Learning with Noisy Labels Using Hyperspherical Margin Weighting

Shuo Zhang[1,2,3], **Yuwen Li**[1,2], **Zhongyu Wang**[1,2], **Jianqing Li**[1,2], **Chengyu Liu**[1,2*]

[1]School of Instrument Science and Engineering, Southeast University, China
[2]State Key Laboratory of Digital Medical Engineering, Southeast University, China
[3]School of Biological Science and Medical Engineering, Southeast University, China
{zs_techo, liyuwen, zhongyu, ljq, chengyu}@seu.edu.cn

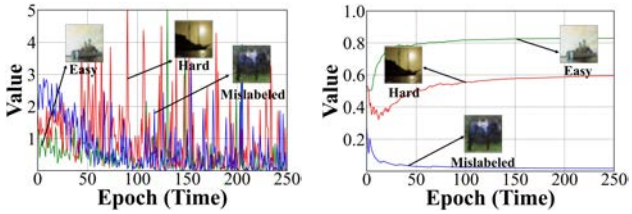
Abstract

Datasets often include noisy labels, but learning from them is difficult. Since mislabeled examples usually have larger loss values in training, the small-loss trick is regarded as a standard metric to identify the clean example from the training set for better performance. Nonetheless, this proposal ignores that some clean but hard-to-learn examples also generate large losses. They could be misidentified by this criterion. In this paper, we propose a new metric called the Integrated Area Margin (IAM), which is superior to the traditional small-loss trick, particularly in recognizing the clean but hard-to-learn examples. According to the IAM, we further offer the Hyperspherical Margin Weighting (HMW) approach. It is a new sample weighting strategy that restructures the importance of each example. It should be highlighted that our approach is universal and can strengthen various methods in this field. Experiments on both benchmark and real-world datasets indicate that our HMW outperforms many state-of-the-art approaches in learning with noisy label tasks. Codes are available at https://github.com/Zhangshuojackpot/HMW.

Introduction

Supervised machine learning has revolutionized artificial intelligence, allowing us to build predictive models that can recognize patterns and make decisions. The effectiveness of models depends heavily on the quality of the training data. Nonetheless, labeling precise annotations is time-consuming and prone to mistakes. Even accepted high-quality datasets, such as ImageNet (Deng et al. 2009), include erroneous labels (Northcutt, Athalye, and Mueller 2021). The presence of noisy labels can dramatically degrade the performance of models. Therefore, developing algorithms to resist them is vital. Many approaches have been proposed. Since sample weighting methods do not suffer from imprecise noise estimates or duplicated model architecture, this kind of strategy is the main focus of this paper.

In fact, the sample weighting method achieves learning with noisy labels by decreasing the importance of the mislabeled examples (giving them a small weight or even removing them) in the training procedure. *Apparently, the recognition of the mislabeled example is significant. It directly determines which examples are ignored. In most cases, the loss value in the convergence is regarded as a tool to identify, called the small-loss trick*. This metric treats the large-loss example as a mislabeled one. Based on this thinking, various approaches have been developed: Co-teaching (Han et al. 2018) and Co-teaching+ (Yu et al. 2019) offered two networks, but each network extracted examples with small losses and fed them to its peer network for further training. DivideMix (Li, Socher, and Hoi 2020a) employed two-component and one-dimensional Gaussian mixture models to fit the loss values of examples and turned noisy samples into labeled and unlabeled sets. JoCoR (Wei et al. 2020) followed this motivation and tried to reduce the diversity of the two networks, making predictions about them closer. In PuriDivER (Bang et al. 2022), the authors applied the small-loss trick for purity-aware sampling.



(a) Traditional Small-Loss Trick (b) Our IAM

Figure 1: The comparison of the traditional small-loss trick metric and the proposed IAM metric. It is obvious that our IAM is more powerful in identifying mislabeled examples, especially from hard-to-learn examples.

