Complex & Intelligent Systems
https://doi.org/10.1007/s40747-024-01360-z

ORIGINAL ARTICLE

Separating hard clean samples from noisy samples with samples' learning risk for DNN when learning with noisy labels

Lihui Deng[1] · Bo Yang[1] · Zhongfeng Kang[2] · Jiajin Wu[1] · Shaosong Li[1] · Yanping Xiang[1]

Received: 29 May 2023 / Accepted: 21 January 2024

Abstract

Learning with **N**oisy **L**abels (LNL) methods aim to improve the accuracy of **D**eep **N**eural **N**etworks (DNNs) when the training set contains samples with noisy or incorrect labels, and have become popular in recent years. Existing popular LNL methods frequently regard samples with high learning difficulty (high-loss and low prediction probability) as noisy samples; however, irregular feature patterns from hard clean samples can also cause high learning difficulty, which can lead to the misclassification of hard clean samples as noisy samples. To address this insufficiency, we propose the **S**amples' **L**earning **R**isk-based **L**earning with **N**oisy **L**abels (SLRLNL) method. Specifically, we propose to separate noisy samples from hard clean samples using samples' learning risk, which represents samples' influence on DNN's accuracy . We show that samples' learning risk is comprehensively determined by samples' learning difficulty as well as samples' feature similarity to other samples, and thus, compared to existing LNL methods that solely rely on the learning difficulty, our method can better separate hard clean samples from noisy samples, since the former frequently possess irregular feature patterns. Moreover, to extract more useful information from samples with irregular feature patterns (i.e., hard samples), we further propose the **R**elabeling-based **L**abel **A**ugmentation (RLA) process to prevent the memorization of hard noisy samples and better learn the hard clean samples, thus enhancing the learning for hard samples. Empirical studies show that samples' learning risk can identify noisy samples more accurately, and the RLA process can enhance the learning for hard samples. To evaluate the effectiveness of our method, we compare it with popular existing LNL methods on CIFAR-10, CIFAR-100, Animal-10N, Clothing1M, and Docred. The experimental results indicate that our method outperforms other existing methods. The source code for SLRLNL can be found at https://github.com/yangbo1973/SLRLNL.

Keywords Learning with noisy labels · Deep neural networks · Generalization error · Learning risk

Introduction

Deep **N**eural **N**etworks (DNNs) have achieved remarkable success across a wide range of fields [1–7], and the research on **L**earning with **N**oisy **L**abels (LNL) methods aims to improve the accuracy of DNNs when the training dataset contains noisy labels (i.e., incorrect labels). This field of research has recently gained significant attention [8–16] for two main reasons. First, DNNs are susceptible to label noise, which can negatively affect DNNs' performance [9, 11, 17–19]. Second, real-world datasets often contain noisy labels [8, 10, 14, 18–20]. Without effective interventions, DNNs' performance on real-world datasets will be severely degraded by noisy labels [9, 11, 13, 17–19]. Therefore, research for

✉ Bo Yang
yangbo@uestc.edu.cn

Lihui Deng
denglh2019@126.com

Zhongfeng Kang
kangzhf@gmail.com

Jiajin Wu
wujiajin1997@gmail.com

Shaosong Li
shaosongli97@gmail.com

Yanping Xiang
2650931580@qq.com

[1] School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, Sichuan, China

[2] Department of Computer Science, University of Copenhagen, Universitetsparken 1, Copenhagen 2100, Denmark

Published online: 01 March 2024

LNL methods is crucial for improving DNN's performance in real-world datasets.

Popular LNL methods for DNNs can be coarsely classified into two types: loss reweighting and label correction. Loss reweighting methods frequently treat high-loss samples as noisy samples and restrain their gradient [9, 11, 14, 19, 21–23]. For instance, methods that modify the loss function to demote the gradient of high-loss samples [9, 11, 21, 24], and methods that utilize the samples' loss during training to filter out potential noisy samples [12, 14, 22, 23, 25, 26]. On the other hand, label correction methods focus on correcting samples with low prediction probability on their observed labels [10, 16, 18, 20, 27–33]. The main approaches for label correction include training with only corrected labels [10, 18, 30, 33], and training with a combination of noisy and corrected labels [20, 31, 32]. In general, for both loss reweighting methods and label correction methods, samples with high learning difficulty (i.e., DNN has low prediction probability on samples' observed labels) are often considered noisy [10, 14, 18, 19, 22]. Since noisy samples are usually fitted through brute-force memorization [34, 35], they tend to have higher learning difficulty compared to simple clean samples. Therefore, existing LNL methods are effective in separating noisy samples from simple clean samples.

Existing LNL methods frequently regard samples with high learning difficulty as noisy samples. Nevertheless, irregular feature patterns from hard clean samples can also cause high learning difficulty for DNNs, thus they can be miscorrected or filtered by existing LNL methods. Although hard clean samples are only minority in the dataset, they play a vital role in improving DNNs' generalization [12, 35–39]. Thus, separating noisy samples from hard clean samples through a more effective criterion can further improve DNN's performance in noisy labeled dataset. Although there have been previous works on utilizing samples' learning difficulty (logits output or loss) during different training epochs to distinguish between noisy samples and hard clean samples [12, 14, 16], they did not utilize the primary difference between hard clean samples and noisy samples. Hard clean samples possess correct labels, implying that their high learning difficulty primarily stems from their irregular feature patterns. This results in the learned features from other clean samples being inapplicable to these hard clean samples. Consequently, the learning difficulty of these samples is higher than that of other clean samples. On the other hand, the high learning difficulty of noisy samples is mainly caused by incorrect labels, and many of them possess feature patterns similar to those of simple clean samples. Thus, ignoring this difference may cause the existing LNL methods to misclassify hard clean samples as noisy samples.

In this paper, we propose the **S**amples' **L**earning **R**isk-based **L**earning with **N**oisy **L**abels (SLRLNL) method to better separate noisy samples from hard clean samples, thus improving DNN's learning for hard clean samples while mitigating label noise. To be specific, samples' learning risks are DNN's accuracy variation on training dataset after learning the sample, as will be demonstrated in this paper, samples' learning risk is comprehensively determined by samples' learning difficulty as well as samples' feature similarity to other samples, and only samples with high learning difficulty, as well as similar feature patterns to other samples, will be detected as noisy samples. Since hard clean samples often possess feature patterns that are dissimilar to other samples, SLRLNL can separate noisy samples from hard clean samples more effectively compared with existing LNL methods that only rely on samples' learning difficulty.

We divide our proposed SLRLNL method into two processes. The first process is the label correction process, in which we propose to identify noisy samples through samples' learning risk and then correct them to obtain clean samples to improve DNN's performance. Furthermore, to extract useful information from the samples with irregular feature patterns (i.e., hard samples), we propose a **R**elabeling-based **L**abel **A**ugmentation (RLA) process as the second process of SLRLNL. This process can prevent the DNN from memorizing hard noisy samples and enhance the learning for hard clean samples, thus extracting useful information from the hard samples. Specifically, in each epoch, the RLA process selects different samples that are likely to be hard samples and temporarily relabels them to another probable class. This process mainly relabels the hard samples, thus can prevent the DNN from memorizing the hard noisy samples. As the relabeled class may contain valuable semantic information, the temporary relabeling of the selected hard samples also encourages the DNN to learn more generalized knowledge from them, thereby improving DNN's generalization performance.

The effectiveness of the learning risk-based label correction process in identifying noisy samples, and the effectiveness of the RLA process in enhancing the learning for hard samples are evaluated through empirical studies. And we conduct experiments on five frequently used real-world datasets to evaluate our method, including four image classification datasets (CIFAR-10 and CIFAR-100 [40]; Animal-10N [27] and Clothing1M [41]) and one natural language processing dataset (Docred [2]). The experimental results from the aforementioned datasets demonstrate that our proposed method achieves better performance compared to other existing LNL methods. The source code for SLRLNL can be found at https://github.com/yangbo1973/SLRLNL.

In summary, the contributions of this paper are as follows.

- We propose the SLRLNL method to better separate noisy samples from hard clean samples. To detect noisy samples, the SLRLNL method utilizes samples' learn-

ing risk as selection criterion. Since samples' learning risk is comprehensively determined by samples' learning difficulty and samples' feature similarity to other samples, the SLRLNL method can correct noisy labels more effectively without hindering the learning of hard clean samples. Compared to existing LNL methods, our proposed SLRLNL can further enhance DNN's performance in practice.
- In addition to identifying and correcting noisy labels using the samples' learning risk, we propose the RLA process to extract more meaningful information from the hard samples.
- We conduct experiments on CIFAR-10, CIFAR-100, Animal-10N, Clothing1M, and Docred datasets. Our proposed method was evaluated against existing popular LNL methods, and the experimental results demonstrate that our method achieved better performance compared to other LNL methods.

The structure of this paper is as follows: The section Related work provides a review of related works in the context of our method. The section Preliminaries presents the preliminaries necessary for our proposed method. In the section The proposed method, we outline our proposed method. The section Experimental setup presents the experimental results obtained using our method. In the section Experimental results, we conclude this paper.

Related work

Loss reweighting methods

Loss reweighting methods are effective in demoting the influence from noisy samples. Popular loss reweighting methods first detect noisy samples through samples' learning difficulty (e.g., loss value or prediction probability on observed label) and then reduce the weight from detected samples or filter them. For example, methods that utilize DNN's prediction probability to detect and filter the noisy samples [12, 14, 19, 22, 25, 26, 42–45]; methods that modified the loss function to reduce the weight from high-loss samples [9, 11, 24, 46]. These methods yield effective results in mitigating the adverse impact from noisy labels; however, since hard clean samples frequently possess irregular feature patterns, the existing loss reweighting methods have an undesirable tendency to ignore the useful hard clean samples and train DNN with only simple samples. Consequently, these methods can probably bias DNN's training process [32] and damage DNN's performance.

Label correction methods

During the training of a DNN, the gradient from clean samples can influence the DNN's prediction probability for noisy samples [19, 35], and therefore, DNN's prediction probability can be utilized to detect and correct noisy labels [10, 15–18, 20, 27–33]. In general, these methods regard samples whose labels are highly inconsistent with DNN's prediction probability as noisy ones, then correct these samples with DNN's prediction output. Nevertheless, since hard clean samples possess feature patterns that are dissimilar to other simple clean samples, the existing label correction methods can easily misinterpret hard clean samples as noisy samples and mis-correct them. Since hard clean samples play an important role in DNN's generalization performance [36–39], mis-correcting them can lead to DNN's performance degradation.

Identify hard samples

Hard samples are essential for DNN's generalization [35, 37, 47], and thus, research in identifying hard clean samples are also important. For example, Lin et al. [48] regard samples with rare labels as hard samples, and Huang et al. [49] identify hard samples through DNN's prediction probability. And Koh et al. [50] evaluate samples' informativeness through DNN's parameters variation after sample removal, and Harutyunyan et al. [37] search for hard samples through the mutual information between the sample and DNN's parameters.

Data augmentation

Data augmentation is an effective measure for improving DNNs' generalization. In general, data augmentation methods improve DNNs' generalization through adopting transformations to the samples' input. For example, image rotation, flipping, cropping, and random scaling in image classification tasks [51], synonym replacement, random insertion, swapping, and deletion in natural language processing tasks [52]. While the methods listed above are task-specific, label augmentation, which trains DNNs with constructed artificial labels, can be used in various tasks to encourage DNNs to learn more generalized knowledge from the training samples [53, 54].

Preliminaries

In this section, we provide notations and definitions related to our proposed method, and the basic notations are listed in Table 1. Generally, the aim of this paper is to improve the DNN's accuracy when the training dataset contains noisy samples. We focus on the multi-classification task and denote

Table 1 Basic notations

Notation	Meaning
D	The clean training dataset, where $D = \{s_1, s_2, \ldots, s_n\}$
$\tilde{D}$	Observed training dataset, where $\tilde{D} = \{\tilde{s}_1, \tilde{s}_2, \ldots, \tilde{s}_n\}$
s_i	ith sample in the clean training dataset D
$\tilde{s}_i$	ith sample in the observed training dataset $\tilde{D}$
$\mathbf{x}_i$	Input for ith sample in D, where $\mathbf{x} \in \mathcal{X}$
$\mathcal{X}$	The space for input data.
y_i	Ground truth label for s_i, where $y \in \mathcal{Y}$
$\mathcal{Y}$	The set of categories, where $\mathcal{Y} = \{1, \ldots, K\}$
$\mathbf{y}_i$	One-hot form of label y_i
$\tilde{y}_i$	Observed label for $\tilde{s}_i$
$\tilde{\mathbf{y}}_i$	One-hot form for the observed label $\tilde{y}_i$
$\check{y}_i$	Label for ith sample after label correction
$\check{D}$	Training dataset after label correction
n	Size of training dataset D
K	Number of categories
$\boldsymbol{\theta}$	DNN's parameters, where $\boldsymbol{\theta} \in \Theta$
Θ	Space for DNN's parameters.
$\phi(\cdot; \boldsymbol{\theta})$	Function that generates DNN's penultimate layer output
$g(\cdot; \boldsymbol{\theta})$	Function that generates DNN's logits output
$f(\cdot; \boldsymbol{\theta})$	Function that generates DNN's prediction probability
$\ell(y, f(\mathbf{x}; \boldsymbol{\theta}))$	Loss function
$E(y, f(\mathbf{x}; \boldsymbol{\theta}))$	Evaluation metric function
N_l	Dim of the penultimate layer
α	Learning rate.
$\mathbf{z}_i$	Abbreviation for $\phi(\mathbf{x}_i; \boldsymbol{\theta}) \in \mathbb{R}^{1\times N_l}$
$\mathbf{u}_i$	Abbreviation for $g(\mathbf{x}_i; \boldsymbol{\theta}) \in \mathbb{R}^{1\times K}$
$\mathbf{p}_i$	Abbreviation for $f(\mathbf{x}_i; \boldsymbol{\theta}) \in \mathbb{R}^{1\times K}$
$\mathbf{Z}_D$	Abbreviation for $[\mathbf{z}_1; \mathbf{z}_2; ...\mathbf{z}_n]$, where $\mathbf{Z}_D \in \mathbb{R}^{n\times N_l}$
$\mathbf{U}_D$	Abbreviation for $[\mathbf{u}_1; \mathbf{u}_2; ...\mathbf{u}_n]$, where $\mathbf{U}_D \in \mathbb{R}^{n\times K}$
$\mathbf{Y}_D$	Abbreviation for $[\mathbf{y}_1; \mathbf{y}_2; ...\mathbf{y}_n]$, where $\mathbf{Y}_D \in \mathbb{R}^{n\times K}$
$\tilde{\mathbf{Y}}_{\tilde{D}}$	Abbreviation for $[\tilde{\mathbf{y}}_1; \tilde{\mathbf{y}}_2; ...\tilde{\mathbf{y}}_n]$, where $\tilde{\mathbf{Y}}_{\tilde{D}} \in \mathbb{R}^{n\times K}$
$\mathbf{P}_D$	Abbreviation for $[\mathbf{p}_1; \mathbf{p}_2; ...\mathbf{p}_n]$, where $\mathbf{P}_D \in \mathbb{R}^{n\times K}$

$D = \{s_1, s_2, \ldots, s_n\}$ as the clean training dataset, where $s_i = (\mathbf{x}_i, y_i) \in (\mathcal{X}, \mathcal{Y})$ is the ith sample of D. $\mathbf{x} \in \mathcal{X}$ is the input for the DNN, $y \in \mathcal{Y}$ is the ground truth label, where $\mathcal{X}$ is the space for input data, and we have $\mathcal{Y} = \{1, \ldots, K\}$, K is the number of categories for the classification task. Then, we define the observed training dataset as $\tilde{D} = \{\tilde{s}_1, \tilde{s}_2, \ldots, \tilde{s}_n\}$, where $\tilde{s}_i = (\mathbf{x}_i, \tilde{y}_i)$ is the ith sample of $\tilde{D}$. In practice, it is unknown whether the label of an observed sample $\tilde{s}_i$ is correct or not. We define the noisy samples as follows:

Definition 1 *(Noisy sample)*

For an observed sample $\tilde{s}_i = (\mathbf{x}_i, \tilde{y}_i)$, $\tilde{s}_i$ is noisy sample if $\tilde{y}_i \neq y_i$.

