robust long-tailed learning under label noise. CoRR abs/2108.11569 (2021)
29. Wu, Z.F., Wei, T., Jiang, J., Mao, C., Tang, M., Li, Y.F.: NGC: a unified framework for learning with open-world noisy data. In: ICCV, pp. 62–71 (2021)
30. Yang, Y., Xu, Z.: Rethinking the value of labels for improving class-imbalanced learning. In: NeurIPS, pp. 19290–19301 (2020)
31. Zhang, H., Cisse, M., Dauphin, Y.N., Lopez-Paz, D.: Mixup: beyond empirical risk minimization. In: ICLR (2017)
32. Zhou, B., Cui, Q., Wei, X., Chen, Z.: BBN: bilateral-branch network with cumulative learning for long-tailed visual recognition. In: CVPR, pp. 9716–9725 (2020)
33. Zhu, Z., Dong, Z., Cheng, H., Liu, Y.: A good representation detects noisy labels. arXiv preprint arXiv:2110.06283 (2021)