DiscrimLoss: A Universal Loss for Hard Samples and Incorrect Samples Discrimination

Tingting Wu, Xiao Ding, Hao Zhang, Jinglong Gao, Minji Tang, Li Du, Bing Qin, and Ting Liu

Abstract—Given data with label noise (i.e., incorrect data), deep neural networks would gradually memorize the label noise and impair model performance. To relieve this issue, curriculum learning is proposed to improve model performance and generalization by ordering training samples in a meaningful (e.g., easy to hard) sequence. Previous work takes incorrect samples as generic hard ones without discriminating between hard samples (i.e., hard samples in correct data) and incorrect samples. Indeed, a model should learn from hard samples to promote generalization rather than overfit to incorrect ones. In this article, we address this problem by appending a novel loss function DiscrimLoss, on top of the existing task loss. Its main effect is to automatically and stably estimate the importance of easy samples and difficult samples (including hard and incorrect samples) at the early stages of training to improve the model performance. Then, during the following stages, DiscrimLoss is dedicated to discriminating between hard and incorrect samples to improve the model generalization. Such a training strategy can be formulated dynamically in a self-supervised manner, effectively mimicking the main principle of curriculum learning. Experiments on image classification, image regression, text sequence regression, and event relation reasoning demonstrate the versatility and effectiveness of our method, particularly in the presence of diversified noise levels.

Index Terms—Machine learning, deep learning, noisy label, label noise, robust methods.

I. INTRODUCTION

Deep neural networks (DNNs) have achieved extraordinary success in various tasks, such as commonsense reasoning [1], [2], natural language generation [3], speech recognition [4], and image classification [5]. However, as a data-driven method, the success is largely attributed to the large-scale human-annotated datasets. Because obtaining massive and high-quality annotations is extremely labor-consuming and infeasible, many researchers often take an inexpensive and imperfect method as a substitution, such as pattern-based extraction in NLP [6] and automatic labeling in multimedia processing [7], [8], [9], [10]. These methods inevitably involve incorrect samples (i.e., samples with label noise), but DNNs can easily overfit to noisy labels and lead to poor generalization performance [11]. To improve model performance and generalization, curriculum learning (CL) presents a new idea that easy samples should be learned before hard ones during training [12], [13]. This can be done by estimating the importance of each sample directly during training based on the intuitive observation that easy and hard samples behave differently in terms of their respective loss, and can therefore be discriminated.

However, previous studies primarily select or reweight samples to suppress the contribution of incorrect samples to the loss function and regard incorrect samples as ordinary hard ones [14], [15], [16], [17], [18]. In this situation, hard samples mix with incorrect ones. [19] states that hard samples rather than incorrect samples are analogs to support vectors, critical for model generalization.

To facilitate this, we follow the core principle of curriculum learning and append a novel loss function DiscrimLoss on top of the existing task loss to discriminate between hard samples and incorrect ones. In contrast to conventional CL, DiscrimLoss focuses on easy, hard, and incorrect samples (exemplified by Fig. 1) simultaneously and dedicate to separating hard samples and incorrect samples as much as possible. Specifically, DiscrimLoss is a stage-wise learning strategy. At the early training stages, we focus on improving the model performance by discriminating easy and difficult samples (including hard and incorrect samples). As we move into the following training stages,
we aim at enhancing model generalization by separating hard instances and incorrect ones. In this way, a model can learn from correct samples (including easy samples and hard ones) more adequately and avoid memorizing noisy labels. To this end, we present a stable sample reweight method to automatically estimate the difficulty and importance of samples via an adaptive indicator. We learn DiscrimLoss in a self-supervised manner by involving an extra learnable parameter. Experimental results on six datasets under two modalities, comprising benchmark-simulated and real-world label noise datasets, demonstrate consistent gain, which verifies the effectiveness and generality of our method.

II. DISCRIMLOSS

We first introduce an essential overview of DiscrimLoss. Then, we present how sample difficulty can be automatically estimated. As a stage-wise learning strategy, sample importance varies with sample difficulty and training stages. DiscrimLoss attempts to harvest a stable weight for each sample to achieve a robust estimation of sample importance. Finally, we demonstrate the complete form of DiscrimLoss and show a detailed analysis of its principle.

A. Overview

The basic idea of CL is to estimate a priori the difficulty of a given sample by measuring the importance of each sample directly during the training in the form of a weight, such that easy samples with small loss can receive larger weights during the early stages of training. However, no matter hard and incorrect samples would be learned in previous CL methods, as they cannot explicitly discriminate between these two kinds of samples. Based on the observation of large-scale training samples in Fig. 2, we argue that: (1) the loss of correct samples should be smaller than incorrect ones; (2) partial correct samples, which overlap with incorrect samples, can be regarded as hard samples, whose losses are larger than easy ones and smaller than incorrect ones. The challenge is how to accurately separate easy, hard and incorrect samples.

To address this issue, we propose a stage-wise training strategy. As shown in Fig. 3, we first separate easy samples and difficult ones using a threshold \( k_{\text{dyn}} \), where \( k_{\text{dyn}} \in \mathbb{R} \) is an adaptive threshold and initialized with a small value at the early training stages (Stage 1). The learning capability of the model is weak at this stage. Most samples are regarded as difficult ones, which would generate large losses. With the improvement of the model’s learning ability, the value of \( k_{\text{dyn}} \) gradually becomes larger, and more hard samples can be learned by the model. We use an extra learnable parameter \( \delta_i \geq 0 \) to monitor whether the model can learn knowledge from correct samples, where \( \delta_i \) denotes the importance (or weight) of the \( i \)-th sample. When the model cannot gain any knowledge from correct samples, \( k_{\text{dyn}} \) reaches its maximum value, so that we can separate hard and incorrect samples using the maximum value of \( k_{\text{dyn}} \) at this stage (Stage 2).

Formally, let \( \{(x_i, y_i)\}_{i=1}^N \) denote the data, where \( x_i \) is the \( i \)-th sample, and \( y_i \) is its corresponding label. \( l_i = g(M_0(x_i), y_i) \) is the loss of sample \( x_i \) predicted by the model \( M \) with trainable parameters \( \theta \), where \( g(\cdot) \) denotes the task loss function. We append DiscrimLoss \( L_i \) on top of \( l_i \) (referred to as inner loss in the following) to discriminate among easy, hard and incorrect samples. Let \( k_{\text{dyn}} = k_1 \in \mathbb{R} \) (minimum of \( k_{\text{dyn}} \)) denote the threshold at Stage 1 that preferably distinguishes easy samples and difficult samples according to their respective losses. Similarly, let \( k_{\text{dyn}} = k_2 \in \mathbb{R} \) (maximum of \( k_{\text{dyn}} \)) be the threshold at Stage 2 that ideally separates hard samples from incorrect ones based on their corresponding losses.