Nonetheless, the small-loss trick is actually defective. This criterion does not consider the influence of clean but hard-to-learn examples. Let us take some images in CIFAR10 as an example to illustrate. When various images that are labeled to"ship" are employed for training, their loss curves are shown in Fig. 1(a). As can be seen, the clean and easy-to-learn "ship" (the green line) has a small loss and is distinguishable from the clean but hard-to-learn "ship" (the red line) and the mislabeled "ship" (the blue line). However, identifying the red line from the blue line is tough, especially in the late stages of training. *It reveals that the small-loss trick is not sensitive enough to distinguish the clean but*

*Corresponding author.

hard-to-learn example from the mislabeled example. Nevertheless, even some recent works (Bang et al. 2022; Garg et al. 2023) *are still triggered by it. Designing new criteria to avoid it is urgent*. Recently, the Area Under the Margin (AUM) ranking has been offered to address this problem (Pleiss et al. 2020). Motivated by it, in this paper, we point out that mislabeled examples and hard-to-learn examples can not only be classified by the AUM ranking but also by the proposed Top-K Under the Margin (TKUM) ranking. Combined with them, we create a new metric called the Integrated Area Margin (IAM). Moreover, we further develop a new sample weighting method called Hyperspherical Margin Weighting (HMW). It should be highlighted that our method is universal and can be applied with some state-of-the-art (SOTA) approaches to improve their performances. Our major contributions can be summarized as follows:

- We provide a new criterion called the **IAM**, which is more powerful in distinguishing clean and mislabeled examples than the traditional small-loss trick in learning with noisy labels, especially in distinguishing **hard-to-learn** and **mislabeled** examples.
- Based on the IAM, we transfer it to the hyperspherical space and propose the **HMW** method. It redistributes the importance of each example, resulting in clean examples having greater contributions than mislabeled examples.
- Our HMW is **general** and can enhance **various techniques**. Experiments on benchmark and real-world datasets demonstrate that the HMW can perform better than many SOTA approaches in this field.

Related Work

Besides the sample weighting which has been illustrated above, three other branches should be introduced. Specifically, robust architecture is a hopeful branch. Its concept is to use a noise adaptation layer on top of a deep neural network (DNN) to either learn the process of label transition or create an architecture that can support various types of label noise. For example, Webly learning (Chen and Gupta 2015) taught the DNN to retrieve only simple examples and utilized the confusion matrices of all training examples as the initial weight in the noise adaptation layer. Recently, contrastive-additive noise networks were introduced, such as MOIT (Ortego et al. 2021a), Sel-CL (Li et al. 2022), ProtoMix (Li, Xiong, and Hoi 2021a), *etc.*. Another branch is robust regularization. It aims to prevent DNNs from overfitting mislabeled examples by imposing training restrictions. Bilevel learning (Jenni and Favaro 2018) used a bilevel optimization strategy to regularize the overfitting of a model using a clean validation dataset. Besides, the robust loss design is also a great motivation to solve this problem. Its purpose is to adjust the loss value based on the confidence of the loss or label by designing a new loss function. Backward (Patrini et al. 2017a) was one of the first methods that approximated the noise transition matrix by using the softmax output of a DNN trained without loss correction. Then, it updated the DNN with a revised loss based on the estimated matrix. After that, Generalized Cross-Entropy (GCE) (Zhang and Sabuncu 2018) was proposed. It could be seen as a combination of CCE and MAE.

Algorithm

Preliminaries

We are given a dataset $D = \{(x_i, y_i)\}_{i=1}^{N}$ to train a K-class ($K > 2$) classification network $f(\cdot; \Theta_f)$, where N is the size of the set. x_i is the sample of the i-th example, and $y_i \in [1, K]$ is the corresponding label. $o \in \{0, 1\}$ is the q-th value of the one-hot label. As such, the posterior probability of y_i can be obtained by the softmax function as

$$p(y_i|x_i) = \frac{e^{f_{y_i}(x_i)}}{\sum_{q=1}^{K} e^{f_q(x_i)}}, \tag{1}$$

where $f_{y_i}(x_i)$ denotes the logit of the label y_i. If we employ CCE for training, the loss L can be written as

$$L = -\frac{1}{N}\sum_{i=1}^{N} \log(p(y_i|x_i)). \tag{2}$$

L should be minimized to fit $f(\cdot; \Theta_f)$, and it is usually regarded as a paradigm to train a model.