Define $\boldsymbol{\theta}$ as the parameters for the DNN. When given DNN's parameters $\boldsymbol{\theta}$, define $\phi(\cdot; \boldsymbol{\theta}) : \mathcal{X} \to \mathbb{R}^{N_l}$, $g(\cdot; \boldsymbol{\theta}) : \mathcal{X} \to \mathbb{R}^K$, and $f(\cdot; \boldsymbol{\theta}) : \mathcal{X} \to \mathbb{R}^K$ as the functions that map the sample's input to DNN's penultimate layer output, DNN's logits output, and DNN's prediction probability (output after the softmax layer), respectively. $g(\cdot; \boldsymbol{\theta})$ can be considered as DNN's feature extractor, and DNN's penultimate layer output for the sample can also be considered as the feature representation for that sample.

In this paper, for the sake of simplicity, when given the DNN's parameters $\boldsymbol{\theta}$, the DNN's penultimate layer output, logits output, and prediction probability for the sample with input $\mathbf{x}_i$ can be abbreviated as $\mathbf{z}_i$, $\mathbf{u}_i$, and $\mathbf{p}_i$, respectively. The matrices for the DNN's penultimate layer output, logits output, and prediction probability for the dataset D can be abbreviated as $\mathbf{Z}_D$, $\mathbf{U}_D$, and $\mathbf{P}_D$, respectively. The matrices for the one-hot form of clean labels and observed labels for

datasets D and $\tilde{D}$ can be abbreviated as $\mathbf{Y}_D$ and $\tilde{\mathbf{Y}}_{\tilde{D}}$, respectively. When replacing the subscript D of these abbreviated symbols above with another set of samples, it denotes the DNN's output matrices for that set.

Generally, existing LNL methods frequently attempt to identify noisy samples through DNN' learning difficulty or its extension (sample's loss value or prediction probability) [10, 19, 22]. Then, the learning difficulty is defined as follows:

Definition 2 *(Learning difficulty)*
Define $1 - f_{\tilde{y}_i}(\mathbf{x}_i; \boldsymbol{\theta})$ be DNN's learning difficulty for observed sample $(\mathbf{x}_i, \tilde{y}_i)$.

Higher learning difficulty for sample $\tilde{s}_i$ indicates that the DNN will take a longer time to eventually fit these data. Although utilizing samples' learning difficulty can effectively identify noisy samples, they may tend to mistake hard clean samples with irregular feature patterns as noisy samples . Although these samples are the minority in the dataset, they are essential for DNN's generalization [35–39]. In this paper, we define hard samples as samples with high learning difficulty when trained with a clean dataset. They are defined as hard clean samples if their observed labels are consistent with their ground truth labels; otherwise, they are defined as hard noisy samples, as shown below:

Definition 3 *(Hard clean/noisy sample)*
For an observed sample $\tilde{s}_i = (\mathbf{x}_i, \tilde{y}_i)$, $\tilde{s}_i$ is hard clean sample if $\tilde{y}_i = y_i$ and $1 - f_{y_i}(\mathbf{x}_i; \boldsymbol{\theta}^*) > \tau$, and is hard noisy sample if $\tilde{y}_i \neq y_i$ and $1 - f_{y_i}(\mathbf{x}_i; \boldsymbol{\theta}^*) > \tau$. $\boldsymbol{\theta}^*$ is the optimal parameters for DNN trained with clean dataset D, and τ is the selection criteria for hard clean samples.

Other than noisy samples and hard clean samples, the simple clean sample is defined as follows:

Definition 4 *(Simple clean sample)*
For an observed sample $\tilde{s}_i = (\mathbf{x}_i, \tilde{y}_i)$, $\tilde{s}_i$ is simple clean sample if $\tilde{y}_i = y_i$ and $1 - f_{y_i}(\mathbf{x}_i; \boldsymbol{\theta}^*) \leq \tau$, where $\boldsymbol{\theta}^*$ is the optimal parameter for DNN trained with clean dataset.

Define $\ell(y, f(\mathbf{x}; \boldsymbol{\theta}))$ and $E(y, f(\mathbf{x}; \boldsymbol{\theta}))$ as the loss function and evaluation metric function, respectively. Both functions can be utilized to measure how close the DNN's prediction probability is to the sample's label. In this paper, we propose to better separate noisy samples from both hard clean samples and simple clean samples through the learning risk of samples. The definitions of samples' learning risk are given as follows:

Definition 5 *(Learning risk)*
Denote the learning risk from sample $\tilde{s}_i$ to be $\triangle E_{\tilde{s}_i \to D}$, which is the variation of DNN's empirical risk in clean training set D after updating the gradient from $\tilde{s}_i$:

$$\triangle E_{\tilde{s}_i \to D} = \frac{1}{|D|} \sum_{(\mathbf{x}_d, y_d) \in D} E(y_d, f(\mathbf{x}_d; \boldsymbol{\theta} + \triangle \boldsymbol{\theta}_i)) - E(y_d, f(\mathbf{x}_d; \boldsymbol{\theta})), \tag{1}$$

where $\triangle \boldsymbol{\theta}_i$ is the variation of the DNN's parameters after updating the gradient from $\tilde{s}_i$.

The proposed method

Our proposed method is presented in this section. In the section Calculation for samples' learning risk, we demonstrate the calculation method for samples' learning risk. The section Empirical study of separating noisy samples from hard clean samples provides an empirical study of our method in separating hard clean samples from noisy samples. We illustrate the methods for label correction in the section Label correction for noisy samples and present the proposed RLA in the section Relabeling-based label augmentation. Finally, in the section Implementation detail, we present the overall algorithm for SLRLNL and implementation details.

Calculation for samples' learning risk

This subsection demonstrates the calculation method for samples' learning risks. To calculate the learning risk, we use Mean Square Error (MSE) as the evaluation metric function for DNN accuracy

$$E_{MSE}(y, g(\mathbf{x}; \boldsymbol{\theta})) = ||\mathbf{y} - g(\mathbf{x}; \boldsymbol{\theta})||_2^2, \tag{2}$$

then DNN's empirical risk in D is

$$\frac{1}{n} \sum_{(\mathbf{x}_d, y_d) \in D} ||\mathbf{y}_d - g(\mathbf{x}_d; \boldsymbol{\theta})||_2^2, \tag{3}$$

then, for a DNN with parameters $\boldsymbol{\theta}$, suppose MSE is adopted as the loss function and gradient descent is adopted as the optimizer, after updating gradient from $\tilde{s}_i = (\mathbf{x}_i, \tilde{y}_i)$, the variation for the logits output of sample $s_d = (\mathbf{x}_d, y_d)$ is

$$\triangle \mathbf{u}_{\tilde{s}_i \to s_d} = -2\alpha(\mathbf{z}_i(\mathbf{z}_d + \triangle \mathbf{z}_{\tilde{s}_i \to s_d})^T + 1)(\mathbf{u}_i - \tilde{\mathbf{y}}_i), \tag{4}$$

where α is the learning rate, $\mathbf{z}_i = \phi(\mathbf{x}_i; \boldsymbol{\theta}) \in \mathbb{R}^{1 \times N_l}$ and $\mathbf{z}_d = \phi(\mathbf{x}_d; \boldsymbol{\theta}) \in \mathbb{R}^{1 \times N_l}$ are DNN's penultimate layer output for sample $\tilde{s}_i$ and sample s_d, respectively. $\mathbf{u}_i = g(\mathbf{x}_i; \boldsymbol{\theta}) \in \mathbb{R}^{1 \times K}$ and $\tilde{\mathbf{y}}_i \in \mathbb{R}^{1 \times K}$ are the logits output and one-hot form observed label for sample $\tilde{s}_i$, respectively. $\triangle \mathbf{z}_{\tilde{s}_i \to s_d}$ is the variation for DNN's penultimate layer output for sample s_d after updating gradient from sample $\tilde{s}_i$. Then, the learning risk $\triangle E_{\tilde{s}_i \to D}$ for sample $\tilde{s}_i = (\mathbf{x}_i, \tilde{y}_i)$ is

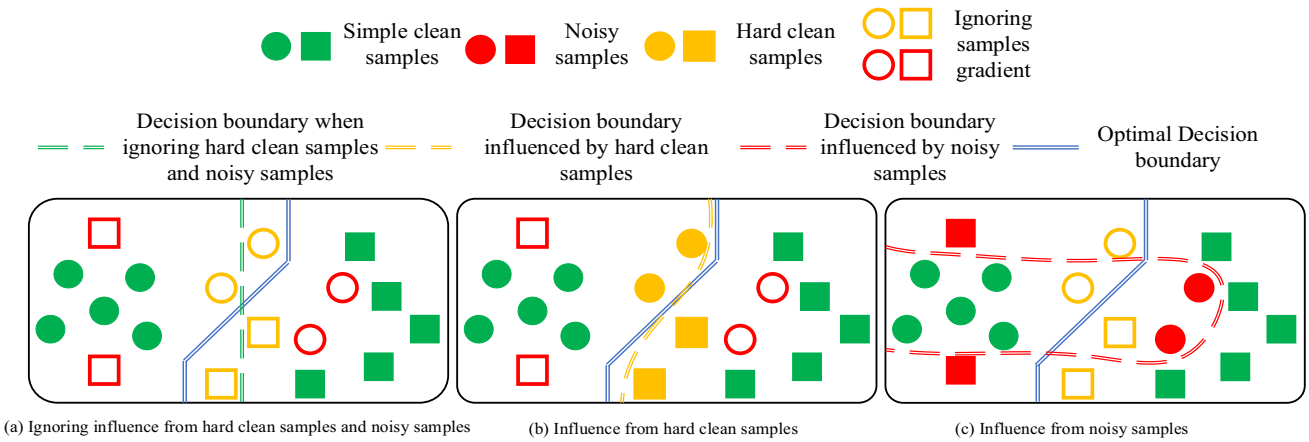


Fig. 1 Illustration for the difference between hard clean samples and noisy samples

$$
\begin{aligned}
&\triangle E_{\tilde{s}_i \to D} \\
&= \frac{1}{|D|} \sum_{(\mathbf{x}_d, y_d) \in D} ||\mathbf{y}_d - \mathbf{u}_d - \triangle \mathbf{u}_{\tilde{s}_i \to s_d}||_2^2 - ||\mathbf{y}_d - \mathbf{u}_d||_2^2 \\
&= \frac{4\alpha}{|D|} (\mathbf{z}_i (\mathbf{Z}_D)^T + 1)(\mathbf{U}_D - \mathbf{Y}_D)(\tilde{\mathbf{y}}_i - \mathbf{u}_i)^T \\
&\quad + \frac{4\alpha^2}{|D|} (1 + \mathbf{u}_i \mathbf{u}_i^T - 2\mathbf{u}_i \tilde{\mathbf{y}}_i^T) \sum_{s_d \in D} (\mathbf{z}_i (\mathbf{z}_d)^T + 1)^2. \qquad (5)
\end{aligned}
$$