B. Automatic Estimation of Sample Difficulty

In the early stages of training, the model focuses on fitting data and improving model performance. For sample \( x_i \), if its loss \( l_i < k_1 \), we take the sample \( x_i \) as an easy sample and vice versa. In the following stages of training, the model dedicates to enhancing generalization. For sample \( x_j \), if its loss \( l_j > k_2 \), we take the sample \( x_j \) as an incorrect sample and vice versa. We take the form \( k_{\text{dyn}} \) to unify multiple thresholds utilized in varied training stages, and \( k_{\text{dyn}} \) can be defined as

\[
k_{\text{dyn}} = (a \cdot \text{Tanh}(p(e_c - q)) + a + 1) \cdot k_1, \quad (1)
\]

where \( e_c \) is the current training epoch number, and \( \text{Tanh}(\cdot) \in [-1, 1] \) denotes the classical hyperbolic tangent function. The purple curve in Fig. 3 depicts the relation between \( k_{\text{dyn}} \) and \( e_c, a, p \) and \( q \in \mathbb{R} \) are hyperparameters. \( a \) manipulates the switching threshold and makes \( k_{\text{dyn}} \) fall into the range \( [k_1, k_2] \), where \( k_2 = (1 + 2a)k_1 \), \( p \) controls the switching speed, which leads

Fig. 2. Distribution statistics of normalized losses from samples training on CIFAR-100 with 40% noise with cross-entropy at 80-th epoch.

Fig. 3. Transfer from Stage 1 to Stage 2 by automatically adjusting an adaptive threshold \( k_{\text{dyn}} \) in the training process.
to a more rapid switching when getting smaller and \( q \) governs the switching moment.

C. Stable Estimation of Sample Weight

By estimating sample importance, training samples are arranged in a more meaningful sequence (from easy to hard then to incorrect), which is consistent with the core principle of CL. Specifically, at the early stages of training, we increase the importance of easy samples and decrease the importance of difficult ones. For an easy sample, the operation magnifies its contribution to the loss function and obtains a large \( L_i \), to accelerate the training process. Likewise, at the final stages of training, we increase the importance of hard examples and decrease the importance of incorrect ones. For an incorrect sample, the operation suppresses its contribution to the loss function and gets a small \( L_i \), to prevent overfitting.

However, Fig. 4(a) exhibits incorrect samples are more likely to fluctuate than correct samples when estimating sample importance. In Fig. 4(b), we randomly picked an incorrect sample, whose importance may be comparable with the correct one at many epochs. In this case, the model may confuse correct and incorrect samples, leading to overfitting. Hence, stable sample importance (or weight) estimation is essential. As a learnable parameter, the sample weight \( \delta_i \) is updated by

\[
\delta_i = \delta_i - \tau \frac{\partial L_i}{\partial \delta_i}, \tag{2}
\]

where \( \tau \) denotes the learning rate. Also, \cite{20} states that \( \delta_i \) also represents the confidence of the model’s prediction. From this perspective, we argue that a stable estimation of \( \delta_i \) should consider the following two components.

1) The initial stage of training would be detrimental to a stable estimation of the sample weight: Random initialization of model parameters in DNNs makes neurons more differentiated, resulting in an unstable weight estimation. An intuitive approach to make the weight more stable and reliable is to shrink the present gradient in (2), allowing \( \delta_i \)’s update slower and more cautious and alleviating the impact of model instability. To this end, we propose a simple but effective constraint called Early Suppression (ES), which imposes a short-term suppression on the gradient during the early epochs, and the gradient of \( \delta_i \) can be defined as

\[
\frac{\partial L_i}{\partial \delta_i} = h(ES(e_c)), \quad ES(e_c) = \begin{cases} e_c, & e_c < e_s, \\ 1, & e_c \geq e_s, \end{cases} \tag{3}
\]

where \( e_c \) is the current epoch number. \( e_s \) denotes the total number of suppressed epochs, which is a hyperparameter. \( h(\cdot) \) denotes a general form of \( \delta_i \)’s gradient, whose concrete form will be discussed in detail in the next section.

2) The losses of hard samples and incorrect ones often fluctuate randomly, which has a bad effect on the stable estimation of the sample weight. Specifically, sample importance varies with sample difficulty, and the latter can be characterized by comparing the sample loss with an adaptive threshold \( k_{dyn} \). Therefore, \( \delta_i \)’s update depends on the sample loss \( l_i \). Based on (3), the gradient of \( \delta_i \) can be further defined as \( \frac{\partial L_i}{\partial \delta_i} = h(ES(e_c), l_i) \). However, a training example may experience from being identified correctly to incorrectly (or incorrectly to correctly) throughout learning in one task \cite{19}, which leads to an inconsistent prediction of a single sample. This phenomenon is more common in hard samples. To alleviate the random fluctuation of the prediction confidence caused by only one prediction, we incorporate the historical loss into the model loss, i.e., Historical Loss (HL). Then, we have

\[
\frac{\partial L_i}{\partial \delta_i} = h(ES(e_c), Avg(l_i)), \tag{4}
\]

where \( Avg(l_i) \) denotes the exponential moving average of \( l_i \) with a fixed smoothing parameter \( \rho \). As to easy samples, model predictions in different epochs tend to be consistent. Using the historical loss does not impair the performance of easy samples.

Moreover, regarding the sample importance \( \delta_i \) defined in (2), we increase the \( \delta_i \)’s value for samples where \( l_i < k_{dyn} \), while decreasing it for samples where \( l_i > k_{dyn} \). To achieve this, we incorporate the term \( l_i - k_{dyn} \) as an additional key constraint in the gradient of \( \delta_i \). The form of \( l_i - k_{dyn} \) ensures that the gradient \( \frac{\partial L_i}{\partial \delta_i} \) in (2) remains consistently negative when \( l_i < k_{dyn} \), thereby leading to an increase in \( \delta_i \), namely

\[
\frac{\partial L_i}{\partial \delta_i} = h(ES(e_c), Avg(l_i), l_i - k_{dyn}). \tag{5}
\]

Notably, although sample weights are introduced as learnable parameters, DiscrimLoss is involved only during training. At inference, we take the inner loss as an alternative and hence has no effect on model complexity.

