Neural Networks 144 (2021) 1–10

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

P-DIFF+: Improving learning classifier with noisy labels by Noisy Negative Learning loss

QiHao Zhao [a], Wei Hu [a,*], Yangyu Huang [b], Fan Zhang [a]

[a] Department of Computer Science and Technology, Beijing University of Chemical Technology, Beijing, China
[b] Microsoft Research, Asia, Beijing, China

ARTICLE INFO

Article history:
Received 26 December 2020
Received in revised form 6 July 2021
Accepted 19 July 2021
Available online 2 August 2021

Keywords:
Classification
Noisy labels
Deep neural networks
Probability difference distribution
Noisy Negative Learning loss

ABSTRACT

Learning deep neural network (DNN) classifier with noisy labels is a challenging task because the DNN can easily over-fit on these noisy labels due to its high capability. In this paper, we present a very simple but effective training paradigm called *P-DIFF+*, which can train DNN classifiers but obviously alleviate the adverse impact of noisy labels. Our proposed probability difference distribution implicitly reflects the probability of a training sample to be clean, then this probability is employed to re-weight the corresponding sample during the training process. Moreover, Noisy Negative Learning(NNL) loss can be further employed to re-weight samples. P-DIFF+ can achieve good performance even without prior-knowledge on the noise rate of training samples. Experiments on benchmark datasets demonstrate that P-DIFF+ is superior to the state-of-the-art sample selection methods.

1. Introduction

DNN-based classifiers achieve state-of-the-art results in many researching fields. DNNs are typically trained with large-scale carefully annotated datasets. However, such datasets are difficult to obtain for classification tasks with large numbers of classes. Some approaches (Divvala, Farhadi, & Guestrin, 2014; Fergus, Fei-Fei, Perona, & Zisserman, 2010; Krause, Sapp, Howard, Zhou, Toshev, Duerig, Philbin, & Fei-Fei, 2016; Niu, Li, & Xu, 2015) provide the possibility to acquire large-scale datasets, but inevitably result in noisy/incorrect labels. Recent studies (Nettleton, Orriols-Puig, & Fornells, 2010; Zhang, Bengio, Hardt, Recht, & Vinyals, 2017) demonstrate that noisy labels for training samples will adversely affect the prediction performance of the trained DNNs classifiers. To solve this problem, many training paradigms have been investigated to learn from noisy datasets.

Some approaches (Han, et al., 2018; Liu & Tao, 2016; Menon, Van Rooyen, Ong, & Williamson, 2015; Natarajan, Dhillon, Ravikumar, & Tewari, 2013) try to estimate the noise transition matrix to correct mis-labeled samples. However, this matrix is difficult to be accurately estimated, especially for classification with many classes. Other correction methods (Ghosh, Kumar, & Sastry, 2017; Li, Yang, Song, Cao, Luo, & Li, 2017; Ma, Wang, Houle, Zhou, Erfani, Xia, Wijewickrema, & Bailey, 2018; Patrini, Rozza, Menon, Nock, & Qu, 2017) are also proposed to reduce the effect of noisy labels. Recently, some approaches focus on selecting clean samples, and update the DNNs only with these samples (Chen, Liao, Chen, & Zhang, 2019; Han, et al., 2018; Huang, Qu, Jia, & Zhao, 2019; Jiang, Zhou, Leung, Li, & Fei-Fei, 2018; Kim, Yim, Yun, & Kim, 2019; Malach & Shalev-Shwartz, 2017; Ren, Zeng, Yang, & Urtasun, 2018; Wang, Liu, Ma, Bailey, Zha, Song, & Xia, 2018; Wei, Feng, Chen, & An, 2020; Yu, Han, Yao, Niu, Tsang, & Sugiyama, 2019). Selecting clean sample is actually a hard weight method. Moreover, the soft weighting strategy (Ren et al., 2018) is also employed to adjust the contributions of training samples.

In this paper, we proposed P-DIFF+, a novel sample selection paradigm, to address the following problem: how to effectively train DNN classifiers on noisy labeled datasets? Compared with previous noisy label learning approaches, it provides a stable but very simple method for evaluating samples being noisy or clean. The main contributions of the paper are summarized as follows:

1. We propose the P-DIFF paradigm to learn DNN classifiers with noisy labels. It employs a probability difference strategy, instead of the broadly utilized small-loss strategy, to estimate the probability of a sample to be noisy. Moreover, P-DIFF employs a global probability distribution generated by accumulating samples of some recent mini-batches, so it demonstrates more stable performance than single mini-batch approaches.
2. We integrate Noisy Negative Learning(NNL) loss into P-DIFF, and propose the P-DIFF+ paradigm. NNL loss can further utilize wrong-labeled samples during training to improve the performance. Based on the sample selection

* Correspondence to: No. 15, Beisanhuan East Road, Chaoyang District, Beijing, China.
E-mail address: huwei@mail.buct.edu.cn (W. Hu).

https://doi.org/10.1016/j.neunet.2021.07.024
0893-6080/

strategy of P-DIFF+, DNNs can use cross-entropy or NNL loss functions to update itself.
3. P-DIFF+ paradigm does not depend on extra datasets, phases, models or information, and is very simple to be integrated into existing softmax-loss based classification models. Further, P-DIFF+ can also train DNNs without a given noise rate.
4. Compared with SOTA sample selection and other noise-tolerant approaches, P-DIFF+ has advantages in many aspects, including classification performance, resource consumption, computation complexity. Experiments on several benchmark datasets, including a large real-world noisy dataset cloth1M (Xiao, Xia, Yang, Huang, & Wang, 2015a), demonstrate that P-DIFF+ outperforms previous state-of-the-art sample selection approaches at different noise rates.

A preliminary version of this work was presented in ICPR 2020 (Hu, Zhao, Huang, & Zhang, 2020). The present paper substantially expands the conference paper. Some of the key additions include: (1) proposing Noisy Negative Learning(NNL) loss function; (2) integrating the NNL loss function into the P-DIFF paradigm; (3) adding more experiments on benchmark datasets to demonstrate that P-DIFF+ with NNL is superior to the SOTA sample selection methods. The codes are available at https://github.com/fistyee/P-DIFF.

2. Related work

Noisy label learning Learning with noisy datasets has been widely explored in classification (Frénay & Verleysen, 2014). Many approaches use pre-defined knowledge to learn the mapping between noisy and clean labels, and focus on estimating the noise transition matrix to remove or correct mis-labeled samples (Liu & Tao, 2016; Menon et al., 2015; Natarajan et al., 2013).

Recently, it has also been studied in the context of DNNs. DNN-based methods to estimate the noise transition matrix are proposed too. Goldberger and Ben-Reuven (2017), Sukhbaatar, Bruna, Paluri, Bourdev, and Fergus (2015), Veit, Alldrin, Chechik, Krasin, Gupta, and Belongie (2017), Xiao, Xia, Yang, Huang, and Wang (2015b) and Hendrycks, Mazeika, Wilson, and Gimpel (2018) use a small clean dataset to learn a mapping between noisy and clean annotations. Ghosh et al. (2017), Patrini et al. (2017) use noise-tolerant losses to correct noisy labels. Li et al. (2017) construct a knowledge graph to guide the learning process. Han, Yao, Niu, et al. (2018) propose a human-assisted approach which incorporates a structure prior to derive a structure-aware probabilistic model. Local Intrinsic Dimensionality is employed in Ma et al. (2018) to adjust the incorrect labels. Rank Pruning (Northcutt, Wu, & Chuang, 2017) is proposed to train models with confident samples, and it can also estimate noise rates. However, Rank Pruning only aims at binary classification. Tanaka, Ikami, Yamasaki, and Aizawa (2018) implement a simple noise-tolerant joint optimization framework to learn the probable correct labels of training samples, and to use the corrected labels to train models. However, it discarded the useful information in the original noisy label. Han, Luo, and Wang (2019) propose an extra label correction phase for its noise-tolerant method. It corrects the wrong labels and achieves good performance, but its two phases training framework affects its efficiency.