The proof for Eq. (4) and Eq. (5) can be found in Appendix A. In Eq. (5), $\mathbf{u}_d \in \mathbb{R}^{1\times K}$, $\mathbf{z}_d \in \mathbb{R}^{1\times N_l}$, and $\mathbf{y}_d \in \mathbb{R}^{1\times K}$ are sample s_d's logits output, penultimate layer output, and one-hot form clean label, respectively. $\mathbf{Z}_D \in \mathbb{R}^{n\times N_l}$, $\mathbf{U}_D \in \mathbb{R}^{n\times K}$, and $\mathbf{Y}_D \in \mathbb{R}^{n\times K}$ are matrices for clean dataset D's penultimate layer output, logits output, and one-hot form clean label, respectively. Since the clean dataset D is generally unavailable in practice, in this paper, we adopt low learning difficulty samples as replacements, which are likely to be simple clean samples [35, 55]. In each epoch, we select a subset of the observed dataset as the nearly clean samples $C^{(t)}$

$$
C^{(t)} = \left\{ (\mathbf{x}_i, \tilde{y}_i) | \frac{1}{t} \sum_{m=1}^{t} \left(1 - f_{\tilde{y}_i}(\mathbf{x}_i; \boldsymbol{\theta}^{(m)}) \right) \leq \tau(t, n_C) \right\}, \qquad (6)
$$

where t is the current epoch, $\tau(t, n_C)$ is the threshold for selecting $C^{(t)}$, and is the $n_C\%$ lowest average learning difficulty from 1th to tth epoch, and n_C is the hyper-parameter of our method in selecting nearly clean samples. Then, the nearly clean samples $C^{(t)}$ are utilized to represent the clean dataset D. Note that the learning rate α is also involved in Eq. (5). In practice, the learning rate α will be set to a small value, and thus, for the sake of simplicity, we set $\alpha \to 0^+$ and ignore the latter part in Eq. (5). Then, the learning risk for sample $\tilde{s}_i$ is

$$
\begin{aligned}
\triangle E_{\tilde{s}_i \to C^{(t)}} &= \frac{4\alpha}{|C^{(t)}|} (\mathbf{z}_i (\mathbf{Z}_{C^{(t)}})^T + 1) \\
&\quad (\mathbf{U}_{C^{(t)}} - \tilde{\mathbf{Y}}_{C^{(t)}})(\tilde{\mathbf{y}}_i - \mathbf{u}_i)^T, \qquad (7)
\end{aligned}
$$

where $\mathbf{Z}_{C^{(t)}}$, $\mathbf{U}_{C^{(t)}}$, and $\tilde{\mathbf{Y}}_{C^{(t)}}$ are the matrices for the nearly clean samples $C^{(t)}$'s penultimate layer output, logits output, and one-hot form observed label, respectively. Moreover, since in practice the cross-entropy loss is frequently utilized for classification tasks, in this paper, we use the cross-entropy loss function to train the DNN. Then, to represent samples' learning risk when trained with cross-entropy, we replace the logits output terms in Eq. (7) with DNN's prediction probability, and thus, the learning risk for sample $\tilde{s}_i$ can be represented as

$$
\begin{aligned}
\triangle E_{\tilde{s}_i \to C^{(t)}} &= \frac{4\alpha}{|C^{(t)}|} (\mathbf{z}_i (\mathbf{Z}_{C^{(t)}})^T + 1)(\mathbf{P}_{C^{(t)}} - \tilde{\mathbf{Y}}_{C^{(t)}}) \\
&\quad (\tilde{\mathbf{y}}_i - \mathbf{p}_i)^T, \qquad (8)
\end{aligned}
$$

where $\mathbf{p}_i = f(\mathbf{x}_i; \boldsymbol{\theta})$ is DNN's prediction probability for sample $\tilde{s}_i$, and $\mathbf{P}_{C^{(t)}}$ is the matrix for the nearly clean samples $C^{(t)}$'s prediction probability. In this paper, we utilize Eq. (8) to calculate samples' learning risk. As shown in Eq. (8), the learning risk of the ith sample is mutually determined by the term $(\tilde{\mathbf{y}}_i - \mathbf{p}_i)$ that similar to sample's learning difficulty, and its feature similarity to other samples: $\mathbf{z}_i (\mathbf{Z}_{C^{(t)}})^T$. This equation indicates that the influence of sample $\tilde{s}_i$ on DNN's empirical risk is comprehensively determined by its feature similarity to other samples and its learning risk, and samples with higher feature similarity and higher learning difficulty will have a greater learning risk.

Hard clean samples typically exhibit irregular feature patterns. Therefore, for a hard clean sample s_i, its feature representation $\mathbf{z}_i$ will be different from any other sample s_d, resulting in a small feature similarity $\mathbf{z}_i (\mathbf{z}_d)^T$. According to Eq. (8), learning these hard clean samples does not signifi-

cantly increase the DNN's empirical risk. In contrast, many noisy samples often have both high feature similarity and high learning difficulty. This means that learning such samples can distort the DNN's decision boundary and increase its empirical risk. Therefore, by setting an appropriate threshold, the learning risk can accurately detect and correct noisy samples while avoiding mis-correcting hard clean samples. The difference between hard clean samples and noisy samples is illustrated in Fig. 1.

In the next subsection, it will be demonstrated that existing LNL methods are ineffective in distinguishing noisy samples from hard clean samples, while from the perspective of samples' learning risk, these samples can be effectively separated.

Empirical study of separating noisy samples from hard clean samples

We conduct numerical experiments on CIFAR-10 and CIFAR-100 with artificially generated label noise to compare our proposed SLRLNL with the existing LNL methods in separating noisy samples from hard clean samples. The selection criteria used by the existing LNL methods to identify noisy samples are listed below:

- Co-teaching [22]: Han et al. [22] train two DNNs, and utilize the loss value from another DNN to identify noisy samples. The selection criteria for identifying noisy samples for co-teaching are samples' loss value from the other DNN

$$-\log f_{\tilde{y}_i}(\mathbf{x}_i;\boldsymbol{\theta}'), \tag{9}$$

  where $\boldsymbol{\theta}'$ is the parameters from another DNN.
- Progressive Label Correction (PLC) [10]: Zhang et al. [10] regard samples whose labels are highly inconsistent with DNN's prediction probability as noisy samples, then progressively correct them. The selection criteria for PLC in identifying noisy samples are

$$\max_{j\neq\tilde{y}}\left(f_j(\mathbf{x}_i;\boldsymbol{\theta})-f_{\tilde{y}_i}(\mathbf{x}_i;\boldsymbol{\theta})\right). \tag{10}$$

- Area Under the Margin ranking (AUM) [19]: Pleiss et al. [19] utilize the average difference between the logits values for sample's observed class and its highest other class to identify noisy samples. Their selection criteria for identifying noisy samples are

$$\frac{1}{t}\sum_{m=1}^{t}\left(\max_{j\neq\tilde{y}}\left(g_j(\mathbf{x}_i;\boldsymbol{\theta}^{(m)})\right)-g_{\tilde{y}_i}(\mathbf{x}_i;\boldsymbol{\theta}^{(m)})\right). \tag{11}$$

Existing LNL methods consider samples with high selection criteria listed above to be noisy. In the numerical experiments, hard clean samples are selected as the samples whose learning difficulty $1-f_{y_i}(\mathbf{x}_i;\boldsymbol{\theta}^*)$ are above the highest 10% learning difficulty, and DNN's parameters $\boldsymbol{\theta}^*$ are obtained by training a DNN with clean dataset. To simulate noisy labels in real-world datasets, we generate 80% uniform flip and 40% pair flip label noise for CIFAR-10 and CIFAR-100, respectively (the details for generating label noise are illustrated in Section 5). Other samples are regarded as simple clean samples if they are neither noisy samples nor hard clean samples. The adopted DNN is ResNet-18 trained from scratch. The batch size is 64, and we train the DNN with SGD. The learning rate is 2e-2, the momentum is set to 0.9, and the weight decay rate is 5e-4. After the warm-up process (we set a 10-epoch warm-up process for CIFAR-10 and 30-epoch warm-up for CIFAR-100), the experimental results are shown in Fig. 2.

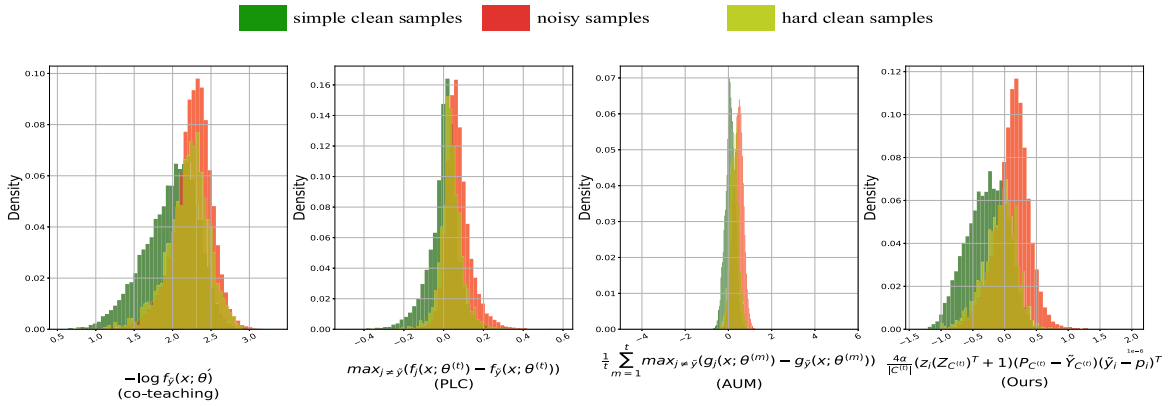
Figure 2 shows the histogram of each selection criteria (Co-teaching, PLC, AUM, and learning risk) for CIFAR-10 and CIFAR-100 with different types of label noise. The horizontal axis represents the value of selection criteria. The vertical axis in Fig. 2 represents the density of the selection criteria, which shows the proportion of data points within each range. To effectively separate noisy samples from simple clean samples and hard clean samples, the selection criteria for noisy samples need to be higher than those for clean samples.

The results presented in Fig. 2 demonstrate that the proposed learning risk criterion can more effectively distinguish noisy samples from both simple clean samples and hard clean samples in both CIFAR-10 and CIFAR-100 under different noise types. Moreover, as shown in Fig. 2b, d, when facing pair flip label noise, existing LNL methods can barely separate noisy samples from hard clean samples. This is due to the fact that pair flip label noise will flip the ground truth labels into other similar classes, thereby will not significantly increasing their learning difficulty. On the other hand, since pair flip label noise can still damage DNN's accuracy performance, the samples with pair flip label noise can still be effectively separated from hard clean samples by the learning risk criterion, as shown in Fig. 2b, d. In this case, correcting samples with high learning risk is able to mitigate the noisy labels without hindering the learning of hard clean samples and further improve DNN's performance in practice.

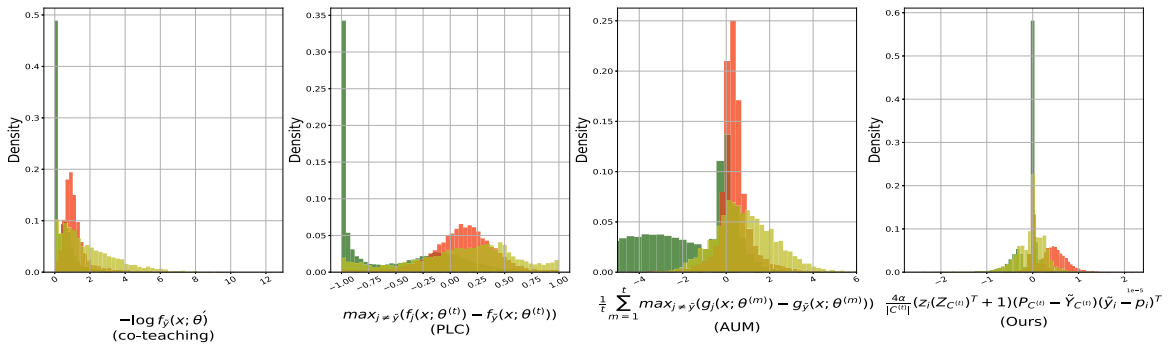
After evaluating the effectiveness of learning risk in separating noisy samples, the label correction method utilized in this paper will be demonstrated in the next subsection.

Label correction for noisy samples

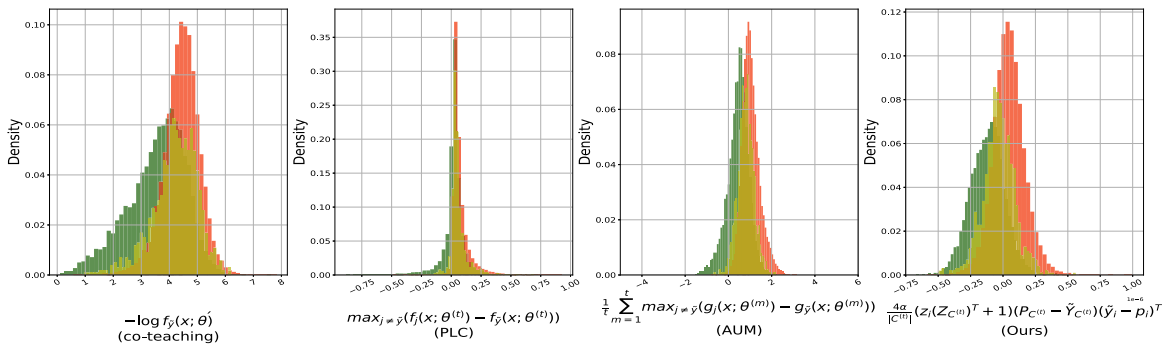
The label correction method utilized in this paper is illustrated in this subsection. As shown in Fig. 2, unlike hard clean



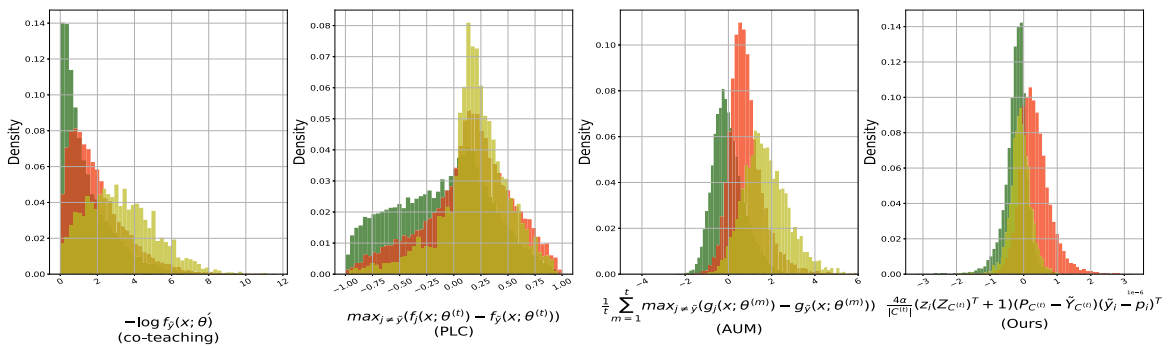
(a) Numerical experiments on CIFAR-10 with 80% uniform flip noise.



(b) Numerical experiments on CIFAR-10 with 40% pair flip noise.



(c) Numerical experiments on CIFAR-100 with 80% uniform flip noise.



(d) Numerical experiments on CIFAR-100 with 40% pair flip noise.