D. Stage-Wise Loss Function

Based on the above analysis, we propose a simple possible formulation of DiscrimLoss that meets design for \( h(\cdot) \), i.e.,

\[
L_i = \frac{ES(e_c)(Avg(l_i) - k_{dyn})}{\delta_i} + \lambda (\log \delta_i)^2, \tag{6}
\]
where \((\log \delta_i)^2\) denotes a regularization term controlled by the hyperparameter \(\lambda > 0\). To better illustrate the underlying principles of DiscrimLoss, we start from the optimization of learnable parameters (i.e., sample weight \(\delta_i\) and model parameters \(\theta\)). Without considering the regularization term, the gradient of the loss w.r.t the sample weight \(\delta_i\) is given by:

\[
\frac{\partial L_i}{\partial \delta_i} = \frac{ES(e_c)(k_{dyn} - \text{Avg}(l_i))}{\delta_i^2}.
\]

Both ES and HL components impose further constraints on \(\delta_i\)’s gradient and make \(\delta_i\) more stable. Based on (7), we summarize the influence on sample weight and model training with different sample difficulties and training stages, as shown in Table I. We take a hard sample in the early stages of training (i.e., training stage 1, \(k_{dyn} = k_1\)) as an example. As a hard sample (i.e., \(\text{Avg}(l_i) > k_1\)), it generates a negative gradient in (7), which leads to the increasing of \(\delta_i\) in (2), i.e., “larger” in Table I. A larger \(\delta_i\) causes a smaller \(L_i\) in (6), which suppresses the loss contribution made by a hard sample. Also, a larger \(\delta_i\) makes the related parameters update slower. Similarly, the gradient of the loss with respect to the model parameters \(\theta\) is given by:

\[
\frac{\partial L_i}{\partial \theta} = \frac{\partial L_i}{\partial l_i} \cdot \frac{\partial l_i}{\partial \theta} = \frac{ES(e_c)}{\delta_i} \cdot \frac{\partial l_i}{\partial \theta},
\]

where \(\frac{\partial l_i}{\partial \theta}\) is the inner loss gradient. The historical loss is a constant in (6), and thus the derivative with respect to \(\theta\) is 0, which does not affect the gradient in (8). In addition, as \(ES(e_c)\) is a constant at each epoch, \(\frac{ES(e_c)}{\delta_i}\) can be regarded as the sample weight, which is ultra-low in the initial stage of training. At this time, the model lacks reliability, and the DiscrimLoss \(L_i\) can decouple from the original loss \(l_i\). When confidence is getting larger, the DiscrimLoss \(L_i\) has a positive correlation with the inner loss \(l_i\).

### III. EXPERIMENTS

We conduct experiments on four different tasks across CV and NLP to test the effectiveness and generality of DiscrimLoss (Sections III-B to III-E). We further validate the effect of our method on model generalization performance (Section III-F). Finally, we investigate the influence of each component of DiscrimLoss through an ablation study (Section III-G).

#### A. Experiment Settings

Before formally introducing the experiments, we first introduce the basic settings of the experiments. Without loss of generality, we mainly focus on both the classification and regression tasks. For the classification tasks, we take the cross-entropy as the inner loss, and utilize accuracy as the evaluation metric. For the regression tasks, the Mean-Square-Error (MSE) loss \(l_2\) or the smooth-L1 loss (smooth-\(l_1\)) are taken as the inner loss, and the mean absolute error (MAE) is adopted as the evaluation metric.

We aim to build robust learning methods on noisy training sets to achieve good generalization performance on clean test sets. To this end, we conduct experiments on two real-world noisy datasets and five manually corrupted datasets, where the latter enables us to examine the specific efficacy of DiscrimLoss under different noise proportions. Following previous works [17], [20], we manually corrupt the dataset by introducing symmetric label noise, as it can be more challenging to avoid incurring such noise into the model during training [21]. The label of each training sample is independently and uniformly flipped to a random class label with a certain probability [20].

In the training process, we empirically initialize \(k_1\) as the global average of the inner loss (simplified as \(k_1 = GA\)), exponential moving average of the inner loss with a fixed smoothing parameter \(\rho' = 0.9\) (simplified as \(k_1 = EMA\)), or a constant \(\eta\) assigned via prior knowledge about the specific task (i.e., \(k_1 = \eta\)). For \(\rho\) in \(\text{Avg}(l_i)\), \(\rho = 0.9\).

For all results in this article, methods are implemented with PyTorch and trained on GeForce RTX 2080Ti GPUs. We report the mean and standard deviation over five runs. All the sample weights are initialized as 1 and are optimized using the standard stochastic gradient descent (SGD). A sample weight \(\delta_i\) will be updated when and only when it is present in a mini-batch. We also refine the hyperparameters for each noise level for datasets with manually corrupted labels. Entire optimal hyperparameter settings are provided in Tables XI and XII of the Appendix A. Hyperparameters are optimized by a common hyperparameter optimization tool, Hyperopt [22].

#### B. Image Classification

**Task and dataset:** Image classification is to identify the target correctly given the images with specific classes, which belongs to a task in CV. We first test the effectiveness of DiscrimLoss by performing image classification experiments on four benchmarks: MNIST [23], CIFAR-10, CIFAR-100 [24], and Clothing1M [25]. We briefly summarize the statistics of four datasets in Table II. Besides, we inject label noise into the first three

| Dataset     | Train Size  | Validation Size | Test Size  | Classes | Image Size |
|-------------|-------------|-----------------|------------|---------|------------|
| MNIST       | 60,000      | -               | 10,000     | 10      | 28 × 28    |
| CIFAR-10    | 50,000      | -               | 10,000     | 100     | 32 × 32    |
| CIFAR-100   | 50,000      | -               | 10,000     | 100     | 32 × 32    |
| Clothing1M  | 1,000,000   | 14,313          | 10,526     | 14      | 224 × 224  |

#### A. Experiment Settings

Before formally introducing the experiments, we first introduce the basic settings of the experiments. Without loss of generality, we mainly focus on both the classification and regression tasks. For the classification tasks, we take the cross-entropy as the inner loss, and utilize accuracy as the evaluation metric. For the regression tasks, the Mean-Square-Error (MSE) loss \(l_2\) or the smooth-L1 loss (smooth-\(l_1\)) are taken as the inner loss, and the mean absolute error (MAE) is adopted as the evaluation metric.

We aim to build robust learning methods on noisy training sets to achieve good generalization performance on clean test sets. To this end, we conduct experiments on two real-world noisy datasets and five manually corrupted datasets, where the latter enables us to examine the specific efficacy of DiscrimLoss under different noise proportions. Following previous works [17], [20], we manually corrupt the dataset by introducing symmetric label noise, as it can be more challenging to avoid incurring such noise into the model during training [21]. The label of each training sample is independently and uniformly flipped to a random class label with a certain probability [20].

In the training process, we empirically initialize \(k_1\) as the global average of the inner loss (simplified as \(k_1 = GA\)), exponential moving average of the inner loss with a fixed smoothing parameter \(\rho' = 0.9\) (simplified as \(k_1 = EMA\)), or a constant \(\eta\) assigned via prior knowledge about the specific task (i.e., \(k_1 = \eta\)). For \(\rho\) in \(\text{Avg}(l_i)\), \(\rho = 0.9\).