Some approaches attempt to update the DNNs only with separated clean samples, instead of correcting the noisy labels. A Decoupling technique (Malach & Shalev-Shwartz, 2017) trains two DNN models to select samples that have different predictions from these two models but it cannot process heavy noisy datasets. Very recently, weighting training samples becomes a hot topic to learn with noisy datasets.

Weighting training samples Weighting training samples (Friedman, Hastie, & Tibshirani, 2001; Kumar, Packer, & Koller, 2010; Lin, Goyal, Girshick, He, & Dollár, 2018) is also applied to select clean samples. The idea of weighting training samples is employed to train DNNs with noisy datasets too, since clean/noisy samples are usually corresponding to easy/hard samples. Ren et al. (2018) compute sample weights via minimizing loss functions on a clean validation set. Previous work manifests that during training, DNNs tend to memorize easy/clean samples firstly, and gradually memorize all samples (Arpit et al., 2017). This research (Arpit et al., 2017) justifies the widely used *small-loss criterion*: regarding samples with small training loss as clean ones. Curriculum learning (Bengio, Louradour, Collobert, & Weston, 2009) proposes to start from easy samples and go through harder samples to guide training. Based on Curriculum learning (Bengio et al., 2009), some recent works (Han, Yao, Yu, et al., 2018; Jiang et al., 2018; Wei et al., 2020; Yu et al., 2019) proposed approaches select clean training samples by using the small-loss strategy. Small-loss approaches select a proportion of training samples that have small-loss in a mini-batch, and only the selected samples are employed to update the DNNs.

MentorNet (Jiang et al., 2018) firstly trains a teacher network, then selects clean samples for guiding the training of the student network. Co-teaching (Han, Yao, Yu, et al., 2018) simultaneously trains two networks for selecting the next batch of data for each other. The next batch is chosen as the data batch, which has small-loss according to pair network. It is claimed that using one network accumulates the noise-related loss error. Co-teaching+ (Yu et al., 2019) constructed on Co-teaching combines the "disagreement" strategy and the "cross-update" strategy. JoCoR (Wei et al., 2020) also selects small-loss samples but updates the networks by Joint training and use the Co-Regularization to maximize agreement between two networks.

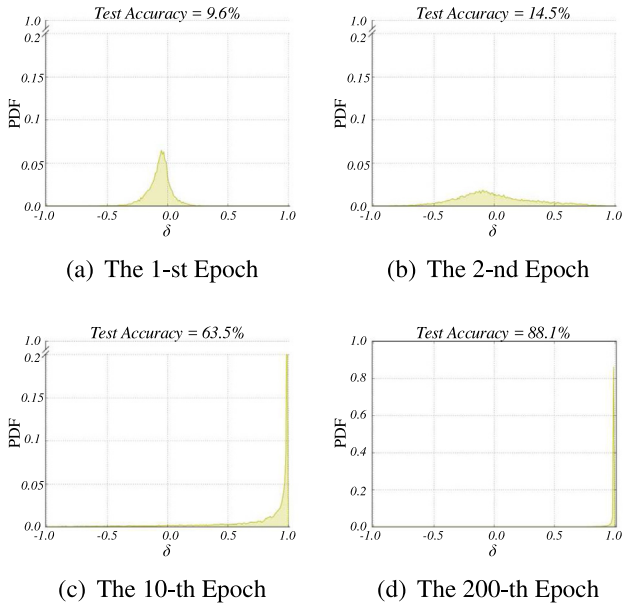
Negative Learning When DNNs are trained with images and noisy labels, wrong information is being provided to it. To solve this problem, Kim et al. (2019) suggest *Negative Learning*, an indirect learning method for training DNNs. It first generates a complementary label to each sample in addition to the ground truth. Then DNNs are trained using a complementary label as in the input image does not belong to this complementary label. In this manner, NL decreases the risk of providing incorrect information.

To conclude, noisy label learning approaches achieve better performance, but they still have some limitations and extra consumption. First of all, clean samples are selected only by considering samples in one mini-batch, which might lead to unstable selections (Han, Yao, Yu, et al., 2018). Secondly, extra models and noisy rate are required (Chen et al., 2019; Han, Yao, Yu, et al., 2018; Wei et al., 2020; Yu et al., 2019). Finally, multi-step training process are used (Huang et al., 2019; Kim et al., 2019).

In this paper, we proposed a very simple sample selection approach P-DIFF+. Compared with previous approaches, *reference sets*, *extra models*, and *multi-step training* are **not** required in P-DIFF+.

3. The proposed P-DIFF paradigm

Samples with incorrect label are referred as *noisy samples*, and their labels are referred as *noisy labels*. Noisy labels fall into two types: **label flips** where the sample has been given a label of another class within the dataset, and **outliers**, where the sample does not belong to any of the classes in the dataset. In some papers, they are named as *closed-set* and *open-set* noisy labels. As most of previous works (Han, Yao, Niu, et al., 2018; Han, Yao, Yu, et al., 2018; Huang et al., 2019; Jiang et al., 2018; Li et al., 2017; Ma et al., 2018; Patrini et al., 2017; Vahdat, 2017;



(a) The 1-st Epoch (b) The 2-nd Epoch

(c) The 10-th Epoch (d) The 200-th Epoch

Fig. 1. $DIST_{all}$ and the corresponding performance results at different training epochs.

Yu et al., 2019), we also address the noisy label problem in a **closed-set setting**. Actually, experiments in Section 3.4 on the large real-world dataset Cloth1M (Xiao et al., 2015a) demonstrate that P-DIFF is capable to process open-set noisy labels too.

3.1. Problem statement

The softmax loss is widely applied to supervise DNN classification, and can be considered as the combination of a softmax function and a cross-entropy loss. The output of the softmax function is $\vec{P} = (p_1, \ldots, p_C)$, where C is the class number. As for an input sample, $p_m \in [0, 1]$ is the predicted probability of belonging to the mth class. And $q \in \mathcal{Y} = \{1, \ldots, c\}$ be its label, $\mathbf{q} \in \{0, 1\}^c$ be its one-hot vector. The conventional objective for supervised classification is to minimize an empirical risk, such as the cross-entropy loss:

$$\mathcal{L} = -\sum_{m=1}^{C} \mathbf{q}_m log(p_m). \tag{1}$$

Generally, when we consider y as the ground truth class, p_y is encouraged to become the maximum component in $\vec{P}$. However, since q_m contains noise, enlarging p_y would lead to adverse effects on the robustness of the trained classifier. Some recently proposed methods (Han, Yao, Yu, et al., 2018; Jiang et al., 2018; Ren et al., 2018; Wei et al., 2020; Yu et al., 2019) employ the small-loss strategy to select clean samples. Actually, samples selected with the small-loss strategy are also the samples with large p_y in the softmax loss.

Different from those approaches, P-DIFF selects the clean samples based on the **probability difference distribution**. The probability difference distribution is computed with the output of the softmax function, and is presented as follows.

3.2. Probability difference distribution

Following previous sample selection approaches, such as O2UNet (Huang et al., 2019), we also remove potential noisy labels to achieve better performance, although noisy and real hard samples may not be distinguished in some cases. Our sample selection strategy is based on two key strategies: *Probability Difference* and *Global Distribution*.