Fig. 2 Comparison between SLRLNL and existing LNL methods in separating noisy samples from hard clean samples

samples and simple clean samples, noisy samples tend to have higher learning risk. Thus, we conduct label correction on samples with high learning risk to mitigate the negative impact from noisy samples, as shown in Eq. (12)

$$\breve{y}_i^{(t+1)} = \begin{cases} \arg\max_{j\in\mathcal{Y}} f_j(\mathbf{x}_i;\boldsymbol{\theta}^{(t)}), & \triangle E_{\tilde{s}_i\to C^{(t)}} \geq \upsilon(t,n_V), \\ \tilde{y}_i, & \triangle E_{\tilde{s}_i\to C^{(t)}} < \upsilon(t,n_V), \end{cases} \tag{12}$$

where $\triangle E_{\tilde{s}_i\to C^{(t)}}$ is calculated through Eq. (8), and $\upsilon(t,n_V)$ is the selection threshold, which is the $n_V\%$ highest learning risk in the tth epoch. $n_V \in [0,100)$ is the correction proportion, and is a hyper-parameter of our method. In this paper, we set n_V to be increased along the training process to achieve more effective label correction, and the implementation detail can be found in hyper-parameters setting part in Section 5.

The label correction process based on samples' learning risk can detect and correct noisy samples without miscorrecting the hard clean samples, thereby improving DNN's performance in practice.

Relabeling-based label augmentation

Since samples with irregular feature patterns are important for DNN's generalization, to better utilize the information contained in these hard samples, we propose the **R**elabeling-based **L**abel **A**ugmentation (RLA) process, which focus on relabeling samples with high learning difficulty. Since samples with common features and noisy labels can be easily corrected by the learning risk-based label correction, those samples with high learning difficulty after the label correction process are typically attributed to their irregular feature patterns, making them likely to be hard samples. Thus, selecting samples with high learning difficulty can focus on relabeling the hard samples.

Specifically, the RLA process is to select different samples with high learning difficulty in each epoch then temporarily relabel them to the most probable class other than the training labels in the previous epoch, as shown in Eq. (13)

$$\begin{aligned} &\ell_{RLA}(\breve{y}_i^{(t)}, f(\mathbf{x}_i;\boldsymbol{\theta}^{(t)})) \\ &= \begin{cases} \ell\big(\arg\max_{j\neq\breve{y}_i^{(t-1)}} f_j(\mathbf{x}_i;\boldsymbol{\theta}^{(t)}), f(\mathbf{x}_i;\boldsymbol{\theta}^{(t)})\big), & i\in I_R^{(t)}, \\ \ell(\breve{y}_i^{(t)}, f(\mathbf{x}_i;\boldsymbol{\theta}^{(t)})), & else, \end{cases} \end{aligned} \tag{13}$$

where $I_R^{(t)}$ is the index set of selected samples for the RLA process in the tth epoch, and we set $I_R^{(t)}$ to select high learning difficulty samples different from the previous epoch, as shown as follows:

$$I_R^{(t)} = \left\{ i | 1 - f_{\breve{y}_i^{(t)}}(\mathbf{x}_i;\boldsymbol{\theta}^{(t)}) \geq \rho(t,n_R) \;\&\; i \notin I_R^{(t-1)} \right\}, \tag{14}$$

where $\rho(t,n_R)$ is the threshold for selecting $I_R^{(t)}$, and is $n_R\%$ highest learning difficulty in the tth epoch, and $n_R \in [0,100)$ is the relabeling proportion that determines the effectiveness of RLA, and is one of the hyper-parameters of our proposed method.

As shown in Eqs. (13) and (14), in each epoch, different samples with high learning difficulty will be selected by the RLA process. Therefore, in each epoch, the RLA process will temporarily relabel the selected samples, preventing the DNN from memorizing hard noisy samples. Moreover, as shown in Eq. (13), the selected samples will be relabeled with the class to which the DNN assigns a certain degree of prediction probability. In a multi-classification task, the relabeled class can retain a certain amount of semantic information, thus assisting the DNN in acquiring more generalized knowledge about the selected hard samples, which will be beneficial for DNN's generalization performance. Additionally, according to Eq. (14), the RLA process does not permanently modify the labels of the hard clean samples, it will not bias their gradient in an incorrect direction, and thus, overall, this process can enhance the learning of hard clean samples.

As will be demonstrated in section 6.5, for CIFAR-10 and CIFAR-100 datasets with different types of label noise, the proposed RLA process can improve the DNN's test accuracy, which indicates that the RLA process effectively mitigates the negative impact of hard noisy samples and improve DNN's generalization performance. Additionally, the experimental results in section 6.6 illustrate that the RLA process can reduce the minimal training loss for the hard clean samples, which proves the efficacy of the RLA process in enhancing the learning process for hard clean samples.

The implementation detail of our method is provided in the next subsection.

Implementation detail

Basic implementation detail

The hyper-parameters related to SLRLNL mentioned above include: nearly clean samples selection parameter n_C, max correction proportion $max\ n_V$, and relabeling proportion n_R for RLA. For the label correction process, in each dataset, we increase the correction proportion n_V by 2 during each epoch until it reaches the maximum proportion $max\ n_V$. Other than the mentioned parameters, we also set up a hyper-parameter warm-up epoch t_w to attain a DNN with basic classification ability before conducting SLRLNL. The details of our hyper-parameter settings can be found in "Hyper-parameters setting" in section "Experimental setup". The flowchart of our SLRLNL method is shown in Fig. 3.

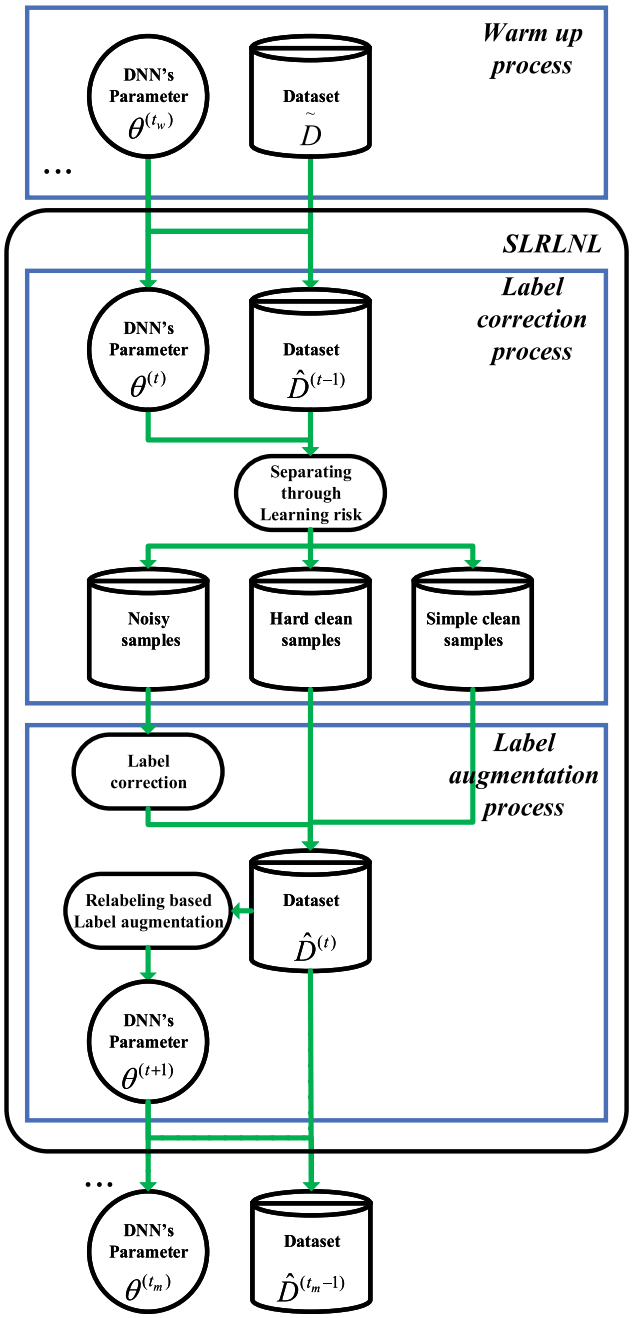


Fig. 3 The flowchart for our method, where t_w is the warm-up epoch and t_m is the total training epochs

The overall algorithm is presented in Algorithm 1. The extra computational complexity of our SLRLNL method is listed in Table 2. The extra computational time is primarily generated during the calculation of the samples' learning risk, which is $O(n^2 n_C\%(N_l + K))$, where n is the size of the training dataset, $n_C\%$ is the ratio for selecting the nearly clean samples, and N_l is dim of DNN's penultimate layer. In practice, we can adjust the proportion for selecting the nearly clean samples n_C to reduce the extra computational time from our proposed method. As will be shown in the experiment section, the extra computation time for our proposed SLRLNL is feasible.

Class imbalance issue

Class imbalance is a common issue in practice. For instance, the data sizes of different classes can significantly vary in real-world scenarios (e.g., datasets such as Clothing1M [41] and Docred [2] utilized in this paper). Consequently, this variation leads to differences in learning difficulty across classes and ultimately impacts the effectiveness of label correction methods. To improve the performance of our method in practice, in this paper, for all datasets, the selection threshold ($\tau(t, n_C)$, $\upsilon(t, n_V)$, and $\rho(t, n_R)$) and the ranking process are performed separately for each class. Let $K_j = \{i|\tilde{y}_i = j\}$ be the set of indexes of samples with observed label j. For class imbalanced datasets, the selection process in tth epoch for the nearly clean samples $C^{(t)}$ is as follows:

$$C^{(t)} = \cup_{j=1}^{K} \left\{ (\mathbf{x}_i, \tilde{y}_i) | \frac{1}{t} \sum_{m=1}^{t} \left(1 - f_{\tilde{y}}(\mathbf{x}_i; \boldsymbol{\theta}^{(m)})\right) \leq \tau_j(t, n_C) \ \& \ i \in K_j \right\}, \tag{15}$$

where $\tau_j(t, n_C)$ is the $n_C\%$ lowest average learning difficulty among samples in K_j from 1th to tth epoch, and the label correction process for the class imbalanced dataset is

$$\check{y}_i^{(t+1)} = \begin{cases} \arg\max_{j \in \mathcal{Y}} f_j(\mathbf{x}_i; \boldsymbol{\theta}^{(t)}), & \triangle E_{\tilde{s}_i \to C^{(t)}} \geq \upsilon_{\tilde{y}_i}(t, n_V), \\ \tilde{y}_i, & \triangle E_{\tilde{s}_i \to C^{(t)}} < \upsilon_{\tilde{y}_i}(t, n_V), \end{cases} \tag{16}$$

where $\upsilon_{\tilde{y}_i}$ is the $n_V\%$ highest learning risk among samples in $K_{\tilde{y}_i}$ in the tth epoch. The selection for $I_R^{(t)}$ for class imbalanced dataset is

$$I_R^{(t)} = \cup_{j=1}^{K} \{i | 1 - f_{\tilde{y}_i}(\mathbf{x}_i; \boldsymbol{\theta}^{(t)}) \geq \rho_j(t, n_R) \ \& \ i \notin I_R^{(t-1)} \ \& \ i \in K_j\}, \tag{17}$$

where $\rho_j(t, n_R)$ is the $n_R\%$ highest learning difficulty among samples in K_j in the tth epoch.

Experimental setup

Datasets

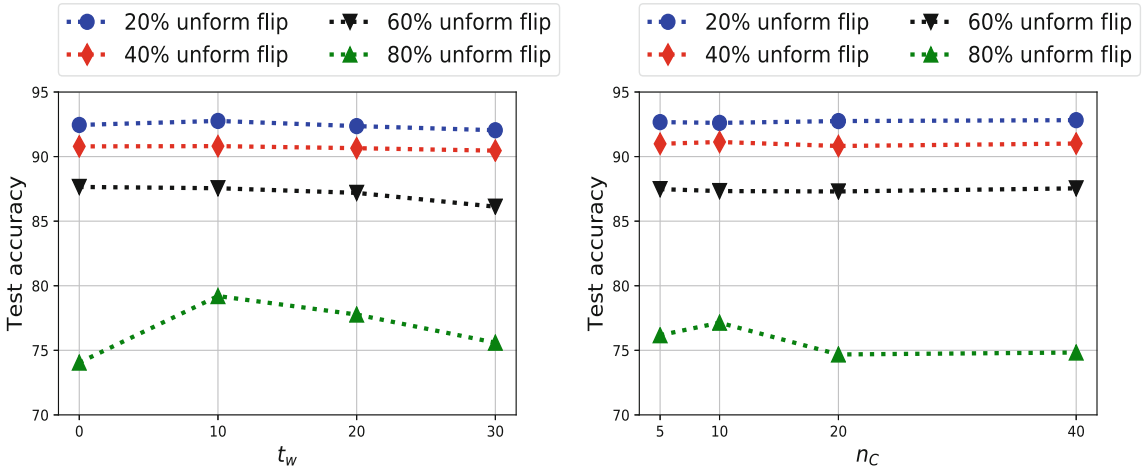
The experiments are conducted on four image classification datasets: CIFAR-10 and CIFAR-100[1] [40], Animal-10N[2]

[1] https://www.cs.toronto.edu/ kriz/cifar.html.

[2] https://dm.kaist.ac.kr/datasets/animal-10n/.