For all results in this article, methods are implemented with PyTorch and trained on GeForce RTX 2080Ti GPUs. We report the mean and standard deviation over five runs. All the sample weights are initialized as 1 and are optimized using the standard stochastic gradient descent (SGD). A sample weight \(\delta_i\) will be updated when and only when it is present in a mini-batch. We also refine the hyperparameters for each noise level for datasets with manually corrupted labels. Entire optimal hyperparameter settings are provided in Tables XI and XII of the Appendix A. Hyperparameters are optimized by a common hyperparameter optimization tool, Hyperopt [22].

#### B. Image Classification

**Task and dataset:** Image classification is to identify the target correctly given the images with specific classes, which belongs to a task in CV. We first test the effectiveness of DiscrimLoss by performing image classification experiments on four benchmarks: MNIST [23], CIFAR-10, CIFAR-100 [24], and Clothing1M [25]. We briefly summarize the statistics of four datasets in Table II. Besides, we inject label noise into the first three

| Dataset     | Train Size  | Validation Size | Test Size  | Classes | Image Size |
|-------------|-------------|-----------------|------------|---------|------------|
| MNIST       | 60,000      | -               | 10,000     | 10      | 28 × 28    |
| CIFAR-10    | 50,000      | -               | 10,000     | 100     | 32 × 32    |
| CIFAR-100   | 50,000      | -               | 10,000     | 100     | 32 × 32    |
| Clothing1M  | 1,000,000   | 14,313          | 10,526     | 14      | 224 × 224  |

---

**TABLE I**

| Stage | \(\delta L_i / \delta \delta_i\) | Sample difficulty | \(\delta_i\)'s update | Gradient change |
|-------|----------------------------------|-------------------|------------------------|-----------------|
| 1     | \(ES(e_c)(k_{dyn} - \text{Avg}(l_i)) / \delta_i^2\) |
|       | \(\text{Avg}(l_i) < k_1\) easy smaller larger |
|       | \(\text{Avg}(l_i) > k_1\) hard incorrect larger smaller |
| 2     | \(ES(e_c)(k_{dyn} - \text{Avg}(l_i)) / \delta_i^2\) |
|       | \(\text{Avg}(l_i) < k_2\) easy smaller larger |
|       | \(\text{Avg}(l_i) > k_2\) hard incorrect larger smaller |

**TABLE II**

| Dataset     | Train Size  | Validation Size | Test Size  | Classes | Image Size |
|-------------|-------------|-----------------|------------|---------|------------|
| MNIST       | 60,000      | -               | 10,000     | 10      | 28 × 28    |
| CIFAR-10    | 50,000      | -               | 10,000     | 100     | 32 × 32    |
| CIFAR-100   | 50,000      | -               | 10,000     | 100     | 32 × 32    |
| Clothing1M  | 1,000,000   | 14,313          | 10,526     | 14      | 224 × 224  |
datasets. For MNIST, the noise rates are set to 0%, 20%, 40%, 60%, and 80%. For CIFAR-10 and CIFAR-100, the noise proportions are specified as 0%, 20%, 40%, and 60%.

**Baselines:** For comparison, six categories of baselines are involved. (1) The model trained with vanilla cross-entropy (CE). (2) Multi-network learning methods, addressing the issue of prediction error accumulation in single network learning: (i) Co-teaching [26] maintains two networks. Each network aims to teach the other with the small-loss principle; (ii) Co-teaching+ [27] comprises two networks and keeps prediction disagreement data. Small-loss data is picked and used to cross-train two DNNs; (iii) JoCoR [28] trains two networks by the joint loss with co-regularization to make predictions. The objective is to reduce the diversity of the two networks during training. (3) Sample reweighting methods: (i) MentorNet [16], which requires an extra neural network to learn a curriculum so as to alleviate overfitting on incorrect data; (ii) Data Parameters [17] learns two types of learnable parameters to realize dynamic curriculum learning; (iii) SuperLoss [20] learns a curriculum by reweighting samples according to sample difficulty. Each sample weight can be calculated by the respective model loss. (4) Loss correction methods: Forward & Backward [21], which multiplies the model prediction by a stochastic matrix \( T \) or an inverse of \( T \) and achieves loss correction. (5) Regularization-based methods: (i) SLN [29], which does label perturbation in SGD, provides an implicit regularization effect for training overparameterized DNNs; (ii) CDR [30], which divides all model parameters into critical and non-critical ones, performs different update rules for different types of parameters to hinder the memorization of noisy labels. (6) Re-designing more robust loss methods: CurriculumLoss [18], which proposes a loss to achieve curriculum sample selection. More details are in Appendix B.

**Implementation and Results:** To compare with the relevant state-of-the-art (SOTA) method, for MNIST, we train a LeNet [23] following the experiment settings in [20] for 20 epochs. For CIFAR-10 and CIFAR-100, we use the WideResNet-28-10 model with the same experiment settings as those in [17] for 120 epochs.\(^2\) We utilize the SGD optimizer with batchsize 128, momentum 0.9, weight decay 5e-4, initial learning rate 0.1 for these three datasets. For CIFAR-10 and CIFAR-100, the learning rate decreased to 0.01 after 80 epochs and 0.001 after 100 epochs. For Clothing1M, we train an Imagenet-pretrained ResNet-50 using the SGD optimizer with momentum 0.9, weight decay 1e-3, and batchsize 32 following the common settings in [21] and [29] for 10 epochs. The initial learning rate is 1e-3 and decreased to 1e-4 after 5 epochs.\(^3\)

**MNIST:** As shown in Table III, DiscrimLoss consistently outperforms the baseline methods under different noise proportions on MNIST. In addition, with the increase of the noise level, DiscrimLoss can achieve a statistically significant gain of 0.11%, 0.12%, 0.24%, 0.62% over the SOTA approach, which confirms our motivation that DiscrimLoss can effectively discriminate between hard samples and incorrect samples.

\(^2\)[Online]. Available:https://github.com/apple/ml-data-parameters

\(^3\)[Online]. Available:https://github.com/chenpf1025/SLN

**CIFAR-10 & CIFAR-100:** For CIFAR-10 and CIFAR-100, we observe a similar conclusion, as shown in Table IV. Moreover, we further explore the effectiveness of DiscrimLoss in separating hard samples from incorrect ones. Fig. 5 demonstrates the distribution statistics of normalized losses from samples trained with cross-entropy or DiscrimLoss at 40-th, 80-th, and 120-th epochs. We discover that DiscrimLoss can effectively distinguish incorrect samples from correct ones with an increase in training epochs by “pushing” losses of correct samples to the minimum and incorrect samples to the maximum.