3.2.1. Probability difference

In this paper, we find that the probability difference δ should be a better value to select clean samples, compared with p_y. We define the **probability difference** δ of a sample, which belongs to the yth class, as

$$\delta = p_y - p_n, \tag{2}$$

where p_n is the largest component except p_y in $\vec{P}$, so $\delta \in [-1, 1]$. Ideally, the δ value should be 1 for a clean sample. If the sample is a label flip noisy sample, we can also ideally infer that $p_y = 0$ and $p_n = 1$ ($\delta = -1$), where n is the correct label. Although we cannot achieve such results in real training, it inspires us to select clean samples according to δ values.

The small-loss strategy has been proven to be an effective way to select clean samples (Han, Yao, Yu, et al., 2018; Jiang et al., 2018; Ren et al., 2018; Yu et al., 2019). However, the small-loss strategy cannot select appropriate clean samples in some cases. For example, $\vec{P}_1 = \{0.2, 0.2, 0.2, 0.2, 0.2\}$ and $\vec{P}_2 = \{0.0, 0.2, 0.0, 0.0, 0.8\}$ are the output values of two training samples, and $y = 1$. It is clear that the $\mathcal{L}$ values of these two samples are equal because of the same $p_y = 0.2$, but the second sample has much higher probability to be wrongly labeled. Furthermore, we calculate $\delta_1 = 0.0$ and $\delta_2 = -0.6$ of two samples mentioned above, which indicates that the second sample has higher probability to be noisy. Experiments in Section 5.2 also verify the effectiveness of δ(probability difference), compared with only p_y(small-loss strategy) to select training samples.

3.2.2. Global distribution

Furthermore, only considering samples in one mini-batch (Han, Yao, Yu, et al., 2018; Jiang et al., 2018; Ren et al., 2018) reduces the stabilization of sample selection, and a global threshold is not applied too since the loss values are rapidly changed especially in early epochs. P-DIFF adopts a selection method based on a δ histogram. We compute the histogram distribution of δ for all input samples, and this global distribution, called $DIST_{all}$, is just the **probability difference distribution**. We divide the entire range $[-1, 1]$ of the distribution into H bins. We set $H = 200$ in our implementation. Let $PDF(x)$ be the ratio of samples whose δ fall into the xth bin as

$$PDF(x) = \frac{1}{N}\sum_{i=1}^{N} \begin{cases} 1 & \lceil H \cdot \frac{\delta_i + 1}{2} \rceil = x \\ 0 & else, \end{cases} \tag{3}$$

where N is the number of training samples. $PDF(x)$ means the probability distribution function of $DIST_{all}$. We then define the probability cumulative function of $DIST_{all}$ as

$$PCF(x) = \sum_{i=1}^{x} PDF(i). \tag{4}$$

Moreover, given the xth bin, we can get its value range as

$$\delta \in (2 \cdot \frac{x-1}{H} - 1, 2 \cdot \frac{x}{H} - 1]. \tag{5}$$

We perform an experiment to show this distribution. The experiment setting is presented in Section 5. We train a normal DNN model with Cifar-10. Fig. 1 shows a probability difference distribution of the DNN at different training epochs. This distribution $DIST_{all}$ is employed in our P-DIFF paradigm to learn classifier with noisy labels. In theory, the distribution $DIST_{all}$ should be computed in each mini-batch training, but it is time-consuming if the number of samples is large. In our implementation, only the δ values of samples belonging to recent M mini-batch samples

are stored to generate the distribution $DIST_{sub}$. If M is too small, $DIST_{sub}$ cannot be considered as a good approximation of $DIST_{all}$. However, a large M is not appropriate too, because the δ values of far earlier training samples cannot approximate their current values (discussed in Section 5.3).

3.3. Learning classifier with noisy labels

The basic idea of P-DIFF paradigm is trying to select clean samples based on the $DIST_{all}$. As discussed in Section 3.2, the samples with larger δ have higher probability of being clean during training, and they should have higher rate to be selected to update the training DNN model. The remaining problem is to find a suitable threshold $\hat{\delta}$.

Given a noise rate τ and a distribution $DIST_{all}$, P-DIFF drops a certain rate (τ) of the training samples that fall into the left part of $DIST_{all}$. We simply find the smallest bin number x which makes

$$PCF(x) > \tau. \tag{6}$$

Therefore, all samples falling into left of the xth bin will be dropped in training. According to Eq. (5), the δ values of these samples should be less than $2 \cdot (x-1)/H - 1$, and we can define the threshold $\hat{\delta}$ as

$$\hat{\delta} = 2 \cdot \frac{x-1}{H} - 1. \tag{7}$$

However, at the beginning of training process, the DNNs do not have the ability to classify samples correctly, so we cannot drop training samples with the rate τ throughout the whole training process. We know the DNNs will be improved as the training iteration increases. Therefore, similar with Co-teaching (Han, Yao, Yu, et al., 2018), we define a dynamic drop rate $R(T)$, where T is the number of training epoch, as

$$R(T) = \tau \cdot \min(\frac{T}{T_k}, 1). \tag{8}$$

We can see that all samples are selected at the beginning, then more and more samples are dropped as T gets larger until $T = T_k$ (a given epoch number), and the final drop rate is τ. Therefore, Eq. (6) is re-written as

$$PCF(x) > R(T). \tag{9}$$

P-DIFF updates DNN models by redefining Eq. (1) as

$$\mathcal{L} = -\omega \sum_{m=1}^{C} \mathbf{q}_m log(p_m), \tag{10}$$

where ω is the computed weight of the sample. We set $\omega = 1$ if $\delta > \hat{\delta}$, or ω set to 0.

Algorithm 1 gives the detailed implementation of our P-DIFF paradigm with a given noise rate τ.

3.4. Training without a given τ

Similar with Co-teaching, a given noise rate τ is required to compute $R(T)$ in P-DIFF (Algorithm 1). If τ is not known in advance, it can be inferred by using the validation set as Han, Yao, Yu, et al. (2018), Liu and Tao (2016). However, the rate inferred using the validation set cannot always accurately reflect the real rate in the training set. We further explore the method for learning classifiers without a pre-given noise rate τ.

According to the algorithm described above, the key of P-DIFF is to find a suitable threshold $\hat{\delta}$ to separate clean and noisy training samples. Based on the definition $\delta = p_y - p_n$, we can reasonably infer that 0 might be a candidate. Considering the gradually learning problem (see Eq. (8)), we compute the threshold as

$$\hat{\delta} = \min(\frac{T}{T_k}, 1) - 1. \tag{11}$$

In other words, all samples are employed to learn the classifier in the beginning, and with the increase of the training epoch number, some samples will be dropped. At last, only samples with $\delta > 0$ are selected as clean samples to update the DNNs.

By using the above method, *Noise Rate Estimation* can be calculated easily. DNNs remember easy/clean samples firstly, and gradually adapt to hard/noisy samples as training epochs become large (Arpit et al., 2017). When noisy label exists, DNNs will eventually memorize incorrect labels. This phenomenon, called Noise-Adaption phenomenon, does not change with the choice of training optimizations or networks. DNNs can memorize noisy labels, so we cannot only trust $\hat{\delta} = 0$. In the section, we further propose a noise rate estimation technique to achieve better performance.