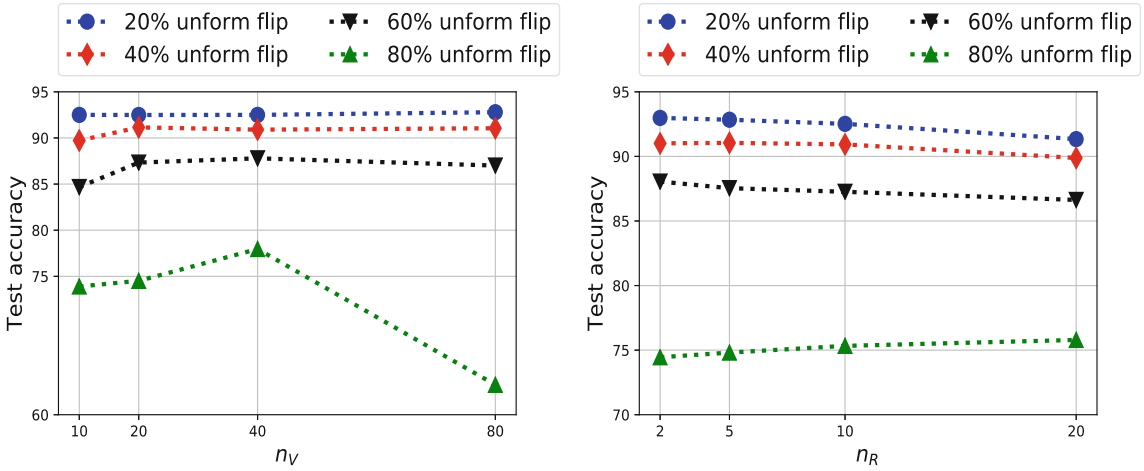
Table 2 Computational complexity of the major components and the overall complexity of our proposed method

Operation	Computation complexity during each epoch
Searching nearly clean samples	$O(n \log n)$
Calculating samples' learning risk	$O(n^2 n_C \%(N_l + K))$
Identifying noisy samples	$O(n \log n)$
Label correction	$O(n)$
Relabeling-based label augmentation	$O(nK)$
Overall	$O(n^2 n_C \%(N_l + K))$



(a) Best test accuracy with different set of t_w.

(b) Best test accuracy with different set of n_C.



(c) Best test accuracy with different set of n_V.

(d) Best test accuracy with different set of n_R.

Fig. 4 Best test accuracy of SLRLNL on CIFAR-10 with different set of hyper-parameters

[27], Clothing1M[3] [41], and one natural language processing dataset: Docred[4] [2]. In this paper, label noise in the original CIFAR-10 and CIFAR-100 datasets can generally be ignored, while Animal-10N and Clothing1M datasets both contain real-world label noise. Docred includes both clean and noisy labeled datasets. To better evaluate our proposed method, two types of artificial label noise are generated for CIFAR-10 and CIFAR-100: uniform flip noise and pair flip noise. Following existing research [15, 18], artificial noise is generated by replacing labels from randomly selected samples, with samples being selected with the probability of the

[3] https://github.com/Cysu/noisy_label.

[4] https://github.com/thunlp/DocRED.

Table 3 Average test accuracy and their standard deviation over three trials for DNN trained on CIFAR-10 with standard method, other LNL methods, and our method

	20% pair flip	30% pair flip	40% pair flip	20% uniform flip	40% uniform flip	60% uniform flip	80% uniform flip
Standard [18]	88.8 ± 0.2	88.4 ± 0.3	84.5 ± 0.3	87.5 ± 0.2	83.1 ± 0.4	76.4 ± 0.4	47.6 ± 2.0
GCE [9, 24]	83.8 ± 0.1	80.8 ± 0.2	71.5 ± 0.2	84.9 ± 0.1	82.4 ± 0.1	75.2 ± 0.1	40.8 ± 0.2
SL [24]	87.4 ± 0.2	83.2 ± 0.2	76.5 ± 0.3	87.6 ± 0.1	85.3 ± 0.1	80.1 ± 0.0	53.8 ± 0.3
MentorNet [18, 42]	90.4 ± 0.2	88.9 ± 0.1	83.3 ± 1.0	90.3 ± 0.3	83.2 ± 0.5	75.5 ± 0.7	34.1 ± 2.5
Co-teaching [18, 22]	91.8 ± 0.1	89.9 ± 0.2	80.1 ± 0.7	90.1 ± 0.4	87.3 ± 0.5	80.9 ± 0.5	25.0 ± 3.6
AUM [19]	88.5 ± 0.4	82.0 ± 0.3	53.3 ± 0.5	90.2 ± 0.1	87.5 ± 0.1	82.1 ± 0.1	54.4 ± 1.6
SELFIE [27]	89.7 ± 0.2	88.2 ± 0.3	86.9 ± 0.2	89.1 ± 0.2	86.7 ± 0.3	73.6 ± 0.2	48.6 ± 1.3
Co-teaching+ [25, 26]	89.4 ± 0.2	87.1 ± 0.5	71.3 ± 0.8	89.8 ± 0.2	86.1 ± 0.2	74.0 ± 0.2	17.9 ± 1.1
RoG [28]	89.6 ± 0.4	88.4 ± 0.5	86.2 ± 0.6	89.2 ± 0.3	83.5 ± 0.4	77.9 ± 0.6	29.1 ± 1.8
PENCIL [29]	90.2 ± 0.2	88.3 ± 0.2	84.5 ± 0.5	88.2 ± 0.2	86.6 ± 0.3	74.3 ± 0.6	45.3 ± 1.4
TopoFilter [26]	90.5 ± 0.2	89.7 ± 0.3	87.9 ± 0.2	90.2 ± 0.2	87.2 ± 0.4	80.5 ± 0.4	45.7 ± 1.0
Me-Momentum [12]	91.3 ± 0.2	88.0 ± 0.8	84.8 ± 2.2	91.4 ± 0.3	88.4 ± 0.3	79.8 ± 0.7	46.6 ± 1.5
LRT [18]	*92.2 ± 0.1*	*91.3 ± 0.3*	89.2 ± 0.4	91.0 ± 0.3	88.7 ± 0.5	81.2 ± 0.4	49.2 ± 2.4
PLC [10]	92.0 ± 0.6	91.1 ± 0.5	*90.1 ± 0.5*	*91.8 ± 0.4*	*90.7 ± 0.2*	*85.9 ± 0.8*	*75.3 ± 3.1*
FBCCR [15]	90.6 ± 0.3	89.1 ± 0.2	87.3 ± 0.1	90.4 ± 0.2	87.3 ± 0.3	81.0 ± 0.3	62.2 ± 0.4
CNLCU-S [14]	83.2 ± 0.3	81.0 ± 0.6	73.2 ± 1.3	83.0 ± 0.2	78.3 ± 0.7	65.3 ± 3.1	26.4 ± 2.7
Ours	**93.1 ± 0.1**	**92.5 ± 0.1**	**92.0 ± 0.2**	**92.5 ± 0.1**	**91.5 ± 0.2**	**88.9 ± 0.4**	**78.9 ± 1.2**

The best results are boldfaced and the second best are italics

Table 4 Average test accuracy and their standard deviation over three trials for DNN trained on CIFAR-100 with standard method, other LNL methods, and our method

	20% pair flip	30% pair flip	40% pair flip	20% uniform flip	40% uniform flip	60% uniform flip	80% uniform flip
Standard [18]	59.5 ± 0.4	52.9 ± 0.6	44.7 ± 1.3	58.9 ± 0.8	52.1 ± 1.0	42.1 ± 0.7	20.8 ± 1.0
GCE [9, 24]	63.0 ± 0.2	63.2 ± 0.3	61.7 ± 1.2	59.1 ± 0.3	53.3 ± 0.7	36.2 ± 0.7	8.4 ± 0.8
SL [24]	65.6 ± 0.1	65.1 ± 0.1	63.1 ± 0.1	60.0 ± 0.2	53.7 ± 0.1	41.5 ± 0.0	15.0 ± 0.0
MentorNet [18, 42]	64.7 ± 0.2	57.4 ± 0.8	47.4 ± 1.7	63.6 ± 0.5	51.4 ± 1.4	38.7 ± 0.8	17.4 ± 0.9
Co-teaching [18, 22]	63.4 ± 0.9	57.6 ± 0.3	49.2 ± 0.3	66.1 ± 0.5	60.0 ± 0.6	48.3 ± 0.1	16.1 ± 1.1
AUM [19]	69.7 ± 0.2	66.6 ± 0.6	60.2 ± 0.1	65.5 ± 0.2	61.3 ± 0.1	*53.0 ± 0.5*	31.7 ± 0.7
SELFIE [27]	65.7 ± 0.3	64.8 ± 0.5	59.2 ± 0.8	66.5 ± 0.3	63.2 ± 0.4	51.6 ± 0.7	30.6 ± 0.9
Co-teaching+ [25, 26]	60.9 ± 0.3	56.8 ± 0.5	48.6 ± 0.4	64.2 ± 0.4	53.1 ± 0.2	25.3 ± 0.5	10.1 ± 1.2
RoG [28]	67.1 ± 0.6	65.6 ± 0.4	58.8 ± 0.1	63.1 ± 0.3	58.2 ± 0.5	47.4 ± 0.8	20.0 ± 0.9
PENCIL [29]	67.5 ± 0.5	66.0 ± 0.4	61.9 ± 0.4	64.9 ± 0.3	61.3 ± 0.4	46.6 ± 0.7	17.3 ± 0.8
TopoFilter [26]	68.0 ± 0.3	66.7 ± 0.6	62.4 ± 0.2	65.6 ± 0.3	62.0 ± 0.6	47.7 ± 0.5	20.7 ± 1.2
Me-Momentum [12]	65.4 ± 0.8	58.7 ± 1.2	50.5 ± 2.3	*68.0 ± 0.5*	*63.5 ± 0.7*	51.3 ± 1.6	25.8 ± 2.2
LRT [18]	68.3 ± 0.2	61.1 ± 0.5	49.8 ± 0.7	67.8 ± 0.1	60.2 ± 0.8	46.5 ± 1.2	24.6 ± 1.1
PLC [10]	*71.8 ± 0.6*	*68.4 ± 0.7*	*64.3 ± 0.5*	67.3 ± 0.6	60.5 ± 1.3	51.6 ± 0.3	*31.8 ± 0.5*
FBCCR [15]	68.3 ± 0.2	60.5 ± 0.4	50.3 ± 0.6	67.7 ± 0.2	61.7 ± 0.2	49.3 ± 0.8	30.1 ± 0.7
CNLCU-S [14]	43.1 ± 0.9	37.5 ± 1.2	30.3 ± 0.7	46.1 ± 0.3	42.1 ± 0.7	33.4 ± 1.7	18.6 ± 2.1
Ours	**72.5 ± 0.2**	**71.9 ± 0.3**	**69.7 ± 0.2**	**69.4 ± 0.3**	**64.0 ± 0.5**	**53.8 ± 0.3**	**32.6 ± 1.1**

The best results are boldfaced, the second best are italics

Algorithm 1 Overall algorithm for SLRLNL

Input: Observed dataset $\tilde{D}$, initial DNN parameters $\boldsymbol{\theta}^{(0)}$, nearly clean samples' selection parameter n_C, max correction proportion $max\ n_V$, relabeling proportion n_R, warm-up epoch t_w, training epoch t_m. **Output:** $\boldsymbol{\theta}^{(t_m)}$ and relabeled dataset $\check{D}$.

$t \leftarrow 0$
$\check{D} \leftarrow \tilde{D}$
for $t < t_w$ **do**
  $\boldsymbol{\theta}^{(t+1)} \leftarrow$ training $\boldsymbol{\theta}^{(t)}$ using $\tilde{D}$
  $t \leftarrow t + 1$
end for
for $t < t_{train}$ **do**
  search for nearly clean samples $C^{(t)}$ using Eq.(15).
  for mini-batch $\in \check{D}$ **do**
    Update $\boldsymbol{\theta}^{(t)}$ with RLA with Eqs. (13) and (17).
  end for
  for $d \in \check{D}$ **do**
    Calculate samples' learning risk with Eq. (8).
    Update $\check{y}_i^{(t+1)}$ using Eq. (16).
  end for
  $\check{D} \leftarrow [(\mathbf{x}_1, \check{y}_1^{(t+1)}), \ldots, (\mathbf{x}_n, \check{y}_n^{(t+1)})]$
  $t \leftarrow t + 1$
end for
return $\boldsymbol{\theta}^{(t_m)}$,$\check{D}$

noise rate. For uniform flip noise, we replace original labels with other random labels, and for pair flip noise, we replace the sample's label with a similar category.

For datasets that contain real-world label noise, both Animal-10N and Clothing1M contain images crawled from online websites and are poor in label quality. Animal-10N contains 50000 training samples and 5000 testing samples, and the categories for this dataset contain five pairs of animals with similar appearances. Meanwhile, Clothing1M is a much larger dataset, containing 1 million clothing images collected from various online shopping websites, and this dataset contains 14k images with correct labels for validation and 10k images for testing.

We also evaluate our method on the relation extraction task in the field of NLP. The popular method for the relation extraction task is distantly supervised learning [57–59], which annotates the entity pairs in the plain text through the open-source knowledge base. Therefore, labels for distant-supervised datasets are generally erroneous [57–59]. The experiments are conducted on the recently proposed Docred dataset [2], which contains 101873 distantly labeled documents, and 3053, 1000, and 1000 documents strictly annotated by well-trained human annotators for training, validation, and testing, respectively.

Meanwhile, many of the datasets in practice are class imbalanced (e.g., Clothing1M [41] and Docred [2]), and thus, to improve the effectiveness of SLRLNL in practice, for all the datasets, the selection threshold and ranking process related to our method are performed separately for each class; the details can be found in the "**Class imbalance issue**" part of section 4.5.

Backbones

The backbone for CIFAR-10 and CIFAR-100 is ResNet-18 [1], and for Clothing1M, the backbone is ResNet-50 pre-trained on Imagenet. For Animal-10N, we use VGG-19 with batch normalization [60] as our backbone. For Docred, we adopt the BiLSTM from Yao et al. [2] as backbone and GloVe [61] as word embedding.

Baselines

To evaluate the effectiveness of our proposed SLRLNL, we compared our method with several recent or classic loss reweighting methods and label correction methods. In CIFAR-10 and CIFAR-100, the compared loss reweighting methods include MentorNet [42], Generalized Cross Entropy (GCE) [9], Symmetric Loss (SL) [24], Co-teaching [22], Area Under the Margin ranking (AUM) [19], SELFIE [27], Co-teaching+ [25], Robust inference via Generative classifiers (RoG) [28], Probabilistic End-to-end Noise Correction In Labels (PENCIL) [29], TopoFilter [26], Momentum of Memorization (Me-Momentum) [12], and Soft version of Combats Noisy Labels by Concerning Uncertainty (CNLCU-S) [14]. The compared label correction methods include Likelihood Ratio Test (LRT) [18], Progressive Label Correction (PLC) [10], and Forward-Backward Cycle-Consistency Regularization (FBCCR) [15].