However, when the dataset is noisily labeled (let $\tilde{D}$ and $\tilde{y}_i$ denote the noisy dataset and label, respectively), this pattern is defective. Specifically, if we illustrate it from a gradient perspective, the contributed gradient of the example $(x_i, \tilde{y}_i)$ can be written as

$$\nabla L(\Theta_f) = \underbrace{(p(q|x_i) - o)}_{\text{scale term}} \cdot \nabla f(\Theta_f). \tag{3}$$

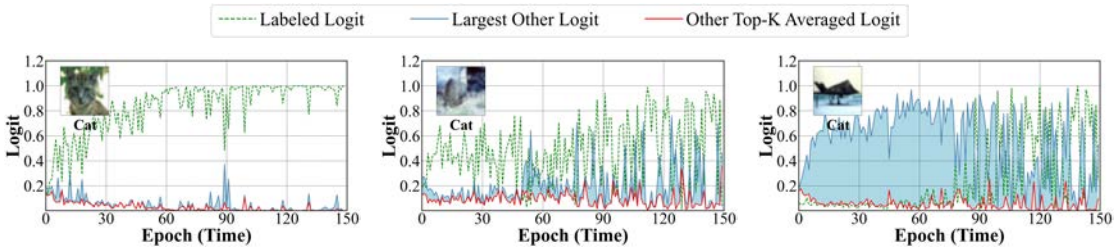
$(p(q|x_i) - o)$ is a scale term. As the convergence goes on, samples with incorrect labels have larger weights than those with correct labels. In other words, the traditional approach pays more attention to those examples that generate larger loss values, even if they are mislabeled (Liu et al. 2020a).

Integrated Area Margin Metric

According to the above analysis, decreasing the contributions of mislabeled examples in training can yield better performance. Nevertheless, many favored methods (Yu et al. 2019; Li, Socher, and Hoi 2020a; Bang et al. 2022) identify mislabeled examples by the loss value. They think that the large-loss examples are mislabeled and give them small weights. This consideration neglects that large losses can also be produced by clean but hard-to-learn examples and leads some SOTA sample weighting methods to be partially robust. In this paper, we create a new criterion, the IAM, to recognize mislabeled examples. As such, we first illustrate the IAM slice S at the epoch t written as:

$$S^{(t)}(x_i) = \alpha \cdot \underbrace{M^{(t)}(x_i)}_{\text{AUM ranking}} - \beta \cdot \underbrace{U^{(t)}(x_i)}_{\text{TKUM ranking}}. \tag{4}$$

where α and β are scale hyperparameters. According to (Pleiss et al. 2020), the clean example (even the hard-to-learn sample) and mislabeled example are distinctive under the difference value of the labeled logit and the largest other



(a) Easy (b) Hard (c) Mislabeled

Figure 2: The curves of focused logits with the easy-to-learn, hard-to-learn, and mislabeled examples. The blue area represents our Top-K Under the Margin (TKUM) ranking.

logit, and the AUM ranking is proposed. It is applied to the first term of our IAM. At epoch t, $M^{(t)}(x_i)$ can be written as

$$M^{(t)}(x_i) = \underbrace{f^{(t)}_{\tilde{y}_i}(x_i)}_{\text{labeled logit}} - \underbrace{max_{c\neq\tilde{y}_i} f^{(t)}_c(x_i)}_{\text{largest other logit}}. \tag{5}$$

Besides, we discover that clean but hard-to-learn examples and mislabeled samples are also distinctive from another perspective. As shown in Fig. 2, the blue line denotes the largest other logit, and the red line denotes the other top-k averaged logit (excluding the labeled logit and the largest other logit). It is obvious that the red line always follows the blue line with the clean example (both the easy-to-learn and the hard-to-learn) but not with the mislabeled example. *Accordingly, the area between the blue line and the red line can also be applied to distinguish clean and mislabeled examples, even if the clean one is hard to learn*. This new criterion is called TKUM ranking. We apply it to the second term of our IAM. As such, at epoch t, $U^{(t)}(x_i)$ can be written as

$$U^{(t)}(x_i) = \underbrace{max_{c\neq\tilde{y}_i} f^{(t)}_c(x_i)}_{\text{largest other logit}} - \underbrace{\frac{1}{k}\sum_{l\neq\tilde{y}_i,c}^{top\text{-}k} f^{(t)}_l(x_i)}_{\text{other top-k averaged logit}}. \tag{6}$$