According to the definition of δ, the δ value should be encouraged to be close to 1 or -1 for clean and noisy samples respectively. Therefore we propose a value ζ to evaluate the performance of the learned classifier as

$$\zeta = \sum_{x=1}^{H} (|2 \cdot \frac{x-1}{H} - 1| \cdot PDF(x)). \tag{12}$$

In fact, $\zeta \in [0, 1]$ is the expected value of $|\delta|$ in the distribution $DIST_{all}$. According to the Noise-Adaption phenomenon and the P-DIFF paradigm, a high ζ should indicate that the DNN model currently mainly memorizes correct labels. Therefore, we can reasonably infer that the proportion of noisy samples in all samples with $\delta > \hat{\delta}$ is small, and the noise rate τ can be estimated based on the above inference.

We firstly train the DNNs model until $T = T_k$ ($\hat{\delta} = 0$), then ζ is computed for each mini-batch. Once ζ is larger than a threshold (discussed in Section 5.5), all samples with $\delta < 0$ are regarded as

Algorithm 1 P-DIFF Paradigm

Input: Training Dataset D, epoch T_k and T_{max}, iteration per-epoch $Iter_{epoch}$, batch size S_{batch}, noise rate τ, batch rate M;
Output: DNN parameter $\vec{W}$;

```
Initialize W⃗;
for T = 1 to T_max do
    Compute the rate R(T) using Eq. (8);
    for Iter = 1 to Iter_epoch do
        Compute the threshold δ̂ using Eqs. (7) and (9);
        Get the mini-batch D̄ from D;
        Set the gradient G = 0;
        for S = 1 to S_batch do
            Get the S-th sample D̄(S);
            Compute P⃗ of D̄(S) using W⃗;
            Compute the δ value using Eq. (2);
            if δ > δ̂ then
                ω = 1;
            else
                ω = 0;
            G+ = ∇ℒ (see Eq. (10));
        Update DIST_sub with the computed δ values of the last
M × Iter_epoch mini-batches;
        Update the parameter W⃗ = W⃗ − η · G;
```

noisy samples to estimate the noise rate τ. With the estimated τ, the threshold $\hat{\delta}$ is then computed by using Eq. (7) instead of Eq. (11), and we train DNNs by using the method with the given noise rate τ as Algorithm 1. If ζ is always less than the threshold, we estimate τ in the end of training by regarding all samples with $\delta < 0$ as noisy samples.

4. P-DIFF+: The P-DIFF paradigm with noisy negative learning

P-DIFF selects clean samples based on the **probability difference distributions** and demonstrates excellent performance. However, in P-DIFF, a sample which are considered noisy will not contribute to the loss function described by Eq. (10). How to learn correct knowledge from noisy samples is still a pending problem in the P-DIFF paradigm.

Many methods apply diverse techniques and regularization terms with *Positive Learning*(PL) to correctly train DNNs on noisy data. PL is a typical supervised learning method which tells the DNNs that the input image belongs to this label (Chen et al., 2019; Han, Yao, Yu, et al., 2018; Huang et al., 2019; Jiang et al., 2018; Ren et al., 2018). On the contrary, NLNL (Kim et al., 2019) proposes *Negative Learning*(NL). NL firstly generates complementary label $\bar{q}$ for each sample. Then the DNNs with NL are trained with the input image does not belong to this complementary label. The complementary label is randomly selected from the labels of all categories except the ground truth q for every iteration during training. Since the probability of choosing a true label as a complementary label is low, NL can provide less error information when training DNNs. The loss function following this definition is as below:

$$\mathcal{L}_{NL} = -\sum_{m=1}^{C} \bar{\mathbf{q}}_m log(1 - p_m), \tag{13}$$

and $\bar{q} \in \mathcal{Y} = \{1, \ldots, c\}$ be its complementary label, $\bar{\mathbf{q}} \in \{0, 1\}^c$ be its one-hot vector. Eq. (13) enables the probability value of the complementary label to be optimized as zero, resulting in an increase in the probability values of other classes.

Eq. (10) is suitable for optimizing the probability value corresponding to the given label as 1 ($p_y \rightarrow 1$) to meet the purpose of PL. NL differs from PL as it optimizes the output probability corresponding to the complementary label to be far from 1 ($p_y \rightarrow 0$). PL and NL all have their drawbacks. Considering samples with noisy labels, training with PL will provide wrong information to DNNs. However, NLNL (Kim et al., 2019) requires additional algorithms to generate complementary labels and other multi-step sample selection processes to update DNNs. Therefore, in order to solve these problems and improve the generalization ability of DNNs on noisy samples, we propose *Noisy Negative Learning*(NNL), a direct learning method for training DNNs with noisy labels, and we call the P-DIFF paradigm with Noisy Negative Learning as P-DIFF+.

A comparison of the concepts between PL, NL and NNL is shown in Fig. 2. As described in the figure, when the image is a car but its label is bird, DNNs with PL are trained to recognize the image as a bird, resulting in receiving wrong information from the noisy sample. DNNs with NL are trained to admit that this image is not a dog. However, DNNs with NNL are trained to consider it is not a bird. We suggest *Noisy Negative Learning*(NNL) loss function as follows:

$$\mathcal{L}_{NNL} = -\sum_{m=1}^{C} \hat{\mathbf{q}}_m log(1 - p_m), \tag{14}$$

and $\hat{q} \in \mathcal{Y} = \{1, \ldots, c\}$ be its noisy label, $\hat{\mathbf{q}} \in \{0, 1\}^c$ be its one-hot vector. NNL optimizes the output probability corresponding to the noisy label $\hat{q}$ instead of complementary label $\bar{q}$ to reach

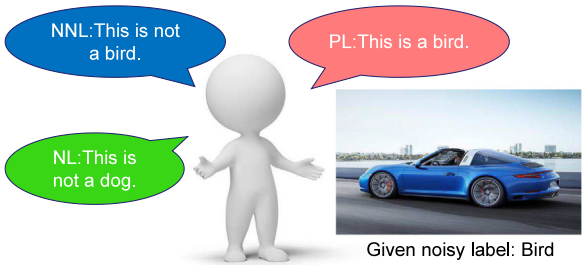


Fig. 2. The concept of a comparison between *Positive Learning*(PL), *Negative Learning*(NL) and *Noisy Negative Learning*(NNL). The sample pictured here is car and it is labeled bird. (a) PL identifies the sample as a bird, resulting in receiving the wrong information from the noisy sample (red balloon). (b) NL thinks it is not a dog (green balloon). (c) NNL considers it is not a bird and is more likely to provide the correct information to the DNNs (blue balloon). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

0 ($p_y \rightarrow 0$) eventually. Unlike the NL method, in which DNNs are trained using complementary labels, NNL does not need complementary labels. In addition, NNL can be easily integrated into P-DIFF. As described in Section 3.3, our P-DIFF's sample selection strategy uses the value of ω to infer whether the sample is clean during training. When the value of ω set to 1, which indicates DNNs consider the sample to be clean, we choose the cross-entropy loss function (Eq. (1)) to update our DNNs. Otherwise, we choose the NNL loss function (Eq. (14)). NNL uses noisy samples directly, which avoids memorizing noisy labels and helps DNNs learn the correct information. Therefore, Eq. (10) is eventually modified to

$$\mathcal{L} = -\sum_{m=1}^{C} \begin{cases} \mathbf{q}_m log(p_m) & \text{w = 1} \\ \hat{\mathbf{q}}_m log(1 - p_m) & \text{w = 0 and (1-}p_m\text{)} > \gamma \\ 0 & \text{others,} \end{cases} \tag{15}$$

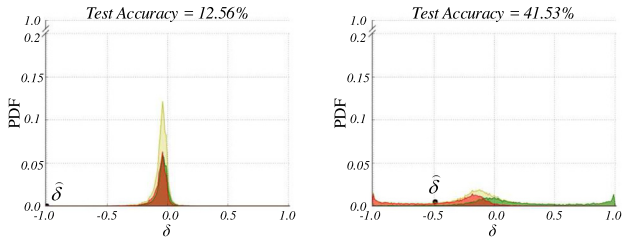
where ω is the computed weight condition of the sample. γ is another parameter (discussed in Section 5.4).