**Clothing1M:** Clothing1M is a large-scale benchmark (1M training images) with real-world noise. The number of images of each class is unbalanced, and we experiment with the noisy-class-balanced sampling, following the settings in [29]. As shown in Table V, DiscrimLoss consistently outperforms all baselines, which indicates that DiscrimLoss can effectively discriminate among samples with various difficulties for instances within a challenging real-world noisy environment.

**Robust loss as the inner loss:** To demonstrate the effectiveness of DiscrimLoss on top of a robust loss, we take SuperLoss as the robust loss and place DiscrimLoss on top of it (i.e., SuperLoss-DiscrimLoss in Table V). As shown in Table V, SuperLoss-DiscrimLoss further achieves higher test accuracy compared to SuperLoss and DiscrimLoss, which indicates DiscrimLoss can boost robust losses effectively by appending on top of them.

**Image Regression**

**Task and dataset:** To further test the generality of DiscrimLoss, we perform experiments on an image regression task (i.e., a CV regression task, aiming to predict the age given face images) on UTKFace [31], containing 23,705 aligned and cropped face images. We follow the data partition settings in [31] and randomly split 90% as the training set and 10% as the test set. Likewise, we manually add different noise levels to the dataset, with specific noise rates of 0%, 20%, 40%, 60%, 80%, respectively.

**Implementation and Results:** We compare with two baselines: (1) No CL trains the model with the vanilla MSE loss or smooth-\(l_1\) loss; (2) SuperLoss [20]. We follow the experiment settings in [20] and train a ResNet-18 model (with a single output) for 100 epochs using SGD with batchsize 128, weight decay 5e-4, and momentum 0.9. We utilize MAE as the evaluation metric and report the results based on \(l_2\) loss and smooth-\(l_1\) loss, which corresponds to a learning rate of 0.001 and 0.1, respectively. Threshold \(k_1\) is assigned as \(k_1 = \text{EMA}\).

| Method          | 0%    | 20%   | 40%   | 60%   | 80%   |
|-----------------|-------|-------|-------|-------|-------|
| MentorNet       | -     | 97.21±0.13 | 93.96±0.76 | -     | -     |
| Co-teaching     | -     | 97.22±0.18 | 94.64±0.33 | -     | -     |
| Co-teaching+    | 98.11±0.07 | 95.87±0.27 | -     | -     | -     |
| JoCoR           | 88.96±0.04 | -     | -     | 84.89±4.55 | -     |
| SuperLoss       | 98.97±0.06 | 98.87±0.05 | 98.40±0.16 | 97.69±0.14 | 95.53±0.18 |
| DiscrimLoss     | 99.08±0.03 | 98.99±0.01 | 98.64±0.01 | 98.11±0.09 | 96.15±0.16 |

**TABLE III**

Test accuracy on MNIST under different proportions of label noise

---

WU et al.: DISCRIMLOSS: A UNIVERSAL LOSS FOR HARD SAMPLES AND INCORRECT SAMPLES DISCRIMINATION 1961

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
TABLE IV
TEST ACCURACY ON CIFAR-10 AND CIFAR-100 UNDER DIFFERENT PROPORTIONS OF LABEL NOISE

| Method       | CIFAR-10  | CIFAR-100 |
|--------------|-----------|-----------|
|              | 0% | 20% | 40% | 60% | 0% | 20% | 40% | 60% |
| MentorNet    | -  | 83.26±0.72 | 78.37±1.73 | -   | 57.27±1.32 | 49.01±2.09 | -   |
| Co-teaching  | -  | 88.20±0.27 | 84.45±0.68 | -   | 61.47±0.41 | 53.44±0.40 | -   |
| Co-teaching+ | -  | 86.54±0.92 | 78.93±0.74 | -   | 72.16±0.34 | 74.30±0.81 | -   |
| Data Parameters | - | 91.10±0.70 | -   | -   | 70.93±0.15 | -   | -   |
| CurriculumLoss | - | 87.94 | -   | -   | 71.24±0.11 | -   | -   |
| SuperLoss    | 94.97±0.03 | 92.98±0.11 | 85.48±0.13 | 78.51±0.10 | 74.34±0.26 | 70.96±0.24 | 62.39±0.17 |
| SLN          | -  | 80.00±0.61 | -   | -   | 50.24±0.41 | -   | -   |
| CDR          | -  | 90.26±0.31 | 87.19±0.43 | -   | 68.63±0.33 | 62.72±0.38 | -   |
| DiscrimLoss  | 95.99±0.17 | 93.67±0.05 | 92.07±0.18 | 86.82±0.06 | 80.77±0.12 | 75.06±0.06 | 71.04±0.07 | 62.46±0.06 |

Fig. 5. Distribution statistics of normalized losses from samples training on CIFAR-100 with 40% noise. (a) Training with cross-entropy at 40-th epoch. (b) Training with cross-entropy at 80-th epoch. (c) Training with cross-entropy at 120-th epoch. (d) Training with DiscrimLoss at 40-th epoch. (e) Training with DiscrimLoss at 80-th epoch. (f) Training with DiscrimLoss at 120-th epoch.

TABLE V
TEST ACCURACY ON CLOTHING1M

| Method       | Test accuracy |
|--------------|---------------|
| CE           | 71.17±0.32    |
| Forward      | 71.28±0.27    |
| Backward     | 71.03±0.33    |
| Co-Teaching  | 72.14±0.28    |
| JoCoR        | 72.30±0.34    |
| SuperLoss    | 71.89±0.44    |
| SLN          | 72.49±0.31    |
| DiscrimLoss  | 73.10±0.11    |
| SuperLoss-DiscrimLoss | 73.79±0.25    |

Top-2 results are in bold.

As shown in Table VI, the DiscrimLoss consistently outperforms both of the baselines by a large margin, regardless of the noise level. This shows the versatility of DiscrimLoss in different task forms.