In this paper, we set $k = 1$ in all experiments. However, another negative appearance is observed in our exploration. That is the AUM ranking and the TKUM ranking are strong to identify the mislabeled example at the beginning of training but gradually weaken in the end (shown in Fig. 2). To overcome it, a cosine-annealing (CA) strategy is designed to decrease the importance of the IAM slice generated late in training. As such, at the epoch t, the IAM metric can finally be written as

$$\mathbf{IAM}(x_i) = \sum_{j=1}^{t} S^{(j)}(x_i) \cdot B[j], \tag{7}$$

$$B = softmax([\epsilon_1, \epsilon_2, ..., \epsilon_j, ..., \epsilon_t]), \tag{8}$$

$$\epsilon_j = 1 + cos(\frac{j}{T} \cdot \pi), \tag{9}$$

where T denotes the training epoch, and B denotes the decay probability matrix at the epoch t.

Hyperspherical Margin Weighting

The above sections have illustrated the motivations and implementations of our IAM metric. Here, we attempt to introduce a new sample weighting method built on it. It is obvious that the IAM is calculated by the logits, which are generated with the inner product between the input and the weight of the final layer. However, many researchers (Wang et al. 2018; Deng et al. 2019; Wen et al. 2022) revealed that learning in a hyperspherical space could offer more outstanding performance, especially in learning with noisy labels (Ke et al. 2022). Encouraged by them, we further design the hyperspherical version of our IAM and develop the HMW method. Let $f'(\cdot;\Theta_{f'})$ and W_q denote the projection before the final layer and the weight of a certain label q at the final layer, respectively. At epoch t, the cosine angle distance $d^{(t)}_q(x_i)$ between the features of x_i and W_q can be written as

$$d^{(t)}_q(x_i) = \frac{W_q \cdot f'^{(t)}(x_i)}{\|W_q\|\|f'^{(t)}(x_i)\|}. \tag{10}$$

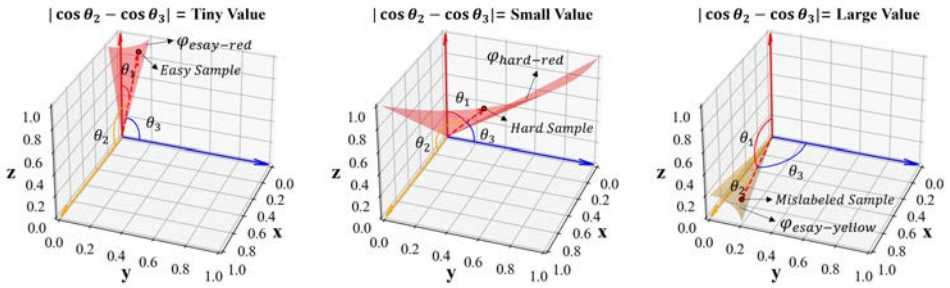
For more convenient to be weights, $d^{(t)}_q(x_i)$ is adjusted to $\widehat{d}^{(t)}_q(x_i)$ written as

$$\widehat{d}^{(t)}_q(x_i) = \frac{1}{2} \times (1 - d^{(t)}_q(x_i)). \tag{11}$$

Then, we replace the mapping f in Equations (5) and (6) with the mapping $\widehat{d}$ and obtain hyperspherical AUM ranking $\tilde{M}^{(t)}(x_i)$ and hyperspherical TKUM ranking $\tilde{U}^{(t)}(x_i)$. Notably, since the larger $f^{(t)}_{\tilde{y}_i}(x_i)$ represents the smaller $\widehat{d}^{(t)}_q(x_i)$, we transform the max-search strategy in Equations (5) and (6) to the min-search strategy for consistency. After that, we further adjust their value domains and monotonicity in Equation (4) to produce the hyperspherical IAM slice S_h. As such, at the epoch t, it can be written as

$$\begin{aligned} S^{(t)}_h(x_i) = &\frac{1}{2} \times (sigmoid(-\alpha \cdot \tilde{M}^{(t)}(x_i)) \\ &+ exp(\beta \cdot \tilde{U}^{(t)}(x_i))), \end{aligned} \tag{12}$$