5. Experiments

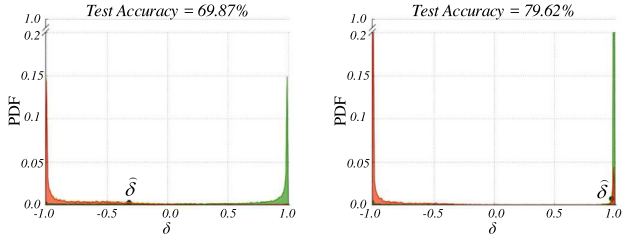
We verify the effectiveness of P-DIFF/P-DIFF+ on 4 benchmark datasets: MNIST, Cifar10, Cifar100, and Mini-ImageNet (Vinyals, Blundell, Lillicrap, Wierstra, et al., 2016), which are popularly used for evaluation of noisy labels in previous works. Furthermore, we also perform experiments on a large real-world noisy benchmark Cloth1M (Xiao et al., 2015a). For the fair comparison, we use 9-layer (Han, Yao, Yu, et al., 2018) and ResNet-101 CNNs in experiments. All models are trained by using the SGD optimizer (momentum=0.9) with an initial learning rate 0.001 on a TitanX GPU. The batch size is set to 128. We fix $T_{max} = 200$ to train all CNN classifiers, and fix $T_k = 20$ in our P-DIFF implementations. Jia, Shelhamer, Donahue, Karayev, Long, Girshick, Guadarrama, and Darrell (2014) are employed to implement P-DIFF/P-DIFF+. Following other approaches, we corrupt datasets with two types of noise transition matrix: **Symmetry flipping** and **Pair flipping**.

5.1. Probability difference distribution in training

We firstly perform an experiment to show the probability difference distribution throughout the training process. In the experiment, Cifar-10 is corrupted by using **Symmetry flipping** with 50% noisy rate. To better illustrate the effectiveness of P-DIFF, as shown in Fig. 3, we present three types of distributions: $DIST_{all}$, $DIST_{clean}$ and $DIST_{noise}$. These distributions are built by using δ values of all samples, clean samples, and noisy samples

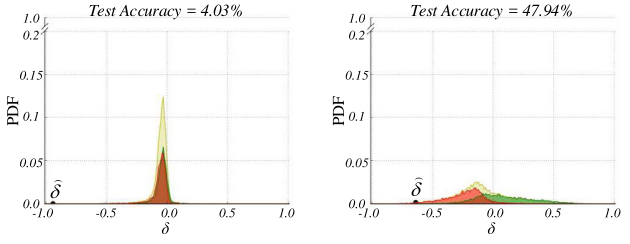


(a) The 1-st Epoch (b) The 2-nd Epoch

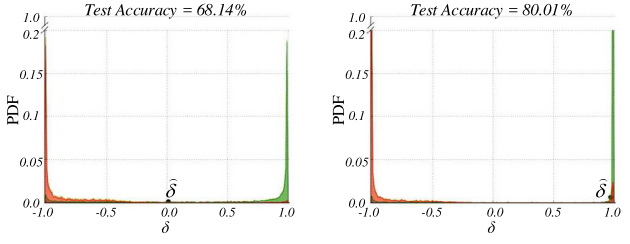


(c) The 13-th Epoch (d) The 200-th Epoch

Fig. 3. $DIST_{all}$ (Yellow), $DIST_{clean}$ (Green) and $DIST_{noise}$ (Red) at different training epochs. The DNNs are trained with given noise rates. The corresponding thresholds $\hat{\delta}$ and the performance results can also be seen in the figure. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)



(a) The 1-st Epoch. $\zeta = 0.05$ (b) The 8-th Epoch. $\zeta = 0.21$



(c) The 21-th Epoch. $\zeta = 0.84$ (d) The 200-th Epoch. $\zeta = 0.93$

Fig. 4. $DIST_{all}$ (Yellow), $DIST_{clean}$ (Green) and $DIST_{noise}$ (Red) at different training epochs. The DNNs are trained **without** given noise rates. The corresponding thresholds $\hat{\delta}$, ζ, and the performance results are presented. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

respectively. Therefore, we can conclude that $DIST_{all} = DIST_{clean} + DIST_{noise}$.

To evaluate P-DIFF without given noise rates, we perform another experiment on the same noisy dataset, but train the DNN by using the method presented in Section 3.4. $DIST_{all}$, $DIST_{clean}$ and $DIST_{noise}$ are also presented in Fig. 4. Fig. 4 shows $\hat{\delta}$, ζ, and the performance results too. We can see that most of clean and noisy samples are separated clearly by using P-DIFF.

Table 1
Average test accuracy on four testing datasets over the last 10 epochs. P-DIFF$_{m1}$ employs p_y to build the distributions.

DataSet	Noise type, rate	P-DIFF$_{m1}$	P-DIFF
Cifar-10	Symmetry,20%	85.59%	**88.61%**
	Symmetry,40%	82.74%	**85.31%**
	Pair,10%	83.69%	**87.78%**
	Pair,45%	73.47%	**83.25%**

Table 2
Average test accuracy on four Cifar-10 testing datasets over the last 10 epochs with different M. $M = 0\%$ means only samples in the single current mini-batch are used.

M	Symmetry 20%	Symmetry 40%	Pair 10%	Pair 45%
0%	87.71%	81.37%	84.87%	74.23%
5%	88.35%	83.09%	86.32%	77.95%
10%	**88.79%**	**85.64%**	88.28%	81.27%
20%	88.61%	85.31%	**87.78%**	**83.25%**
50%	88.13%	84.14%	87.34%	78.04%
100%	88.38%	85.13%	87.67%	78.48%

5.2. Effect of δ

In P-DIFF, the probability difference $\delta = p_y - p_n$, instead of p_y, is employed to construct the distribution $DIST_{all}$. To demonstrate the effectiveness of the probability difference, some experiments are conducted. We firstly train the DNN models with classical softmax losses on Cifar10. At the 1st iteration of the 2nd epoch, we compute two $DIST_{all}$ distributions constructed with δ and p_y respectively. We plot two curves to present the relationship between the drop rates and the real noise rates of dropped samples with two distributions, as shown in Fig. 5. We can observe that the yellow curves are always not lower than the corresponding green curves, which means that more samples with incorrect labels would be dropped if we employ δ to construct the distribution $DIST_{all}$, especially for the hard **Pair** noise type. Therefore, using δ can select more clean samples and should achieve better performance. This phenomenon can also been verified by the following experiment.