D. Text Sequence Regression

Task and dataset: To evaluate the effectiveness of our approach in a few-shot situation, we perform experiments on a low-resource text sequence regression task [32], i.e., predicting the sum of digits when given a sequence of symbols of numbers, which can be regarded as an NLP task. “Low resource” depicts the scenario where only a few samples are available for model training. This task is evaluated on the Digit Sum dataset [32], which is a synthetic dataset and comprises 1000, 200, and 200 samples as training, validation, and test sets. We reproduce the generation of the dataset and related baselines following the implementation by [32]. We low and train an LSTM model [34] with hidden units of 256 without peephole connections and a batchsize of 512 for 100 epochs. We choose $l_2$ loss as the inner loss and MAE as the evaluation metric and utilize the AdamW optimizer with $4$[Online]. Available:https://github.com/volkancirik/curriculum-sstb

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
TABLE VI
MEAN ABSOLUTE ERROR (MAE) ON THE TEST SET OF UTKFace UNDER DIFFERENT PROPORTIONS OF LABEL NOISE

| Method        | UTKFace 0% | UTKFace 20% | UTKFace 40% | UTKFace 60% |
|---------------|------------|-------------|-------------|-------------|
| MSE(\ell_2)  | 7.60±0.16  | 10.05±0.41  | 12.47±0.73  | 15.42±1.13  |
| SuperLoss     | 7.24±0.47  | 8.35±0.17   | 9.10±0.33   | 11.74±0.14  |
| DiscrimLoss   | 5.34±0.03  | 6.22±0.04   | 7.94±0.10   | 10.24±0.20  |
| Smooth-\ell_1 | 6.98±0.19  | 7.40±0.18   | 8.38±0.08   | 11.62±0.08  |
| SuperLoss     | 6.74±0.14  | 6.99±0.09   | 7.65±0.06   | 9.86±0.27   |
| DiscrimLoss   | 5.32±0.03  | 5.77±0.03   | 6.86±0.01   | 9.53±0.21   |

TABLE VII
MEAN ABSOLUTE ERROR (MAE) ON THE TEST SET OF DIGIT SUM UNDER DIFFERENT PROPORTIONS OF LABEL NOISE

| Method        | Digit Sum 0% | Digit Sum 20% | Digit Sum 40% | Digit Sum 60% |
|---------------|--------------|---------------|---------------|---------------|
| No CL         | 9.74±0.77    | 20.32±1.07    | 24.97±1.55    | 30.42±1.15    |
| Baby Steps    | 19.89±2.97   | 31.92±4.72    | 31.12±1.04    | 32.63±1.74    |
| DiscrimLoss(k = EMA) | 9.34±1.26    | 18.72±1.44    | 23.50±1.78    | 28.72±0.84    |
| DiscrimLoss(k = GA) | 9.42±1.19    | 15.15±1.16    | 22.87±2.08    | 27.97±1.71    |
| DiscrimLoss(k = 0.5) | 6.45±0.72    | 17.56±0.98    | 24.52±1.33    | 30.01±1.92    |

TABLE VIII
STATISTICS OF WIKIHOW DATASET

| Test | Train Size | Validation Size | Test Size |
|------|------------|-----------------|----------|
| Step Infer. | 336,851    | 166,708         | 792,516  |
| Goal Infer. | 37,427     | 18,523          | 83,612   |
| Step Ordering | 2,250      | 1,703           | 3,100    |

TABLE IX
TEST ACCURACY ON WIKIHOW OF DIFFERENT SUBTASKS

| Method       | Step Infer. | Goal Infer. | Step Ordering |
|--------------|-------------|-------------|---------------|
| Human        | 96.5        | 98.0        | 97.4          |
| BERT         | 87.4        | 79.8        | 81.9          |
| XLNet        | 86.7        | 78.3        | 82.6          |
| GPT-2        | 83.6        | 68.6        | 80.1          |
| RoBERTa*     | 88.2        | 82.0        | 83.5          |
| RoBERTa (Ours) | 88.21±0.08  | 79.53±0.23  | 83.00±0.28    |
*Indicates our re-implementation.

Learning rate 0.1, weight decay 0. We report three sets of results corresponding to different threshold $k_1$ settings.

Due to the extremely limited size of the dataset, the models are particularly vulnerable to overfit the noise within training set. However, as Table VII shows, our method outperforms both baseline methods under all proportions of label noise regardless of the initialization of $k_1$. Especially on noise-free data, DiscrimLoss significantly reduces the error compared with the SOTA baseline. This indicates the advantages of robust sample weights and auto-switching training objectives in our method, which prevent incurring the noise.

E. Event Relation Reasoning

Task and dataset: To further examine the generalization ability of DiscrimLoss, we experiment on a real-world noisy NLP dataset, WikiHow [35]. It involves three subtasks (i.e., Step Infer., Goal Infer., and Step Ordering), aiming at relation reasoning between procedural events (i.e., GOAL-STEP relation, STEP-GOAL relation, and STEP TEMPORAL relation). For example, “learn poses” is a step in the larger goal of “doing yoga” and “buy a yoga mat” typically precedes “learn poses” [35]. In each subtask, the training set is crawled from the wikiHow website and is automatically generated, which inevitably introduces noisy labels. The original WikiHow dataset only divides the training set and the test set. We randomly split a part of the samples from the training set as the validation set, with a ratio of 9:1. The statistics of WikiHow are shown in Table VIII.

Baselines: We compare the proposed method with the following methods: (1) Human [35], which reports the human performance on these tasks. (2) BERT [36], which finetunes the pretrained BERT model on the training set and reports accuracy on the test set. (3) XLNet [37], which is identical to BERT besides finetuning the pretrained XLNet model. (4) GPT-2 [38], which finetunes the pretrained GPT-2 model; the others are the same as BERT. (5) RoBERTa* [39], which finetunes the pretrained RoBERTa model. The remainder is similar to BERT.

Implementation and Results: We reproduce the strongest baseline with the implementation by [35]. We train a RoBERTa-based model using AdamW [40] for three epochs (for Step Infer. subtask), two epochs (for Step Ordering subtask), and five epochs (for Goal Infer. subtask) with learning rate 5e-5, weight decay 0. Because of the large scale of the dataset, we set the batchsize as 48 and 1000 steps as an iteration. Besides, we set $e_s = 3$, $a = 0.2$, $p = 1.2$, $q = 10$, where $e_s$ and $q$ represent the number of iterations. The regularization parameter for DiscrimLoss is assigned as $\lambda = 1e-6$, and the threshold $k_1$ is set to $k_1 = EMA$.

Table IX exhibits a gap of about 10% to 20% between model and human performance, indicating the existence of massive hard instances. In addition, compared to RoBERTa* optimized using cross-entropy loss, DiscrimLoss significantly improves the model performance by 1.15%, 1.97%, and 0.66% on subtasks Step Infer., Goal Infer., and Step Ordering, respectively.

F. Analysis on Model Generalization Improvement

We further explore the effectiveness of DiscrimLoss on improving model generalization. We analyze a manually corrupted dataset besides finetuning the pretrained XLNet model. (4) GPT-2 [38], which finetunes the pretrained GPT-2 model; the others are the same as BERT. (5) RoBERTa* [39], which finetunes the pretrained RoBERTa model. The remainder is similar to BERT.

Implementation and Results: We reproduce the strongest baseline with the implementation by [35]. We train a RoBERTa-based model using AdamW [40] for three epochs (for Step Infer. subtask), two epochs (for Step Ordering subtask), and five epochs (for Goal Infer. subtask) with learning rate 5e-5, weight decay 0. Because of the large scale of the dataset, we set the batchsize as 48 and 1000 steps as an iteration. Besides, we set $e_s = 3$, $a = 0.2$, $p = 1.2$, $q = 10$, where $e_s$ and $q$ represent the number of iterations. The regularization parameter for DiscrimLoss is assigned as $\lambda = 1e-6$, and the threshold $k_1$ is set to $k_1 = EMA$.