In Animal-10N, the compared baseline loss reweighting methods include ActiveBias [56] and Co-teaching [22]. The compared label correction methods include SELFIE [27] and PLC [10].

In Clothing1M, the compared baseline loss reweighting methods include GCE [9], SL [24], MentorNet [42], Co-teaching [22], AUM [19], and CNLCU-S [14]. The compared label correction methods include LRT [18], PLC [10], Universal Probabilistic Model (UPM) [20], and FBCCR [15].

In Docred, the compared baseline LNL methods include Generalized Cross Entropy (GCE) [9], Symmetric Loss (SL) [24], Noisy Label and Negative Sample Robust Loss function (NLNSRL) [11], AUM [19], LRT [18], and PLC [10].

Hyper-parameters' setting

For datasets CIFAR-10 and CIFAR-100, we adopt random flip, brightness, contrast, and saturation data augmentation. We adopt SGD with an initial learning rate of 2e-2 as our optimizer, and the learning rate is divided by 10 in the 50th and 100th epochs. We set the monument to 0.9 and weight decay rate to 5e-4. Then, we train DNN for 150 epochs with a batch size of 64. For the hyper-parameters of our method, we set $n_C = 10$, $n_R = 2$, and set $t_w = 10$ for CIFAR-10, set $n_C = 10$, $n_R = 10$, and $t_w = 30$ for CIFAR-100. To avoid mis-corrections during the label correction process, we set

the $max\ n_V$ to be half the value of the noise rate. For CIFAR-10 with 40% uniform flip label noise, we set $max\ n_V$ to be 20, and for CIFAR-100 with 30% pair flip noise, $max\ n_V$ will be set to 15. And in each epoch, the correction proportion is increased by 2 until it reaches the $max\ n_V$.

For Animal-10N, we adopt random flip as data augmentation. We set SGD as the optimizer and train the DNN for 360 epochs with an initial learning rate of 1e-1, and is divided by 10 in 150th and 250th epochs. The batch size is 64. For the hyper-parameters related to our method, we set $n_C = 10$, $max\ n_V = 4$, $n_R = 2$, and $t_w = 50$. And in each epoch, the correction proportion is increased by 2 until it reaches $max\ n_V$.

For Clothing1M, we follow the experiment setting in Zhang et al. [10], and use a randomly sampled pseudo-balanced subset, including about 260k images. The data augmentation strategies adopted include random crop and random flip. And we train the DNN with a batch size of 32, adopt the SGD as optimizer with a learning rate of 1e-2 for 20 epochs, and we divide the learning rate by 10 at the 3rd, 6th, and 9th epochs. The hyper-parameters of our method are set as $n_C = 10$, $max\ n_V = 10$, $n_R = 5$, and $t_w = 1$. And in each epoch, the correction proportion is increased by 2 until it reaches $max\ n_V$.

For Docred, Adam with a learning rate of 2e-4 is adopted as the optimizer. Each mini-batch contains 30 documents, and we train DNN for 20 and 100 epochs for the distantly supervised dataset and the human-annotated dataset, respectively. For the hyper-parameters of our method, we set $n_C = 1$ and $n_R = 1$ for both datasets, and the correction proportion n_V is directly set to 2. We set $t_w = 0$ for the distantly supervised dataset, and set $t_w = 30$ for the human-annotated dataset.

Experimental results

This section includes the experimental results for SLRLNL and other methods. The source code for SLRLNL can be found in https://github.com/yangbo1973/SLRLNL.

Experimental results for CIFAR-10 and CIFAR-100

This subsection includes the experimental results for artificially noised CIFAR-10 and CIFAR-100. The performance on the test dataset is reported in Tables 3 and 4. Zheng et al. [18] reported the results of Standard, MentorNet, Co-teaching, and LRT in their study. Wu et al. [26] reported the results of Co-teaching+ , RoG, PENCIL, and TopoFilter in their study. Song et al. [27] reported the results of SELFIE against pair flip noise, and the results for uniform flip noise with rate of 20% and 40%. Wang et al. [24] reported the results of GCE and SL against uniform flip noise. Pleiss et al. [19] reported the results of AUM against uniform flip noise. Bai et al. [12] reported the results of Me-Momentum against uniform flip noise with rate of 20% and 40%. Cheng et al. [15] reported the results of FBCCR against pair flip noise with rate of 20% and 40%, and the results for uniform flip noise with rate of 20%, 40%, and 60%. Xia et al. [14] reported the results of CNLCU-S against pair flip noise with rate of 20% and 40% and uniform flip noise with rate of 20% and 40%. And we reproduced the other experimental results that are listed in Tables 3 and 4 but are not reported in the researches mentioned above.

Table 5 Average training time and their standard deviation over three trials for DNN trained on CIFAR-10 and CIFAR-100 during each epoch with standard method and our method

	CIFAR-10	CIFAR-100
Standard	15.7 ± 0.2s	16.1 ± 0.2s
Ours	18.3 ± 0.2s	18.5 ± 0.2s

The results in Tables 3 and 4 depict that the accuracy score of our method exceeds other methods when faced with noisy samples, which indicates that the proposed SLRLNL can better learn the hard clean samples while mitigating the negative impact from noisy samples. These experiments are conducted with RTX 3080, and the training time in Table 5 also depicts that the extra computation time of our method is feasible.

Experimental results for Animal-10N and Clothing1M

To evaluate the effectiveness of our method when dealing with real-world label noise, we conduct experiments on Animal-10N [27] and Clothing1M [41].

Other than the standard method that only utilizes cross-entropy, we compare our proposed method with the existing LNL methods: ActiveBias [56], Co-teaching [22], SELFIE [27], and PLC [10]. The experimental results are reported in Table 6, where the results of Standard, ActiveBias, Co-teaching, and SELFIE are reported in Song et al. [27], and the results of PLC [10] are reported in it's own paper. The results presented in Table 6 demonstrate that our method surpasses the performance of the existing LNL methods.

For Clothing1M, the experimental results for Clothing1M are reported in Table 7, where the results of Standard, GCE, SL, LRT, and PLC are reported in Zhang et al. [10], the results of MentorNet, Co-teaching, and CNLCU-S are reported in Xia et al. [14], and the results of AUM [19], UPM [20], and FBCCR [15] are reported in their respective papers. Our method is compared against these existing popular LNL methods, and the results listed in Table 7 demonstrate that our method is more effective than the existing baselines.

Table 6 Average test accuracy and its standard deviation over three trials for DNN trained on Animal-10N with standard method, other LNL methods, and our method

Methods	Standard [27]	ActiveBias [27, 56]	Co-teaching [22, 27]	SELFIE [27]	PLC [10]	Ours
Accuracy	79.4 ± 0.1	80.5 ± 0.3	80.2 ± 0.1	81.8 ± 0.1	*83.4 ± 0.4*	**86.4 ± 0.2**

The best results are boldfaced and the second best are italics

Table 7 Accuracy score for test dataset for DNN trained on Clothing1M with standard method, other LNL methods, and our method

Methods	Standard [10]	GCE [9, 10]	SL [10, 24]	MentorNet [14, 42]	Co-teaching [14, 22]	AUM [19]	UPM [20]	LRT [10, 18]	PLC [10]	CNLCU-S [14]	FBCCR [15]	Ours
	68.94	69.75	71.02	68.36	69.37	66.50	*74.02*	71.74	74.02	71.57	70.73	**74.15 ± 0.10**

The best results are boldfaced and the second best are underlined. And we conducted three trials for our method to show the average accuracy score and its standard deviation

Table 8 Evaluation results of average F1 score and its standard deviation over three trials on Dev set for DNN trained on Human-annotated and Distantly supervised dataset of Docred with standard method, other LNL methods, and our method

	Human annotated	Distantly supervised (Noisy)
Standard	50.26 ± 0.22	49.17 ± 0.41
GCE	49.91 ± 0.15	49.86 ± 0.14
SL	50.73 ± 0.11	49.78 ± 0.31
NLNSRL	51.25 ± 0.23	*50.85 ± 1.26*
AUM	51.37 ± 0.35	49.98 ± 0.23
LRT	*51.50 ± 0.30*	49.36 ± 0.12
PLC	51.15 ± 0.26	49.74 ± 0.33
Ours	**52.20 ± 0.22**	**51.03 ± 0.13**

The best results are boldfaced, the second best are italics

The experimental results in Animal-10N and Clothing1M indicate that improving DNN's learning for hard clean samples while mitigating label noise is beneficial for DNN's generalization in practice.

Experimental results for Docred

We conducted experiments on both the distantly supervised dataset and the human-annotated dataset for Docred, and evaluated the results on the validation dataset. We reproduced the results of existing LNL baselines, and we used the F1 score to evaluate the performance of DNN. Table 8 displays the experimental results. The results indicate that our method outperforms several baseline LNL methods for both the distantly supervised and human-annotated datasets, which indicates that SLRLNL can also be applicable for the tasks in the NLP field.

Hyper-parameters' analysis

Four hyper-parameters are involved in the proposed SLRLNL: warm-up epoch t_w, nearly clean samples selection proportion n_C, max correction proportion $max\ n_V$, and relabeling proportion n_R for RLA. Each of these hyper-parameters is determined by different characteristics of the training dataset: Setting the hyper-parameter t_w is to better utilize DNN's memorization effect [34, 35], which indicates that the DNN will first learn the samples with a clean label. And this parameter is determined by the learning efficiency of the DNN on the training dataset. Setting the hyper-parameter n_C is to ensure that the samples' learning risk can be calculated accurately, and this parameter is determined by the severity of the label noise. Setting the hyper-parameter n_V is to adjust the effectiveness of the label correction process, and it is also determined by the severity of the label noise. Setting the hyper-parameter n_R is to regulate the effectiveness of the RLA process, and its value is associated with the overall learning difficulty of the training dataset. A training dataset with a higher proportion of samples exhibiting high learning difficulty suggests that this parameter should be set to a higher value. To evaluate the effect of these parameters and determine the best values for our method, we adjust each of these parameters individually while leaving the other three fixed. First, we fix the hyper-parameters as $t_w = 10$, $n_C = 10$, $max\ n_V = 20$, and $n_R = 2$, and then test t_w in the range of $[0, 10, 20, 30]$, n_C in the range of $[5, 10, 20, 40]$, $max\ n_V$ in the range of $[10, 20, 40, 80]$, and n_R in the range of $[2, 5, 10, 20]$. The backbone DNN and optimization parameters are identical with the setting for CIFAR-10 in "Hyper-parameters setting" part of Section 5. The experimental results on CIFAR-10 are reported in Fig. 4.

As shown in Fig. 4a, for the hyper-parameter t_w, when the dataset is heavily noised, starting the correction process in an early time can benefit DNN's performance. And if the dataset contains only a few noisy labels, extending the warm-up process reasonably can mitigate the mis-correction from DNN.

When the dataset contain only a few label noise, the influence of the nearly clean samples' selection proportion n_C on DNN's performance is negligible, as shown in Fig. 4b. How-

Table 9 Average test accuracy and its standard deviation over three trials on CIFAR-10 and CIFAR-100 for SLRLNL with or without RLA

CIFAR-10	20 % pair flip	30 % pair flip	40 % pair flip	20 % uniform flip	40 % uniform flip	60 % uniform flip	80 % uniform flip
With RLA	**93.1** $\pm$ **0.1**	**92.4** $\pm$ **0.1**	**91.4** $\pm$ **0.2**	**92.5** $\pm$ **0.1**	**91.5** $\pm$ **0.2**	**88.9** $\pm$ **0.4**	**78.9** $\pm$ **1.2**
Without RLA	93.0 $\pm$ 0.1	92.2 $\pm$ 0.3	91.2 $\pm$ 0.2	92.2 $\pm$ 0.2	91.3 $\pm$ 0.2	88.3 $\pm$ 0.6	78.0 $\pm$ 1.2
CIFAR-100	20 % pair flip	30 % pair flip	40 % pair flip	20 % uniform flip	40 % uniform flip	60 % uniform flip	80 % uniform flip
With RLA	**72.5** $\pm$ **0.2**	**71.4** $\pm$ **0.8**	**69.7** $\pm$ **0.2**	**69.4** $\pm$ **0.3**	**64.0** $\pm$ **0.5**	**53.8** $\pm$ **0.3**	**32.6** $\pm$ **1.1**
Without RLA	70.9 $\pm$ 0.4	69.0 $\pm$ 0.6	67.0 $\pm$ 0.3	66.4 $\pm$ 0.3	62.0 $\pm$ 0.5	52.1 $\pm$ 0.4	31.1 $\pm$ 1.2

The best results are boldfaced

ever, for the heavily noised dataset, the proportion n_C needs to be tuned to improve the efficacy of label correction.

As for the max correction proportion $max\ n_V$, it is suggested to set $max\ n_V$ to a lower value when the dataset contains only a few label noise to guarantee the precision of correction. When the dataset is heavily noised, $max\ n_V$ can be set to a higher value to eliminate the negative impact from the label noise. However, it is important to note that setting $max\ n_V$ too high can result in a significant amount of miscorrection. This is evident from the results shown in Fig. 4c where setting $max\ n_V$ to 80 lowers the DNN's performance. Therefore, it is crucial to carefully tune this hyper-parameter in practical applications.

Under different level of the label noise, DNN's performance is insensitive to the setting of n_R. However, to extract more information from the hard samples, it is recommended to adjust this parameter to a reasonably higher value when the training dataset contains high proportion of samples with high learning difficulty.

In general, DNN performance remains insensitive to different settings of SLRLNL's hyper-parameters when the label noise is not severe (20%, 40%, and 60% uniform flip label noise). However, when the training dataset is heavily noised (80% uniform flip label noise), the warm-up epoch t_w, the nearly clean samples' selection proportion n_C, and correction proportion n_V cause influence to DNN performance. This indicates that these three hyper-parameters need to be carefully tuned in practice.

Ablation study

An ablation study is conducted in this subsection to validate the effectiveness of RLA in improving DNN's generalization performance. To be specific, we perform SLRLNL on CIFAR-10 and CIFAR-100 with artificial noise, and the DNN adopted is ResNet-18, and hyper-parameters' setting is identical with the "Hyper-parameters setting" part in Section 5. Then, we compare DNN's performance between SLRLNL with RLA ($n_R = 2$ for CIFAR-10, $n_R = 10$ for CIFAR-100) and SLRLNL without RLA ($n_R = 0$ for CIFAR-10 and CIFAR-100), and the results are reported in Table 9.