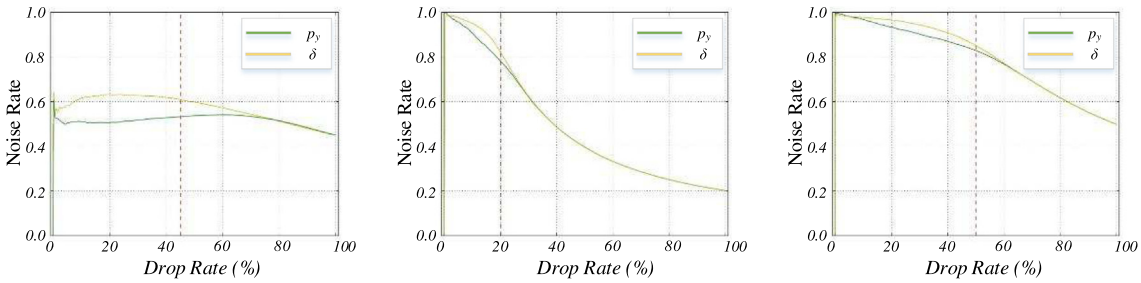
To further demonstrate the effectiveness of δ, we train the DNNs (abbreviated as P-DIFF) with some noisy datasets as shown in Table 1. We also employ P-DIFF paradigm to train the DNN (abbreviated as P-DIFF$_{m1}$) but using p_y instead of δ. Comparing the performance of P-DIFF and P-DIFF$_{m1}$ in Table 1, we can see that the probability difference δ plays the key role to achieve satisfied performance, especially on Pair Flipping datasets.

5.3. Effect of M

M indicates the rate of recent batch number used to generate the distribution $DIST_{sub}$. To demonstrate the effect of M, we perform some experiments on **Cifar-10** with different M value. Table 2 gives the comparison result. We can observe that only using samples in one mini-batch (as Han, Yao, Yu, et al., 2018; Jiang et al., 2018; Yu et al., 2019) cannot achieve satisfied performance. Meanwhile, a large M is also not preferred as discussed in Section 3.2.2, which can be observed from the table too. According to our experiments on several datasets, setting $M = 20\%$ can achieve good results in all experiments. Actually, we can observe that M is not a very sensitive parameter for achieving good performance.

5.4. Effect of γ

In Eq. (14), γ is a threshold to filter noise samples which has high confidence. According to the results shown in Table 3, γ is not a sensitive parameter to achieve good performance. So, we set $\gamma = 0.9$ in all our experiments.



(a) Pair,45% (b) Symmetry,20% (c) Symmetry,50%

Fig. 5. The plotted curves show the relationship between the drop rates and the real noise rates of dropped samples. The green and yellow curves are plotted with the real noise rates computed with two $DIST_{all}$ distributions, which are constructed by employing p_y and δ respectively. **Cifar-10** is used in these experiments. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 3
Average test accuracy on four Cifar-10 testing datasets over the last 10 epochs with different γ values.

γ	Symmetry 20%	Symmetry 40%	Pair 10%	Pair 45%
0.85	88.23%	85.83%	87.65%	83.44%
0.90	**88.93%**	86.57%	**88.32%**	**84.34%**
0.95	88.45%	**87.13%**	87.14%	83.11%

Table 4
Average test accuracy on four Cifar-10 testing datasets over the last 10 epochs with different ζ values.

ζ	Symmetry 20%	Symmetry 40%	Pair 10%	Pair 45%
0.5	87.71%	78.35%	82.39%	81.49%
0.8	**88.28%**	84.27%	85.38%	83.86%
0.85	87.37%	84.93%	85.65%	86.24%
0.90	87.61%	**85.74%**	**87.43%**	**86.73%**
0.95	86.22%	84.49%	86.94%	63.24%
1.0	86.19%	84.13%	86.32%	60.57%

5.5. Effect of ζ

We also perform several experiments to evaluate the effect of ζ in Eq. (12) when the noise rate is not given. ζ is employed in P-DIFF to reflect the degree of convergence of the model, which can be observed in Fig. 4. According to the results shown in Table 4, ζ is not a very sensitive parameter for achieving good performance too, as long as the value is not close to 1.0. Therefore, we set $\zeta = 0.9$ in all our experiments.

5.6. Experiments without a given τ

To evaluate P-DIFF without a given noise rate τ, we train the DNNs on benchmarks again, but by using the method presented in Section 3.4. Moreover, we apply P-DIFF to train DNNs with clean training datasets to demonstrate its effectiveness. The results are shown in Table 5. From the table, we can see that our estimated τ_{est} are very close to the real rates in many cases, especially when the corresponding ζ value is high. This phenomenon also proves that the ζ can be applied to evaluate the performance of the DNNs trained with P-DIFF.

To verify the effectiveness of ζ, Table 5 presents the test accuracy results (abbreviated as TA_1, and $TA_1 = TA$ if $\zeta < 0.9$) without considering ζ (using Eq. (11)). TA should be equal to TA_1 if ζ cannot exceed a threshold 0.9 throughout the training process. As shown in the table, the performance of DNNs can be further improved if the noise rates can be estimated with ζ. We also observed that P-DIFF can deal with clean datasets and achieved good results too.

By comparing with the results in Table 6, it is surprised that the DNNs trained without given noise rates even achieve better performance than the DNNs trained with correct given noise rates. More exploration should be conducted to find the reason behind this phenomenon.

5.7. Comparison with SOTA sample selection approaches

We compare the P-DIFF/P-DIFF+ with four outstanding sample selection approaches: Co-teaching (Han, Yao, Yu, et al., 2018), Co-teaching+ (Yu et al., 2019), INCV (Chen et al., 2019), and O2U-Net (Huang et al., 2019):

Co-teaching: Co-teaching simultaneously trains two networks for selecting samples. We compare Co-teaching because it is an important sample selection approach.

Co-teaching+: This work is constructed on Co-teaching, and heavily depends on samples selected by small-loss strategy. Therefore, it is suitable to compare with P-DIFF for comparison.

INCV: This recently proposed approach divides noisy datasets and utilizes cross-validation to select clean samples. Moreover, Co-teaching strategy is also applied in the method.

O2U-Net: This work also computes the probability of a sample to be noisy label by adjusting hyper-parameters of DNNs in training. Multiple training steps are employed in the approach. Its simplicity and effectiveness make it to be a competitive approach for comparison.

As the baseline, we also compare P-DIFF/P-DIFF+ with the DNNs (abbreviated as **Normal**) trained with the same noisy datasets by using the classic softmax loss. The DNNs (abbreviated as **Clean**) trained only with clean samples (For example, only 80% clean samples are used for a Symmetry-20% noisy dataset) are also presented as the *upper bound*. We corrupt datasets with 80% noise rate to demonstrate that P-DIFF can deal with extremely noisy datasets.

5.7.1. P-DIFF results

Table 6 reports the accuracy on the testing sets of four benchmarks. We can see that the DNNs trained with P-DIFF are superior to the DNNs trained with these previous state-of-the-art approaches.

We further perform experiments on a large-scale real-world dataset Cloth1M, which contains 1M/14k/10k train/val/test images with 14 fashion classes. Table 7 lists the performance results. Not only P-DIFF addresses noisy problem in the closed-set setting, but also achieve good results on real-world open-set noisy labels.

Table 5
Test accuracy TA, estimated rate τ_{est} and the corresponding ζ on three datasets. We supply TA_1, the test accuracy of the DNNs trained without considering ζ, for comparison.