Owing to the above transforms, S_h is limited to (0, 1). Finally, with the same CA strategy shown in Equations (7) to (9), the hyperspherical IAM metric is generated. A Hyperspherical Margin (HM) weight γ_i is obtained to participate



(a) Easy (b) Hard (c) Mislabeled

Figure 3: The theoretical illustration of our method in various situations.

in training. As such, if CCE is employed, the loss function L can be adjusted by the HM weight as

$$L = -\frac{1}{N}\sum_{i=1}^{N}\gamma_i \cdot \log(p(y_i|x_i)). \tag{13}$$

The entire procedure of using HM weights to redistribute training examples is called our HMW method.

Discussion

Why Does the IAM Work? It should be highlighted that the traditional small-loss trick easily misidentifies hard-to-learn examples as mislabeled examples. However, owing to two outstanding sub-metrics, our IAM can effectively improve this flaw. *Strikingly, we provide new insight and offer the TKUM ranking*. Let us give an intuitive explanation of it. In fact, one easy-to-learn example is close with only one label embedding in the feature space in most cases. Inversely, the main reason to make training hard can be regarded as some ambiguous feature representations, which lead to one hard-to-learn example being close to multiple label embeddings. Specifically speaking, as shown in Fig. 3, we assume that the coordinate axes denote the label embeddings in the penultimate layer. Various colors denote various categories. A sample labeled with the red category appears. θ_1, θ_2, and θ_3 denote the angles between this sample and label embeddings of the red, yellow, and blue categories, respectively. When the sample is clean and easy to learn, it distributes around the conical surface $\varphi_{easy-red}$, which stays near the red axis. Removing the θ_1 produced by the sample and labeled embedding, $|\theta_2 - \theta_3|$ is a tiny value (shown in Fig. 3(a)). When it is clean but hard to learn, it distributes around the conical surface $\varphi_{hard-red}$, which stays away from the red axis. Removing the θ_1, $|\theta_2 - \theta_3|$ is a small value (shown in Fig. 3(b)). When it belongs to the yellow category in truth but is mislabeled as the red category, it distributes around the conical surface $\varphi_{easy-yellow}$, which stays near the yellow axis. Removing the θ_1, $|\theta_2 - \theta_3|$ is a large value (shown in Fig. 3(c)). These differences trigger us to propose the TKUM ranking. The experimental results in Fig. 2 also support it. Additionally, *we first reveal that the difficulty of dividing the hard-to-learn and mislabeled examples increases as training proceeds. To deal with it, the IAM generates a CA strategy to improve the importance of the AUM and the TKUM rankings at the beginning of training.* These advantages are significant for learning with noisy label tasks.

IAM versus AUM. Both our IAM and the AUM metrics seem to have some abilities to identify hard-to-learn examples for mislabeled examples, but they have the following distinctions: 1) In the paper of the AUM (Pleiss et al. 2020), the authors propose only one ranking method of the AUM that is sensitive to the hard-to-learn examples. For extending this thinking, we offer a new ranking strategy of the TKUM that is also adept at detecting them. These two metrics are finally combined to generate the IAM metric. *The AUM and the TKUM rankings can cooperate and achieve better results. Our IAM metric is a more universal version of the AUM metric*. 2) As shown in Fig. 2, the strength of the AUM metric gradually decreases as convergence continues. *In our IAM, the CA strategy is further employed to enhance this defect*. 3) Motivated by the robustness of hyperspherical space towards label noises (Ke et al. 2022), *we transform our IAM into this space for better performance but the AUM not.* 4) The strategies to apply these two metrics are different. In this paper, we employ our IAM metric as a foundation to build a new sample weighting method. However, in the paper of the AUM (Pleiss et al. 2020), the authors employ the AUM metric to produce a threshold and develop a sample selection method. *In our opinion, the HMW is safer, since it can adaptively retain all examples for training, certainly limiting the mistakes caused by the criteria itself.*