Table IX exhibits a gap of about 10% to 20% between model and human performance, indicating the existence of massive hard instances. In addition, compared to RoBERTa* optimized using cross-entropy loss, DiscrimLoss significantly improves the model performance by 1.15%, 1.97%, and 0.66% on subtasks Step Infer., Goal Infer., and Step Ordering, respectively.

F. Analysis on Model Generalization Improvement

We further explore the effectiveness of DiscrimLoss on improving model generalization. We analyze a manually corrupted
CIFAR-100 dataset, with 40% manually injected noisy labels, and compare with two baselines CE and SuperLoss.

As illustrated in Fig. 6(a), the accuracy of DiscrimLoss-optimized model on the manually corrupted training set is 60.38%, close to the theoretical upperbound 60%. However, SuperLoss and CE obtain 83.03% and 99.58% accuracy, respectively, indicating apparent overfitting of the noise. Correspondingly, as demonstrated in Fig. 6(b), the test accuracy of DiscrimLoss level off after about 80 epochs, rather than a significant downward trend like other methods, confirming a better generalization performance of DiscrimLoss. We get a similar conclusion in Fig. 6(c), where after about 80 epochs, the test loss of DiscrimLoss tends to convergence, keeping stabilized. On the contrary, the other methods show a notable increase in test loss manifesting obvious overfitting.

G. Ablation Study

We explore the specific influence of components of DiscrimLoss, and we begin by providing a more detailed discussion of the ES component. In (3), the $ES(e_c)$ exhibits linear growth within suppressed epochs. Moreover, we also explore two typical nonlinear constraints, namely cosine and exponential suppressions. To assess their efficacy, we conduct experiments on CIFAR-10 and CIFAR-100 under 40% and 60% noise, and the results are shown in Table X. For CIFAR-10, the performance of nonlinear suppressions is slightly inferior to that of linear suppression under 40% noise, but marginally superior to linear suppression under 60% noise. A similar trend is observed in the case of CIFAR-100. Given the comparable performance between nonlinear and linear suppressions, we ultimately opt for the simpler linear form.

We proceed to analyze the ES and HL components. Fig. 7 depicts the model’s performance following the removal of either the ES or HL components. It is evident that the exclusion of either results in considerable performance degradation, demonstrating their necessity and importance in DiscrimLoss. Components ES and HL enable a stable estimation of sample weight and better distinguish easy, hard, and incorrect samples compared with existing studies.

Finally, we investigate the adaptive threshold $k_{dyn}$. DiscrimLoss will degenerate into a general reweighting method if the threshold no longer allows adaptive adjustments. In this case, a fixed threshold $k_{fixed}$ is employed in Eq. (6) to differentiate between correct and incorrect samples. We compare the model performance between using $k_{dyn}$ and using $k_{fixed}$ as shown in Fig. 7. We note that the optimal outcome is significantly better than that obtained with $k_{fixed}$. This suggests that utilizing adaptive threshold $k_{dyn}$ can greatly improve the generalization ability of the model when compared to using $k_{fixed}$, manifesting the effectiveness of the stage-wise training strategy.

H. Discussion on Hyperparameters.

The DiscrimLoss contains three crucial hyperparameters, i.e., switching speed $a$, switching threshold $p$, and switching moment $q$, for controlling the switching of different training stages and separating easy, hard, and incorrect samples. We study their effects by adjusting the value of one hyperparameter at a time while keeping the others fixed. In Fig. 8, we provide the experimental results under different noise levels (20% and 40%) on multiple datasets (CIFAR-10, CIFAR-100, and MNIST). To ensure comprehensive coverage, we set a wide range of values for each hyperparameter, encompassing all optimal values for the respective dataset under different noise levels in Table XI.

According to the settings in [20], $k_{fixed} = \log C$, where $C$ represents the number of classes.
As shown in Fig. 8, \( a, p, \) or \( q \) can result in stable and good performance regardless of how they are chosen, demonstrating that model performance is insensitive to their selection despite introducing several hyperparameters. Furthermore, many existing tools for hyperparameter optimization (e.g., Hyperopt [22]) and algorithms (e.g., Successive Halving [41] and Hyperband [42]) can facilitate the determination of optimal hyperparameters. The model can achieve enhanced results by fine-tuning these hyperparameters, as indicated by the highlighted red circles in Fig. 8.

In addition to \( a, p, \) and \( q \), the DiscrimLoss design incorporates several hyperparameters, namely \( \rho, \tau, \lambda, \) and \( e_s \). Among these, the smoothing parameter \( \rho \), the learning rate \( \tau \), and the regularization coefficient \( \lambda \) are not unique to our method and have been...
extensively discussed in previous works, such as [17]. Consequently, determining their optimal values does not require significant effort in hyperparameter tuning, as they can be set based on prior knowledge. On the other hand, the hyperparameter $c_s$, proposed in this article to represent the total number of suppressed epochs (a discrete value), has a limited range of values (typically 2 to 4 in Tables XI and XII), making it relatively easy to identify its optimal setting.

### IV. RELATED WORK

**Curriculum learning:** As a generic learning paradigm, CL benefits the learning process by presenting training examples in a meaningful order and has been widely applied in NLP [43], [44], [45], CV [46], [47], and many other fields [48], [49], [50]. Early works of CL heavily rely on the quality of heuristical prior knowledge to select samples while ignoring feedback from the learner, which triggers a discrepancy between the fixed curriculum and the model. To address this shortcoming, [14] proposes self-paced learning (SPL), which implements sample selection with a weight variable and no longer requires human intervention. The weight variable and model parameters are optimized alternatively. Recently, several studies have been proposed to further refine weighting strategies of SPL [15], [16], [17], [50]. However, they mainly depend on the auxiliary model framework or are limited by specific tasks. Like SPL, DiscrimLoss is also a method of learning a curriculum dynamically without the limitation of inconsistency. Moreover, as an all-purpose CL-based method, DiscrimLoss is suitable for many tasks without modifying model architecture.

**Learning on noisy samples:** To alleviate the influence of incorrect samples, most previous works focus on devising sample reweighting strategy to suppress the weight of incorrect samples [16], [18], [51]. For example, Co-teaching [26] and Co-teaching+ [27] cross-train two DNNs, which have different learning abilities, to filter the noise. DivideMix [52] models the per-sample loss distribution with a mixture model for correct and incorrect samples’ division. [18] proposes CurriculumLoss, which can be deemed as a curriculum sample selection strategy. SuperLoss [20] learns a curriculum by reweighting samples according to sample difficulty. On the other hand, both hard samples and incorrect samples are pervasive but challenging for the model to tell one from the other. However, previous studies simply model incorrect samples as general hard ones, which is not strictly distinguished from the concept: hard samples. In this article, DiscrimLoss tries to discriminate between hard samples and incorrect samples as much as possible, which benefits the model to learn hard examples but avoid memorizing from incorrect ones.