As shown in Table 9, the proposed RLA process achieved greater improvement in CIFAR-100. This is because the learning difficulty for samples in CIFAR-100 is higher than that in CIFAR-10. Thus, the RLA process, which aims to extract more information from the hard samples, can bring about more significant improvements. This finding indicates that after the risk-based label correction process, the RLA process can effectively prevent the DNN from memorizing hard noisy samples, and enhance the learning for the hard samples, thus improving DNN's generalization performance.

Empirical study of influence of the RLA process on the hard clean samples

The hard clean samples are vital for DNN's generalization performance, since the RLA process focus on relabeling the hard samples, and it is important to validate whether this process will influence DNN's learning for the hard clean samples.

In this subsection, we conducted empirical study experiments on CIFAR-10 and CIFAR-100 with uniform flip or pair flip label noise to test the influence of the RLA process on the hard clean samples. The backbone DNN adopted is ResNet-18, and the hyper-parameters setting is identical with the "Hyper-parameters setting" part in Section 5. Then, we compare DNN's training loss on the hard clean samples. The hard clean samples selected as the samples with the highest 10% learning difficulty from the DNN trained by the clean training dataset, and we keep these samples clean in the training dataset. And the experimental results are listed in Table 10, which show the minimal training loss for the hard clean samples during the training process. Moreover, we also evaluated DNN's performance on the hard clean samples in the test set. These hard clean samples in the test set are selected as the highest 10% learning difficulty from the DNN trained by the clean training dataset. And the experimental results

Table 10 Average minimal training loss (cross-entropy) over three trials on the hard clean samples of CIFAR-10 and CIFAR-100 during training process

CIFAR-10	20 % pair flip	30 % pair flip	40 % pair flip	20 % uniform flip	40 % uniform flip	60 % uniform flip	80 % uniform flip
With RLA	**0.1302**	**0.1745**	**0.2274**	**0.2037**	**0.3762**	**0.5550**	**0.8726**
Without RLA	0.1370	0.1857	0.2471	0.2229	0.3912	0.5755	0.9050
CIFAR-100	20 % pair flip	30 % pair flip	40 % pair flip	20 % uniform flip	40 % uniform flip	60 % uniform flip	80 % uniform flip
With RLA	**0.3213**	**0.4900**	**0.6330**	**0.5920**	**1.2260**	**1.8961**	**2.9281**
Without RLA	0.4071	0.5003	0.6574	0.7098	1.3716	1.9977	2.9692

The best results are boldfaced

Table 11 Average minimal loss (cross-entropy) over three trials on the hard clean samples in the test set of CIFAR-10 and CIFAR-100

CIFAR-10	20 % pair flip	30 % pair flip	40 % pair flip	20 % uniform flip	40 % uniform flip	60 % uniform flip	80 % uniform flip
With RLA	**1.2036**	**1.2778**	**1.3182**	**1.2713**	**1.5669**	**2.3464**	**2.4521**
Without RLA	1.2433	1.3366	1.3530	1.3452	1.6083	2.3982	2.4920
CIFAR-100	20 % pair flip	30 % pair flip	40 % pair flip	20 % uniform flip	40 % uniform flip	60 % uniform flip	80 % uniform flip
With RLA	**1.1722**	**1.2703**	**1.3097**	**1.3013**	**1.6514**	**2.0240**	**3.1386**
Without RLA	1.2751	1.3067	1.3897	1.3844	1.8086	2.1340	3.2123

The best results are boldfaced

are listed in Table 11, which show the minimal loss for the hard clean samples in the test set.

As shown in Tables 10 and 11, for CIFAR-10 and CIFAR-100 datasets, the RLA process reduced the loss for hard clean samples in both the training and test datasets. This indicates that the RLA process encourages the DNN to extract useful information from the hard clean samples. Thus, in summary, although the RLA process may temporarily relabel the hard clean samples and slow down the DNN's learning on them, this process can still improves the learning for the hard clean samples.

Discussion

In contrast to the existing LNL methods that rely on samples' learning difficulty [10, 18, 19, 22], our proposed SLRLNL method can better distinguish noisy samples from hard clean samples. As a result, it effectively mitigates the adverse effects of label noise without compromising the learning progress of hard clean samples, ultimately leading to better performance compared to the existing baseline LNL methods. Moreover, to extract extra information from the hard samples, we proposed the RLA process to prevent the DNN from memorizing the hard noisy samples and further enhancing DNN's learning for hard clean samples.

In general, the experimental results from Subsection 6.1 reveal that our proposed method can effectively improve DNN's performance when the training dataset contains artificially generated noisy labels. And the experimental results from Subsections 6.2 and 6.3 reveal that our method achieved better performance when compared with baseline label correction and loss reweighting methods, which shows that our proposed method can detect noisy samples more effectively in practice.

Conclusion and future work

Conclusion

In conclusion, the primary purpose of the proposed SLRLNL is to detect and correct noisy samples without mis-correcting hard clean samples, and thus improve DNN's performance in practice. Its benefits for DNN accuracy stem from two aspects. First, we utilize the learning risk of samples to correct noisy samples without mis-correcting hard clean samples. Since the latter are vital for DNN generalization, SLRLNL can further improve DNN performance in practice. Second, our proposed RLA can enhance DNN generalization by encouraging the learning of more generalized knowledge about the hard samples, resulting in improved generaliza-

tion performance in practice. The empirical study in section "Empirical study of separating noisy samples from hard clean samples" shows that our method can more accurately separate noisy samples from hard clean samples, and the empirical study in section "Emprical study of influence of the RLA process on the hard clean samples" indicates that the RLA process can enhance the learning for hard clean samples. The experimental results in sections "Experimental results for CIFAR-10 and CIFAR-100", "Experimental results for Animal-10N and Clothing1M", and "Experimental results for Docred" demonstrate the effectiveness of our proposed SLRLNL in improving DNN accuracy when trained with artificial or real-world label noise compared to existing popular LNL methods.

Future work

The proposed SLRLNL is effective in separating noisy samples from hard clean samples, and in the future works, we will incorporate our work with Semi-Supervised Learning method to further improve the performance of our work in practice.

Limitations

This work still has a few limitations. First, although compared to existing LNL methods that are based on the learning difficulty, the proposed SLRLNL can separate noisy samples from hard clean samples more effectively, separating hard noisy samples from hard clean samples still remains challenging for the proposed SLRLNL method. To effectively separate these two types of samples, it is required to construct a more efficient feature extractor, which, in practice, is frequently task-specific and is out of the scope for this paper.

Second, although the experimental results in the ablation study show that the proposed RLA process can improve DNN's performance, it lacks sufficient theoretical evidence to prove that it can reduce the noise rate or contribute to better convergence toward the optimal classifier learned in the clean dataset.

Third, the proposed method in this paper aims to avoid mis-corrections on samples with clean labels but irregular feature patterns. However, in multimodal scenarios, the high learning risk of the samples can also be caused by the inconsistency between the different modalities of the samples' input (e.g., in the image-text multimodal scenario, the image input is mismatched with the text). Under such circumstances, the proposed method can potentially mistake samples with inconsistent input from different modalities as samples with noisy labels, thus degrading the performance of the proposed

Acknowledgements This work is supported by National Natural Science Foundation of China (Project No. 61977013).

Data availability The source code for SLRLNL can be found at https://github.com/yangbo1973/SLRLNL.

Declarations

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

Appendix A: Proof for Eqs. (4) and (5)

Proof This is the proof for Eq. (5). First, when adopting MSE as loss function, after updating the gradient from $\tilde{s}_i$, the update for weight parameters $\mathbf{W}^u$ and bias parameters $\mathbf{B}^u$ in the output layer is

$$\begin{aligned} \triangle \mathbf{W}^u &= -2\alpha(\mathbf{u}_i - \tilde{\mathbf{y}}_i)\mathbf{z}_i \\ \triangle \mathbf{B}^u &= -2\alpha(\mathbf{u}_i - \tilde{\mathbf{y}}_i), \end{aligned} \tag{A.1}$$

where α is the learning rate, $\mathbf{u}_i$, and $\mathbf{z}_i$ are DNN's logits output and penultimate layer output for sample $\tilde{s}_i$, respectively. $\tilde{\mathbf{y}}_i$ is the one-hot form of the observed label $\tilde{y}_i$. According to Eq. (A.1), after updating gradient from $\tilde{s}_i$, the variation for the logits output of sample $s_d = (\mathbf{x}_d, y_d)$ is

$$\triangle \mathbf{u}_{\tilde{s}_i \to s_d} = -2\alpha(\mathbf{z}_i(\mathbf{z}_d + \triangle \mathbf{z}_{\tilde{s}_i \to s_d})^T + 1)(\mathbf{u}_i - \tilde{\mathbf{y}}_i). \tag{A.2}$$

This complete the proof for Eq. (4). In Eq. (A.2), $\triangle \mathbf{z}_{\tilde{s}_i \to s_d}$ is the variation for DNN's penultimate layer output for sample s_d after updating gradient from sample $\tilde{s}_i$, and $\mathbf{z}_d$ is DNN's penultimate layer output for sample s_d. Then, the learning risk from sample $\tilde{s}_i$ in dataset D is

$$\begin{aligned} &\triangle E_{\tilde{s}_i \to D} \\ &= \frac{1}{n} \sum_{s_d \in D} \left(||\mathbf{y}_d - \mathbf{u}_d - \triangle \mathbf{u}_{\tilde{s}_i \to s_d}||_2^2 - ||\mathbf{y}_d - \mathbf{u}_d||_2^2 \right) \\ &= \frac{1}{n} \sum_{s_d \in D} \left(2 \triangle \mathbf{u}_{\tilde{s}_i \to s_d} (\mathbf{u}_d - \mathbf{y}_d)^T + || \triangle \mathbf{u}_{\tilde{s}_i \to s_d}||_2^2 \right) \end{aligned} \tag{A.3}$$

$$= \frac{1}{n}\sum_{s_d\in D} 4\alpha(\mathbf{z}_i(\mathbf{z}_d+\triangle\mathbf{z}_{\tilde{s}_i\to s_d})^T+1)(\tilde{\mathbf{y}}_i-\mathbf{u}_i)(\mathbf{u}_d-\mathbf{y}_d)^T + \frac{1}{n}\sum_{s_d\in D}(2\alpha(\mathbf{z}_i(\mathbf{z}_d+\triangle\mathbf{z}_{\tilde{s}_i\to s_d})^T+1))^2||\mathbf{u}_i-\tilde{\mathbf{y}}_i||_2^2 \quad (A.4)$$

$$= \frac{1}{n}\sum_{s_d\in D} 4\alpha(\mathbf{z}_i(\mathbf{z}_d+\triangle\mathbf{z}_{\tilde{s}_i\to s_d})^T+1)(\mathbf{u}_d-\mathbf{y}_d)(\tilde{\mathbf{y}}_i-\mathbf{u}_i)^T + \frac{1}{n}\sum_{s_d\in D}(2\alpha(\mathbf{z}_i(\mathbf{z}_d+\triangle\mathbf{z}_{\tilde{s}_i\to s_d})^T+1))^2||\mathbf{u}_i-\tilde{\mathbf{y}}_i||_2^2 \quad (A.5)$$

$$= \frac{4\alpha}{n}(\mathbf{z}_i(\mathbf{Z}_D+\triangle\mathbf{Z}_{\tilde{s}_i\to D})^T+1)(\mathbf{U}_D-\mathbf{Y}_D)(\tilde{\mathbf{y}}_i-\mathbf{u}_i)^T + \frac{4\alpha^2}{n}(1+\mathbf{u}_i\mathbf{u}_i^T-2\mathbf{u}_i\tilde{\mathbf{y}}_i^T)\sum_{s_d\in D}(\mathbf{z}_i(\mathbf{z}_d+\triangle\mathbf{z}_{\tilde{s}_i\to s_d})^T+1)^2, \quad (A.6)$$

where $\mathbf{Z}_D \in \mathbb{R}^{n\times N_l}$, $\mathbf{U}_D \in \mathbb{R}^{n\times K}$, and $\mathbf{Y}_D \in \mathbb{R}^{n\times K}$ are the matrices for the clean dataset D's penultimate layer output, logits output, and one-hot form label, respectively. $\mathbf{y}_d$ and $\mathbf{u}_d$ are the one-hot form label and logits output for sample s_d, respectively. $\triangle\mathbf{Z}_{\tilde{s}_i\to D}$ and $\triangle\mathbf{z}_{\tilde{s}_i\to s_d}$ are the variations after updating the gradient from $\tilde{s}_i$ in the penultimate layer output of s_d and D, respectively.