DataSet	Noise type, rate	TA_1	TA	τ_{est}	ζ
MNIST	Clean, 0%	99.62%	**99.68** %	0.2%	0.99
	Symmetry, 20%	99.53%	**99.58** %	20.5%	0.99
	Symmetry, 40%	99.23%	**99.43** %	40.2%	0.99
	Pair, 10%	99.56%	**99.61** %	10.4%	0.99
	Pair, 45%	98.62%	**98.70** %	44.6%	0.99
Cifar-10	Clean, 0%	90.68%	**91.18** %	7.3%	0.96
	Symmetry, 20%	87.12%	**87.61** %	28.6%	0.96
	Symmetry, 40%	85.31%	**85.74** %	42.7%	0.93
	Pair, 10%	86.82%	**87.43** %	10.1%	0.96
	Pair, 45%	85.76%	**86.73** %	45.8%	0.95
Cifar-100	Clean, 0%	–	64.99 %	11.8%	0.83
	Symmetry, 20%	–	62.87 %	29.2%	0.85
	Symmetry, 40%	–	52.43 %	47.3%	0.71
	Pair, 10%	–	63.26%	12.8%	0.84
	Pair, 45%	–	43.24%	39.3%	0.80
Mini-ImageNet	Clean, 0%	–	55.81%	12.4%	0.81
	Symmetry, 20%	–	53.63%	30.4%	0.83
	Symmetry, 40%	–	46.34%	48.6%	0.69
	Pair, 10%	–	54.76%	13.3%	0.83
	Pair, 45%	–	37.14%	42.3%	0.79

Table 6
Average test accuracy on three testing datasets over the last 10 epochs. Accuracies of O2U-Net are cited from the original paper (Huang et al., 2019), since its authors do not provide the source codes. P-DIFF+ is P-DIFF with NNL.

DataSet	Noise type, Rate	Normal	Clean	Co-teaching	Co-teaching+	INCV	O2U-net	P-DIFF	P-DIFF+
MNIST	Symmetry, 20%	94.05%	99.68%	97.25%	99.26%	97.62%	–	99.58%	**99.65**%
	Symmetry, 40%	68.13%	99.51%	92.34%	98.55%	94.23%	–	99.38%	**99.43**%
	Symmetry, 80%	23.61%	99.04%	81.43%	93.79%	92.66%	–	97.26%	**97.61**%
	Pair, 10%	95.23%	99.84%	97.76%	99.03%	98.73%	–	99.54%	**99.66**%
	Pair, 45%	56.52%	99.59%	87.63%	83.57%	88.32%	–	99.33%	**99.52**%
Cifar-10	Symmetry, 20%	76.25%	89.10%	82.66%	82.84%	84.87%	85.24%	88.61%	**88.93**%
	Symmetry, 40%	54.37%	87.86%	77.42%	72.32%	74.65%	79.64%	85.31%	**86.57**%
	Symmetry, 80%	17.28%	80.27%	22.60%	18.45%	24.62%	34.93%	37.02%	**37.61**%
	Pair, 10%	82.32%	90.87%	85.83%	85.10%	86.27%	88.22%	87.78%	**88.32**%
	Pair, 45%	49.50%	87.41%	72.62%	50.46%	74.53%	–	83.25%	**84.34**%
Cifar-100	Symmetry, 20%	47.55%	66.37%	53.79%	52.46%	54.87%	60.53%	63.72%	**64.67**%
	Symmetry, 40%	33.32%	60.48%	46.47%	44.15%	48.21%	52.47%	54.92%	**55.68**%
	Symmetry, 80%	7.65%	35.12%	12.23%	9.65%	12.94%	**20.44**%	18.57%	19.15%
	Pair, 10%	52.94%	69.27%	57.53%	54.71%	58.41%	64.50%	67.44%	**68.57**%
	Pair, 45%	25.99%	61.29%	34.81%	27.53%	36.79%	–	45.36%	**46.74**%
Mini-ImageNet	Symmetry, 20%	37.83%	58.25%	41.47%	40.06%	43.12%	45.32%	56.71%	**57.43**%
	Symmetry, 40%	26.87%	53.88%	34.81%	34.62%	35.65%	38.39%	47.21%	**48.68**%
	Symmetry, 80%	4.11%	23.63%	6.65%	4.38%	6.71%	8.47%	11.69%	**11.93**%
	Pair, 10%	43.19%	61.64%	45.38%	43.24%	46.34%	50.32%	57.85%	**58.62**%
	Pair, 45%	19.74%	57.92%	26.76%	26.76%	28.57%	–	37.21%	**38.53**%

Table 7
Comparison on cloth1M.

Method	ResNet-101	9-Layer CNN
Coteaching	78.52%	68.74%
Coteaching+	75.78%	69.16%
INCV	80.36%	69.89%
O2U-Net	82.38%	75.61%
P-DIFF	83.67%	77.38%
P-DIFF+	**84.25**%	**78.74**%

Table 8
Training time of different approaches. The time of O2U-Net is not provided because of its closed-source.

Approach	In Theory	Real Cost/Epoch
Normal	$1\times$	64 s
Co-teaching	$\approx 2\times$	131 s
Co-teaching+	$\approx 2\times$	143 s
INCV	$> 3\times$	217 s
O2U-Net	$> 3\times$	–
P-DIFF	$\approx 1\times$	71 s
P-DIFF+	$\approx 1\times$	73 s

5.7.2. Comparison on computational efficiency

Compared with these approaches, P-DIFF/P-DIFF+ also has advantages in resource consumption and computational efficiency, since other approaches require extra DNN models or complex computation to achieve good performance. Table 8 shows the training time of these approaches for comparison. All data are measured with the 9-Layer CNNs trained on Cifar-10 with 40% symmetry noise rate. Furthermore, P-DIFF/P-DIFF+ only requires an extra small memory to store the distribution, so it costs fewer memory than other noise-free approaches too.

5.7.3. Improvement of P-DIFF with NNL

Tables 6, 7 demonstrate the P-DIFF+ accuracy on the testing sets of five benchmarks. P-DIFF+ is P-DIFF with Noisy Negative Learning, which achieve the best overall accuracy beyond P-DIFF in all cases. In the easiest datasets **MNIST** and Symmetry-40% of **CIFAR-10**, P-DIFF+ accuracy almost touch the **Clean** baseline. In Section 5.7.2 shows that the resource consumption and computational efficiency of P-DIFF+ as similar as P-DIFF, but it achieves better accuracy performance on the benchmark than P-DIFF.

Table 9
Compared with noise-tolerant approaches. We also provide the results obtained by our algorithm with label correction(†) and iterative training(‡).

model	Training Set	Accuracy
Cross Entropy (Tanaka et al., 2018)	1M Noisy	69.15%
JointOptim (Tanaka et al., 2018)	1M Noisy	72.23%
Self-Learning(Init) (Han et al., 2019)	1M Noisy	72.09%
Self-Learning(Final) (Han et al., 2019)	1M Noisy	74.43%
P-DIFF+	1M Noisy	72.6%
P-DIFF+†	1M Noisy	73.12%
P-DIFF+‡	1M Noisy	**74.67**%

5.8. Discussion with the noise-tolerant training paradigm

In this Cloth1M experiment, we compare the P-DIFF+ with two outstanding noise-tolerant approaches. In detail, we follow previous works (Han et al., 2019; Tanaka et al., 2018) and also use the ResNet-50 pre-trained on ImageNet. Compared with these SOTA approaches, P-DIFF+ shows its great potential to be a very competitive method. Furthermore, we also adopt label correction (Han et al., 2019; Tanaka et al., 2018) and iterative training (Han et al., 2019) in our algorithm. Table 9 lists the performance results.

6. Conclusion

Based on *probability difference*, *global distribution* schemes and *Noisy-negative Learning* loss function, we propose a **very simple but effective** training paradigm P-DIFF+ to train DNNs classifiers with noisy data. According to our experiments on both synthetic and real-world datasets, we can conclude that P-DIFF+ can achieve satisfied performance on datasets with different noise type and noise rate. P-DIFF+ has some parameters, such as M and ζ, but we can conclude that the performance of our paradigm is not sensitive to them according to our experiments. Since P-DIFF+ only depends on a Softmax layer, it can be easily employed for training DNN classifiers. We also empirically show that P-DIFF+ outperforms other state-of-the-arts sample selection approaches both on classification performance and computational efficiency.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

Arpit, D., Jastrzebski, S., Ballas, N., Krueger, D., Bengio, E., Kanwal, M. S., et al. (2017). A closer look at memorization in deep networks. In *ICML*.