**Hard samples mining** is a popular technique utilized to accelerate convergence and improve the discriminative capabilities of a model [53]. To obtain hard samples, existing approaches often consider inductive bias from a pruning metric perspective, namely that not all training examples are created equal [54]. For instance, in the work of [55], the proposal of EL2N scores allows for the retention of only the most challenging examples, enabling their identification at an early stage of training. Another study [19] observed that hard samples typically transition from being correctly classified to being classified incorrectly (i.e., forgotten), and they performed mining based on forgetting dynamics. Hard sample mining has been extensively explored across various domains, such as contrastive learning [53], [56] and active learning [57], [58]. Diverging from previous research, our focus lies in identifying hard samples within the context of learning from noisy data. This presents a greater challenge as hard and incorrect samples often coexist and exhibit similar behavior during model training.

### V. CONCLUSION

Aiming at the problem of ubiquitous noise, we propose DiscrimLoss for separating hard samples and incorrect samples. Our method employs two-stage training objectives with auto-switching to concurrently guarantee the model performance as well as generalization. Moreover, from the perspective of prediction confidence enhancement, we devise efficient components into DiscrimLoss and generate more stable sample weights, which makes DiscrimLoss outperforms other baselines even in noise-free scenarios. For future work, we are interested in adapting DiscrimLoss to more domains with more datasets. The code will be released at.\(^9\)

### ACKNOWLEDGMENT

We thank the anonymous reviewers for their constructive comments.

### APPENDIX A

**OPTIMAL HYPERPARAMETERS SETTINGS**

We report the optimal settings of hyperparameters unique to our method in Table XI and Table XII.

### APPENDIX B

**BASELINES IN THE IMAGE CLASSIFICATION TASK**

Similar to baselines selection in [30], we do not compare with some SOTA methods like SELF [59] and DivideMix [52] as baselines because of the following reasons. (1) Their proposed methods are aggregations of multiple techniques while we only focus on one. Therefore the comparison is not fair. (2) We focus on essentially solving the problem of memorization of noisy labels, i.e., how to automatically separating hard and incorrect samples as much as possible, but not on boosting the classification performance.

---

\(^9\)[Online]. Available: https://github.com/tangminji/DiscrimLoss
[52] J. Li, R. Socher, and S. C. Hoi, “DivideMix: Learning with noisy labels as semi-supervised learning,” in Proc. Int. Conf. Learn. Representations, 2020.

[53] Y. Suh, B. Han, W. Kim, and K. M. Lee, “Stochastic class-based hard example mining for deep metric learning,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2019, pp. 7251–7259.

[54] B. Sorscher, R. Geirhos, S. Shekhar, S. Ganguli, and A. Morcos, “Beyond neural scaling laws: Beating power law scaling via data pruning,” in Proc. Adv. Neural Inf. Process. Syst., 2022, pp. 19523–19536.

[55] M. Paul, S. Ganguli, and G. K. Dziugaite, “Deep learning on a data diet: Finding important examples early in training,” in Proc. Adv. Neural Inf. Process. Syst., 2021, pp. 20596–20607.

[56] Y. Kalantidis, M. B. Sariyildiz, N. Pion, P. Weinzaepfel, and D. Larlus, “Hard negative mixing for contrastive learning,” in Proc. Adv. Neural Inf. Process. Syst., 2020, pp. 21798–21809.

[57] V. Birodkar, H. Mobahi, and S. Bengio, “Semantic redundancies in image-classification datasets: The 10% you don’t need,” 2019, arXiv:1901.11409.

[58] K. Chitta, J. M. Álvarez, E. Haussmann, and C. Farabet, “Training data subset search with ensemble active learning,” IEEE Trans. Intell. Transp. Syst., vol. 23, no. 9, pp. 14741–14752, Sep. 2022.

[59] D. T. Nguyen et al., “Self: Learning to filter noisy labels with self-ensembling,” in Proc. Int. Conf. Learn. Representations, 2020. [Online]. Available: https://openreview.net/forum?id=HkgsPhNYPS

Tingting Wu received the master’s degree from the University of Science and Technology of China, Hefei, China. She is currently working toward the Ph.D. degree with the School of Computer Science and Technology, Harbin Institute of Technology, Harbin, China. Her research interests include machine learning, label-noise learning, and natural language processing applications.

Xiao Ding received the Ph.D. degree from the School of Computer Science and Technology, Harbin Institute of Technology, Harbin, China. He is currently a Full Professor with the Department of Computer Science, Harbin Institute of Technology, Harbin, China. His research interests include natural language processing, text mining, social computing, and commonsense inference.

Hao Zhang received the B.S. and Ph.D. degrees from the University of Science and Technology of China, Hefei, China, in 2014. He is currently an Associate Researcher with the Computer Science Department, Harbin Institute of Technology, Harbin, China. His research interests include parallel computing, federated learning, pervasive computing, and deep learning application.

Jinlong Gao received the bachelor’s degree in computer science and technology in 2020 from the Harbin Institute of Technology, Harbin, China, where he is currently working toward the Ph.D. degree in computer science and technology. His research interests include natural language processing, causality identification, and event-centric reasoning.

Minji Tang received the bachelor’s degree in computer science and technology in 2022 from the Harbin Institute of Technology, Harbin, China, where he is currently working toward the master’s degree in computer science and technology. His research interests include natural language processing, label-noise learning, and event extraction.

Li Du received the master’s degree in statistics from Fudan University, Shanghai, China, in 2017. Since 2018, he has been working toward the Ph.D. degree with the Department of Computer Science, Harbin Institute of Technology, Harbin, China. His research interests include natural language processing, commonsense reasoning, and pretrained language models.

Bing Qin received the Ph.D. degree from the Department of Computer Science, Harbin Institute of Technology, Harbin, China, in 2005. She is currently a Full Professor with the Department of Computer Science, and the Director of the Research Center for Social Computing and Information Retrieval (HIT-SCIR), Harbin Institute of Technology. Her research interests include natural language processing, information extraction, document-level discourse analysis, and sentiment analysis.

Ting Liu received the Ph.D. degree from the Department of Computer Science, Harbin Institute of Technology, Harbin, China, in 1998. He is currently a Full Professor with the Department of Computer Science, and the Director of the Research Center for Social Computing and Information Retrieval (HIT-SCIR), Harbin Institute of Technology. His research interests include information retrieval, natural language processing, and social media analysis.