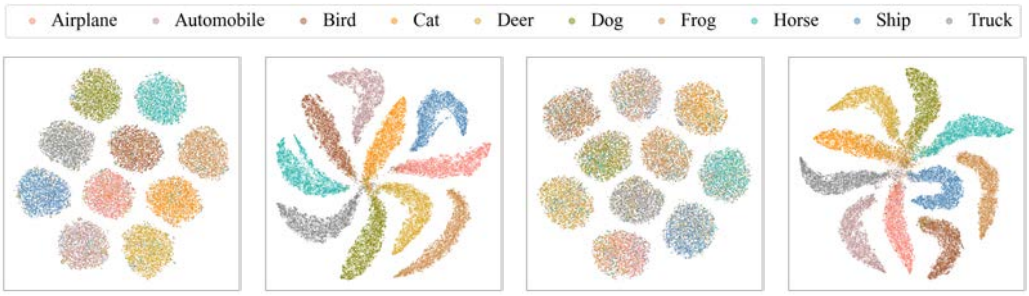
Experiments

In this section, we first illustrate the effectiveness of our HMW through various empirical understanding experiments. Then, the HMW is compared with some SOTA methods in this field on both benchmark datasets and real-world datasets. Besides, to discuss the hyperparameter settings of our strategy, some ablation studies are also conducted.

Noise Settings: Our noise settings are consistent with those in (Karim et al. 2022a). η denotes the noise rate in this paper.

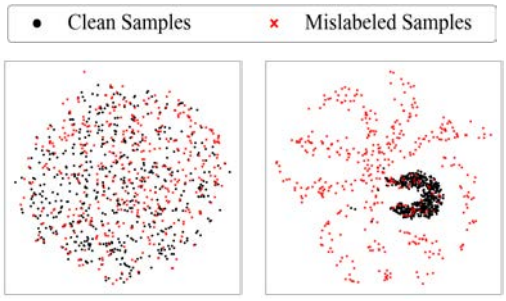
Empirical Understandings

Experimental Setup: We practice the CCE loss and its enhanced version with the HMW on CIFAR-10 as an example to explore. The related experimental settings are consistent with those in (Karim et al. 2022a). After training,



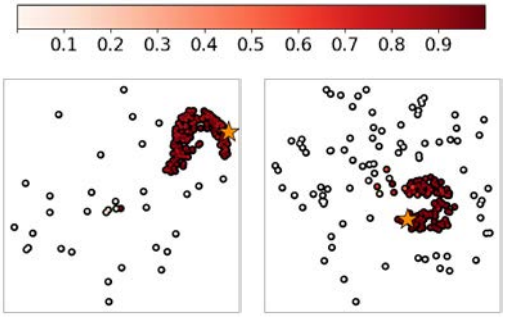
(a) CCE, $\eta = 0.2$ (b) CCE + HMW, $\eta = 0.2$ (c) CCE, $\eta = 0.5$ (d) CCE + HMW, $\eta = 0.5$

Figure 4: Feature representations using CCE and CCE+HMW under the various noise rates η of symmetric noise. Our approach can harvest more clean clusters under $\eta > 0$



(a) CCE, $\eta = 0.5$ (b) CCE + HMW, $\eta = 0.5$

Figure 5: Distributions of clean and mislabeled samples in the category of "Ship" on CIFAR-10 under the noise rate $\eta = 0.5$. CCE+HMW can effectively classify those two, but original CCE can not.



(a) HM Weights, $\eta = 0.2$ (b) HM Weights, $\eta = 0.5$

Figure 6: The distributions of our Hyperspherical Margin (HM) weights in the noisy category of "Ship" on CIFAR-10 under various noise rates η of symmetric noise. The yellow pentagram denotes the label embedding in the feature space. Nearly all clean samples have large weights, and mislabeled samples have tiny weights.

the feature representations are discussed with the help of the t-distributed Stochastic Neighbor Embedding (t-SNE) (Van der Maaten and Hinton 2008) algorithm.