Bengio, Y., Louradour, J., Collobert, R., & Weston, J. (2009). Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning* (pp. 41–48). ACM.

Chen, P., Liao, B., Chen, G., & Zhang, S. (2019). Understanding and utilizing deep neural networks trained with noisy labels. In *ICML*.

Divvala, S. K., Farhadi, A., & Guestrin, C. (2014). Learning everything about anything: Webly-supervised visual concept learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 3270–3277).

Fergus, R., Fei-Fei, L., Perona, P., & Zisserman, A. (2010). Learning object categories from internet image searches. *Proceedings of the IEEE*, *98*(8), 1453–1466.

Frénay, B., & Verleysen, M. (2014). Classification in the presence of label noise: a survey. *IEEE Transactions on Neural Networks and Learning Systems*, *25*(5), 845–869.

Friedman, J., Hastie, T., & Tibshirani, R. (2001). *The elements of statistical learning (vol. 1)*. (10), Springer series in statistics New York, NY, USA:.

Ghosh, A., Kumar, H., & Sastry, P. (2017). Robust loss functions under label noise for deep neural networks. In *AAAI* (pp. 1919–1925).

Goldberger, J., & Ben-Reuven, E. (2017). Training deep neural-networks using a noise adaptation layer. In *ICLR*.

Han, J., Luo, P., & Wang, X. (2019). Deep self-learning from noisy labels. In *Proceedings of the IEEE international conference on computer vision* (pp. 5138–5147).

Han, B., Yao, J., Niu, G., Zhou, M., Tsang, I., Zhang, Y., et al. (2018). Masking: A new perspective of noisy supervision. In *NIPS*.

Han, B., Yao, Q., Yu, X., Niu, G., Xu, M., Hu, W., et al. (2018). Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *NIPS*.

Hendrycks, D., Mazeika, M., Wilson, D., & Gimpel, K. (2018). Using trusted data to train deep networks on labels corrupted by severe noise. In *NIPS*.

Hu, W., Zhao, Q., Huang, Y., & Zhang, F. (2020). P-DIFF: learning classifier with noisy labels based on probability difference distributions. In *25th international conference on pattern recognition* (pp. 1882–1889). IEEE.

Huang, J., Qu, L., Jia, R., & Zhao, B. (2019). O2U-net: A simple noisy label detection approach for deep neural networks. In *Proceedings of the IEEE international conference on computer vision* (pp. 3326–3334).

Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., et al. (2014). Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the 22nd ACM international conference on multimedia* (pp. 675–678). New York, NY, USA: ACM.

Jiang, L., Zhou, Z., Leung, T., Li, L.-J., & Fei-Fei, L. (2018). MentorNet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *International conference on machine learning* (pp. 2309–2318).

Kim, Y., Yim, J., Yun, J., & Kim, J. (2019). Nlnl: Negative learning for noisy labels. In *Proceedings of the IEEE international conference on computer vision* (pp. 101–110).

Krause, J., Sapp, B., Howard, A., Zhou, H., Toshev, A., Duerig, T., et al. (2016). The unreasonable effectiveness of noisy data for fine-grained recognition. In *European conference on computer vision* (pp. 301–320). Springer.

Kumar, M. P., Packer, B., & Koller, D. (2010). Self-paced learning for latent variable models. In *Advances in neural information processing systems* (pp. 1189–1197).

Li, Y., Yang, J., Song, Y., Cao, L., Luo, J., & Li, L.-J. (2017). Learning from noisy labels with distillation. In *ICCV* (pp. 1928–1936).

Lin, T.-Y., Goyal, P., Girshick, R., He, K., & Dollár, P. (2018). Focal loss for dense object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Liu, T., & Tao, D. (2016). Classification with noisy labels by importance reweighting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *38*(3), 447–461.

Ma, X., Wang, Y., Houle, M. E., Zhou, S., Erfani, S. M., Xia, S.-T., et al. (2018). Dimensionality-driven learning with noisy labels. In *ICML*.

Malach, E., & Shalev-Shwartz, S. (2017). Decoupling "when to update" from "how to update". In *Advances in neural information processing systems* (pp. 960–970).

Menon, A., Van Rooyen, B., Ong, C. S., & Williamson, B. (2015). Learning from corrupted binary labels via class-probability estimation. In *International conference on machine learning* (pp. 125–134).

Natarajan, N., Dhillon, I. S., Ravikumar, P. K., & Tewari, A. (2013). Learning with noisy labels. In *Advances in neural information processing systems* (pp. 1196–1204).

Nettleton, D. F., Orriols-Puig, A., & Fornells, A. (2010). A study of the effect of different types of noise on the precision of supervised learning techniques. *Artificial Intelligence Review*, *33*(4), 275–306.

Niu, L., Li, W., & Xu, D. (2015). Visual recognition by learning from web data: A weakly supervised domain generalization approach. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2774–2783).

Northcutt, C. G., Wu, T., & Chuang, I. L. (2017). Learning with confident examples: Rank pruning for robust classification with noisy labels. In *Proceedings of the thirty-third conference on uncertainty in artificial intelligence*. AUAI Press.

Patrini, G., Rozza, A., Menon, A. K., Nock, R., & Qu, L. (2017). Making deep neural networks robust to label noise: A loss correction approach. In *Proc. IEEE conf. comput. vis. pattern recognit.* (pp. 2233–2241).

Ren, M., Zeng, W., Yang, B., & Urtasun, R. (2018). Learning to reweight examples for robust deep learning. In *International conference on machine learning*.

Sukhbaatar, S., Bruna, J., Paluri, M., Bourdev, L., & Fergus, R. (2015). Training convolutional networks with noisy labels. In *ICLR*.

Tanaka, D., Ikami, D., Yamasaki, T., & Aizawa, K. (2018). Joint optimization framework for learning with noisy labels. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 5552–5560).

Vahdat, A. (2017). Toward robustness against label noise in training deep discriminative neural networks. In *Advances in neural information processing systems* (pp. 5596–5605).

Veit, A., Alldrin, N., Chechik, G., Krasin, I., Gupta, A., & Belongie, S. J. (2017). Learning from noisy large-scale datasets with minimal supervision. In *CVPR* (pp. 6575–6583).

Vinyals, O., Blundell, C., Lillicrap, T., Wierstra, D., et al. (2016). Matching networks for one shot learning. In *Advances in neural information processing systems* (pp. 3630–3638).

Wang, Y., Liu, W., Ma, X., Bailey, J., Zha, H., Song, L., et al. (2018). Iterative learning with open-set noisy labels. In *CVPR*.

Wei, H., Feng, L., Chen, X., & An, B. (2020). Combating noisy labels by agreement: A joint training method with co-regularization. *CVPR*.

Xiao, T., Xia, T., Yang, Y., Huang, C., & Wang, X. (2015a). Learning from massive noisy labeled data for image classification. In *2015 IEEE conference on computer vision and pattern recognition*.

Xiao, T., Xia, T., Yang, Y., Huang, C., & Wang, X. (2015b). Learning from massive noisy labeled data for image classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 2691–2699).

Yu, X., Han, B., Yao, J., Niu, G., Tsang, I. W., & Sugiyama, M. (2019). How does disagreement help generalization against label corruption? In *ICML*.

Zhang, C., Bengio, S., Hardt, M., Recht, B., & Vinyals, O. (2017). Understanding deep learning requires rethinking generalization. In *ICLR*.