When the learning rate α is set to a small value, $\triangle\mathbf{Z}_{\tilde{s}_i\to D}$ and $\triangle\mathbf{z}_{\tilde{s}_i\to s_d}$ can be ignored compared to $\mathbf{Z}_D$ and $\mathbf{z}_d$, and thus, the learning risk can be represented as

$$\triangle E_{\tilde{s}_i\to D} = \frac{4\alpha}{n}(\mathbf{z}_i(\mathbf{Z}_D)^T+1)(\mathbf{U}_D-\mathbf{Y}_D)(\tilde{\mathbf{y}}_i-\mathbf{u}_i)^T + \frac{4\alpha^2}{n}(1+\mathbf{u}_i\mathbf{u}_i^T-2\mathbf{u}_i\tilde{\mathbf{y}}_i^T)\sum_{s_d\in D}(\mathbf{z}_i(\mathbf{z}_d)^T+1)^2. \quad (A.7)$$

This completes the proof. □

References

1. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: 2016 IEEE conference on computer vision and pattern recognition (CVPR), pp 770– 778 . https://doi.org/10.48550/arXiv.1512.03385
2. Yao Y, Ye D, Li P, Han X, Lin Y, Liu Z, Liu Z, Huang L, Zhou J, Sun M (2019) DocRED: a large-scale document-level relation extraction dataset. In: Proceedings of the 57th annual meeting of the association for computational linguistics (ACL), pp 764–777. https://doi.org/10.18653/v1/P19-1074
3. Cheng P, Wang H, Stojanovic V, Liu F, He S, Shi K (2022) Dissipativity-based finite-time asynchronous output feedback control for wind turbine system via a hidden markov model. Int J Syst Sci 1–13 . https://doi.org/10.1080/00207721.2022.2076171
4. Song X, Wu N, Song S, Stojanovic V (2023) Switching-like event-triggered state estimation for reaction-diffusion neural networks against dos attacks. Neural Process Lett. https://doi.org/10.1007/s11063-023-11189-1
5. Zhuang Z, Tao H, Chen Y, Stojanovic V, Paszke W (2023) An optimal iterative learning control approach for linear systems with nonuniform trial lengths under input constraints. IEEE Trans Syst Man Cybern Syst 3461–3473 . https://doi.org/10.1109/TSMC.2022.3225381
6. Wang S, Wu F, Takyi-Aninakwa P, Fernandez C, Stroe D-I, Huang Q (2023) Improved singular filtering-gaussian process regression-long short-term memory model for whole-life-cycle remaining capacity estimation of lithium-ion batteries adaptive to fast aging and multi-current variations. Energy 284:128677. https://doi.org/10.1016/j.energy.2023.128677
7. Wang S, Fan Y, Jin S, Takyi-Aninakwa P, Fernandez C (2023) Improved anti-noise adaptive long short-term memory neural network modeling for the robust remaining useful life prediction of lithium-ion batteries. Reliab Eng Syst Saf 230:108920. https://doi.org/10.1016/j.ress.2022.108920
8. Ghosh A, Manwani N, Sastry PS (2015) Making risk minimization tolerant to label noise. Neurocomputing 160:93–107. https://doi.org/10.1016/j.neucom.2014.09.081
9. Zhang Z, Sabuncu MR (2018) Generalized cross entropy loss for training deep neural networks with noisy labels. In: Proceedings of the 32nd conference on neural information processing systems (NeurIPS), pp. 8792– 8802 . https://doi.org/10.48550/arXiv.1805.07836
10. Zhang Y, Zheng S, Wu P, Goswami M, Chen C (2021) Learning with feature-dependent label noise: a progressive approach. In: International conference on learning representations (ICLR) . https://doi.org/10.48550/arXiv.2103.07756
11. Deng L, Yang B, Kang Z, Yang S, Wu S (2021) A noisy label and negative sample robust loss function for dnn-based distant supervised relation extraction. Neural Netw 139:358–370. https://doi.org/10.1016/j.neunet.2021.03.030
12. Yingbin B, Tongliang L (2021) Me-momentum: extracting hard confident examples from noisily labeled data. In: Proceedings of the IEEE international conference on computer vision (ICCV), pp 9292–9301 . https://doi.org/10.1109/ICCV48922.2021.00918
13. Kong K, Lee J, Kwak Y, Cho Y-R, Kim S-E, Song W-J (2022) Penalty based robust learning with noisy labels. Neurocomputing 489:112–127. https://doi.org/10.1016/j.neucom.2022.02.030
14. Xia X, Liu T, Han B, Gong M, Yu J, Niu G, Sugiyama M (2022) Sample selection with uncertainty of losses for learning with noisy labels. In: International conference on learning representations (ICLR). https://doi.org/10.48550/arXiv.2106.00445
15. Cheng D, Ning Y, Wang N, Gao X, Yang H, Du Y, Han B, Liu T(2022) Class-dependent label-noise learning with cycle-consistency regularization. In: Advances in neural information processing systems (NeurIPS)
16. Zhu C, Chen W, Peng T, Wang Y, Jin M (2022) Hard sample aware noise robust learning for histopathology image classification. IEEE Trans Med Imaging 41:881–894. https://doi.org/10.1109/TMI.2021.3125459
17. 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/CVF international conference on computer vision (CVPR), pp 3326–3334. https://doi.org/10.1109/ICCV.2019.00342
18. Zheng S, Wu P, Goswami A, Goswami M, Metaxas D, Chen C (2020) Error-bounded correction of noisy labels. In: Proceedings of machine learning research (PMLR), pp 11447–11457 . https://doi.org/10.48550/arXiv.2011.10077
19. Pleiss G, Zhang T, Elenberg ER, Weinberger KQ (2020) Identifying mislabeled data using the area under the margin ranking. Adv Neural Inf Process Syst (NeurIPS) 33:17044–17056. https://doi.org/10.48550/arXiv.2001.10528
20. Wang Q, Han B, Liu T, Niu G, Yang J, Gong C (2021) Tackling instance-dependent label noise via a universal probabilistic model. In: Proceedings of the 35th AAAI conference on artificial intelligence. https://doi.org/10.48550/arXiv.2101.05467

21. Liu T, Tao D (2016) Classification with noisy labels by importance reweighting. IEEE Trans Pattern Anal Mach Intell 38:447–461. https://doi.org/10.1109/TPAMI.2015.2456899
22. Han B, Yao Q, Yu X, Niu G, Xu M, Hu W, Tsang I, Sugiyama M (2018) Co-teaching: robust training of deep neural networks with extremely noisy labels. In: Advances in neural information processing systems (NeurIPS), pp 8535–8545. https://doi.org/10.48550/arXiv.1804.06872
23. Arazo E, Ortego D, Albert P, O'Connor N, McGuinness K (2019) Unsupervised label noise modeling and loss correction. In: Proceedings of the 36th international conference on machine learning (ICML), pp 312–321. https://doi.org/10.48550/arXiv.1904.11238
24. Wang Y, Ma X, Chen Z, Luo Y, Yi J, Bailey J (2019) symmetric cross entropy for robust learning with noisy labels. In: Proceedings of the IEEE/CVF international conference on computer vision (ICCV), pp 322–330. https://doi.org/10.1109/ICCV.2019.00041
25. Yu X, Han B, Yao J, Niu G, Tsang I, Sugiyama M (2019) How does disagreement help generalization against label corruption? In: Proceedings of machine learning research (PMLR), pp 7164–7173 . https://doi.org/10.48550/arXiv.1901.04215
26. Wu P, Zheng S, Goswami M, Metaxas D, Chen C (2020) A topological filter for learning with label noise. Adv Neural Inf Process Syst (NeurIPS) 33:21382–21393. https://doi.org/10.48550/arXiv.2012.04835
27. Song H, Kim M, Lee J-G (2019) Selfie: refurbishing unclean samples for robust deep learning. In: Proceedings of machine learning research (PMLR), pp 5907–5915
28. Lee K, Yun S, Lee K, Lee H, Li B, Shin J (2019) Robust inference via generative classifiers for handling noisy labels. In: Proceedings of the 36th international conference on machine learning (ICML), Vol. 97, pp 3763–3772. https://doi.org/10.48550/arXiv.1901.11300
29. Yi K, Wu J (2019) Probabilistic end-to-end noise correction for learning with noisy labels. In: 2019 IEEE/CVF conference on computer vision and pattern recognition (CVPR), pp 7010–7018. https://doi.org/10.1109/CVPR.2019.00718
30. Cheng J, Liu T, Ramamohanarao K, Tao D (2020) Learning with bounded instance- and label-dependent label noise. In: Proceedings of the 37th international conference on machine learning (ICML). https://doi.org/10.48550/arXiv.1709.03768
31. Lukasik M, Bhojanapalli S, Menon A, Kumar S (2020) Does label smoothing mitigate label noise? In: Proceedings of the 37th international conference on machine learning (ICML), 6448–6458. https://doi.org/10.48550/arXiv.2003.02819
32. Berthon A, Han B, Niu G, Liu T, Sugiyama M (2021) Confidence scores make instance-dependent label-noise learning possible. In: Proceedings of the 38th international conference on machine learning (ICML), pp 825–836. https://doi.org/10.48550/arXiv.2001.03772
33. Li J, Xiong C, Hoi SCH (2021) MoPro: webly supervised learning with momentum prototypes. In: International conference on learning representations (ICLR) . https://doi.org/10.48550/arXiv.2009.07995
34. Zhang C, Bengio S, Hardt M, Recht B, Vinyals O (2017) Understanding deep learning requires rethinking generalization. In: International conference on learning representations (ICLR) . https://doi.org/10.48550/arXiv.1611.03530
35. Arpit D, Jastrzundefinedbski S, Ballas N, Krueger D, Bengio E, Kanwal MS, Maharaj T, Fischer A, Courville A, Bengio Y, Lacoste-Julien S (2017) A closer look at memorization in deep networks. In: Proceedings of the 34th international conference on machine learning (ICML), pp 233–242. https://doi.org/10.48550/arXiv.1706.05394
36. Kremer J, Steenstrup Pedersen K, Igel C (2014) Active learning with support vector machines. Data Min Knowl Disc 4:313–326. https://doi.org/10.1002/widm.1132
37. Harutyunyan H, Achille A, Paolini G, Majumder O, Ravichandran A, Bhotika R, Soatto S (2021) Estimating informativeness of samples with smooth unique information. In: International conference on learning representations (ICLR). https://doi.org/10.48550/arXiv.2101.06640
38. Bengio Y, Louradour J, Collobert R, Weston J (2009) Curriculum learning. In: Proceedings of the 26th international conference on machine learning (ICML), pp 41– 48
39. Settles B (2009) Active learning literature survey
40. Krizhevsky A, Hinton G (2009) Learning multiple layers of features from tiny images. Master's thesis, Department of Computer Science, University of Toronto 1(4)
41. Xiao T, Xia T, Yang Y, Huang C, Wang X (2015) Learning from massive noisy labeled data for image classification. In: Proceedings of the IEEE Conference on computer vision and pattern recognition (CVPR), pp 2691–2699. https://doi.org/10.1109/CVPR.2015.7298885
42. 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: Proceedings of the 35th international conference on machine learning (ICML), pp 2304–2313. https://doi.org/10.48550/arXiv.1712.05055
43. Nguyen DT, Mummadi CK, Ngo TPN, Nguyen THP, Beggel L, Brox T (2020): SELF: learning to filter noisy labels with self-ensembling. In: International conference on learning representations (ICLR) . https://doi.org/10.48550/arXiv.1910.01842
44. Lee J, Chung S-Y (2020) Robust training with ensemble consensus. In: International conference on learning representations (ICLR) . https://doi.org/10.48550/arXiv.1910.09792
45. Ji D, Oh D, Hyun Y, Kwon O-M, Park M-J (2021) How to handle noisy labels for robust learning from uncertainty. Neural Netw 143:209–217. https://doi.org/10.1016/j.neunet.2021.06.012
46. Ghosh A, Kumar H, Sastry PS (2017) Robust loss functions under label noise for deep neural networks. In: Proceedings of the 31th AAAI conference on artificial intelligence, pp 1919–1925. https://doi.org/10.48550/arXiv.1712.09482
47. Toneva M, Sordoni A, Combes RT, Trischler A, Bengio Y, Gordon GJ (2019) An empirical study of example forgetting during deep neural network learning. In: International conference on learning representations (ICLR). https://doi.org/10.48550/arXiv.1812.05159
48. Lin T, Goyal P, Girshick R, He K, Dollár P (2017) Focal loss for dense object detection. In: 2017 IEEE international conference on computer vision (ICCV), pp 2999–3007. https://doi.org/10.1109/TPAMI.2018.2858826
49. Huang S-J, Jin R, Zhou Z-H (2010) Active learning by querying informative and representative examples. Adv Neural Inf Process Syst (NeurIPS) 23:892–900. https://doi.org/10.1109/TPAMI.2014.2307881
50. Koh PW, Liang P (2017) Understanding black-box predictions via influence functions. In: Proceedings of the 34th international conference on machine learning (ICML), pp 1885–1894. https://doi.org/10.48550/arXiv.1703.04730
51. Shorten C, Khoshgoftaar TM (2019) A survey on image data augmentation for deep learning. J Big Data 6:1–48
52. Wei, J., Zou, K(2019) EDA: easy data augmentation techniques for boosting performance on text classification tasks. In: Proceedings of the 2019 conference on empirical methods in natural language processing (EMNLP), pp 6382–6388. https://doi.org/10.18653/v1/D19-1670
53. Lee H, Hwang SJ, Shin J (2020) Self-supervised label augmentation via input transformations. In: Proceedings of the 37th international conference on machine learning (ICML), pp 5714–5724. https://doi.org/10.48550/arXiv.1910.05872
54. Gao W, Wu M, Lam S-K, Xia Q, Zou J (2022) Decoupled self-supervised label augmentation for fully-supervised image clas-

sification. Knowl-Based Syst 235:107605. https://doi.org/10.1016/j.knosys.2021.107605
55. Gui X, Wang W, Tian Z (2021) Towards understanding deep learning from noisy labels with small-loss criterion. In: Proceedings of the 30th international joint conference on artificial intelligence (IJCAI), pp 2469–2475 https://doi.org/10.48550/arXiv.2106.09291
56. Chang H-S, Learned-Miller E, McCallum A (2017) Active bias: training more accurate neural networks by emphasizing high variance samples. Adv Neural Inf Process Syst (NeurIPS) 30:1002–1012. https://doi.org/10.48550/arXiv.1704.07433
57. Li Y, Long G, Shen T, Zhou T, Jiang J (2020) Self-attention enhanced selective gate with entity-aware embedding for distantly supervised relation extraction. In: Proceedings of the AAAI conference on artificial intelligence, Vol. 34, pp 8269–8276. https://doi.org/10.48550/arXiv.1911.11899
58. Nayak T, Ng HT (2020) Effective modeling of encoder-decoder architecture for joint entity and relation extraction. In: Proceedings of the AAAI conference on artificial intelligence, Vol. 34, pp 8528–8535. https://doi.org/10.48550/arXiv.1911.09886
59. Geng Z, Chen G, Han Y, Lu G, Li F (2020) Semantic relation extraction using sequential and tree-structured lstm with attention. Inf Sci 509:183–192. https://doi.org/10.1016/j.ins.2019.09.006
60. Simonyan K, Zisserman A(2015) Very deep convolutional networks for large-scale image recognition. In: International conference on learning representations (ICLR), pp 2691–2699. https://doi.org/10.48550/arXiv.1409.1556
61. Pennington J, Socher R, Manning C (2014) GloVe: global vectors for word representation. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pp 1532–1543. https://doi.org/10.3115/v1/D14-1162