More Distinguishable Representation: The feature representation of the training set in the penultimate layer has been shown in Fig. 4. As can be seen, the label noise produces a serious influence on traditional CCE. When the noise rate $\eta = 0.2$, there are some mislabeled examples mixed in the example cluster with the correct label. This rate further enlarges when $\eta = 0.5$, resulting in an obvious mess in each cluster (shown in Figs. 4(a) and 4(c)). Inversely, our approach harvests more robustness. Regardless of the tested low-noise ($\eta = 0.2$) and high-noise situation ($\eta = 0.5$), employing the HMW obtains more distinguishable feature clusters in general (shown in Figs. 4(b) and 4(d)). For more powerful illustrations, we take the noisy category "Ship" with $\eta = 0.5$ as an example to show. As shown in Fig. 5(a), following CCE training, the mislabeled examples present **obvious overlap** with clean samples. Nevertheless, as shown in Fig. 5(b), when CCE is enhanced by our HMW, nearly all clean samples gather tightly while the mislabeled samples scatter, and there is **little overlap**.

More Logical Weighting Distribution: To further discover the weighting distribution of examples in training, we also take the noisy "Ship" cluster to exhibit. The information about the HM weight of each example is shown in Fig. 6. Combined with Figs. 4 and 5, we can conclude that clean examples are gathered tightly around the label embedding of "Ship" while mislabeled samples scatter, whether $\eta = 0.2$ or $\eta = 0.5$. *More strikingly, with the help of the HMW, clean samples obtain larger weights than mislabeled samples in training. It is definitely logical and significant in learning with noisy labels*.

Performance on Benchmark Datasets

We compare our HMW with seven SOTA methods as well as the CCE loss. They are Forward (Patrini et al. 2017b), SCE (Wang et al. 2019), P-correction (Yi and Wu 2019), DivideMix (Li, Socher, and Hoi 2020b), ELR (Liu et al. 2020b), MOIT+ (Ortego et al. 2021b), and UNICON (Karim et al. 2022b). SCE (Wang et al. 2019) and UNICON (Karim et al. 2022b) are also practiced with the HMW to evaluate the generalization of our method. We reproduce them in our experimental environment for fair comparisons. The related hyperparameter settings are consistent with their literature.

Dataset	CIFAR-10							CIFAR-100						
Methods	Symmetric				Asymmetric			Symmetric				Asymmetric		
	0.2	0.5	0.8	0.9	0.1	0.3	0.4	0.2	0.5	0.8	0.9	0.1	0.3	0.4
CCE	82.7	57.9	26.1	16.8	88.8	81.7	76.0	61.8	37.3	8.8	3.5	68.1	53.3	44.5
Forward	83.1	59.4	26.2	18.8	90.4	81.9	76.7	61.4	37.3	9.0	3.4	68.7	54.4	45.3
SCE	92.0	81.7	44.1	26.6	91.5	81.1	76.3	60.8	37.0	10.2	3.9	69.6	53.8	44.5
P-correction	92.0	88.7	76.5	58.2	93.1	92.6	91.6	68.1	56.4	20.7	8.8	76.1	59.3	48.3
DivideMix	95.0	93.7	92.4	74.2	93.8	92.5	91.4	74.8	72.1	57.6	29.2	69.5	68.3	51.0
ELR	93.8	92.6	88.0	63.3	94.4	91.5	85.3	74.5	70.2	45.2	20.5	75.8	73.6	70.0
MOIT+	94.1	91.8	81.1	74.7	94.2	94.3	93.3	75.9	70.6	47.6	41.8	77.4	75.1	74.0
UNICON	94.2	95.2	93.6	89.4	93.2	94.9	93.6	76.5	75.5	63.0	40.3	77.1	76.1	70.7
CCE + HMW	**93.2**	**89.8**	**65.8**	**45.0**	**93.7**	**91.0**	**85.6**	**72.6**	**63.6**	**21.7**	**7.5**	**74.1**	**67.2**	**55.6**
SCE + HMW	**92.5**	**89.3**	**65.2**	**37.1**	**93.2**	**90.5**	**84.5**	**73.3**	**64.6**	**24.3**	**7.2**	**75.0**	**67.6**	**55.2**
UNICON + HMW	93.5	**95.2**	**93.7**	**90.7**	**93.5**	94.7	**93.7**	**76.6**	**75.8**	**63.4**	**43.4**	76.7	**76.3**	**72.1**

Table 1: Test accuracy (%) of different methods on benchmark datasets under various rates of symmetric and asymmetric noise. The reproduced methods are underlined. The better results of our methods than the original versions are in bold.

Experimental Setup: The related experimental settings are consistent with those in (Karim et al. 2022a). As for the hyperparameter settings in the HMW, when using CCE and SCE for training, we set $\alpha = 100$ and $\beta = 100$. While UNICON is applied for CIFAR-10 and the noise rate $\eta \leqslant 0.5$, we set $\alpha = 1$ and $\beta = 1$. When $\eta > 0.5$, we set $\alpha = 100$ and $\beta = 100$. While UNICON is applied for CIFAR-100 in symmetric noise experiments and the noise rate $\eta \leqslant 0.5$, we set $\alpha = 0.1$ and $\beta = 0.1$. When $\eta = 0.8$, we set $\alpha = 1$ and $\beta = 1$. When $\eta = 0.9$, we set $\alpha = 100$ and $\beta = 100$. While UNICON is applied for CIFAR-100 in the asymmetric noise experiments, we set $\alpha = 1$ and $\beta = 1$.

Results: The classification accuracy is reported in Table 1. As can be seen, enhanced models by our HMW outperform original baselines in most cases, and improvements are even larger than 25% in some situations. The largest gap under symmetric noise represents 39.7% appearing when employing CCE and CCE+HMW with $\eta = 0.8$. The efficiency of our strategy is well supported by these results. Besides, it is obvious that the effect of our method under symmetric noise is better than that under asymmetric noise.

The validity of our method is convincing. It brings an all-around enhancement. all enhanced approaches obtain higher accuracy in most cases. Besides, another finding should be further discussed. We notice that the improvements of our HMW to enhance the existing robust methods (SCE and UNICON in Table 1) seem to be less than those to enhance CCE. The possible reason is that some existing robust methods may be homological to our approach at some points. The optimal hyperparameter settings for the original method are not the best for the enhanced one. However, we still employ the recommended settings from their papers and official implementations in our experiments for fairness.

Performance on Real-World Noisy Datasets

We compare our approaches with 11 SOTA methods as well as the CCE loss. They are CED (Chen et al. 2021), SELEIE (Song, Kim, and Lee 2019), PLC (Zhang et al. 2021), InstanceGM (Garg et al. 2023), Forward (Patrini et al. 2017b), D2L (Ma et al. 2018), MentorNet (Jiang et al. 2018), Co-teaching (Han et al. 2018), Iterative-CV (Chen et al. 2019), ELR+ (Liu et al. 2020b), and ProtoMix (Li, Xiong, and Hoi 2021b). Similarly, we also employ CCE with our HMW as an example to test. Here, we attempt to briefly introduce applied real-world datasets. The ANIMAL-10N dataset encompasses 10 animal classes with intricate visual appearances. Its estimated label noise rate is around 8%. Webvision covers 1,000 categories, mirroring the ImageNet ILSVRC12 dataset. Its estimated label noise rate is around 20%, Similar to (Ma et al. 2020; Wang, Sun, and Fu 2022), we employ the first 50 categories of the Google image subset for training data and evaluate the performance on both the Webvision and ILSVRC12 validation sets.

Experimental Setup: For ANIMAL-10N, the VGG19-BN backbone is applied for training. The batch size and the epoch are set to 128 and 200, respectively. As for our HMW, we set $\alpha = 100$ and $\beta = 100$ when using CCE for training. They are set to 10 when using SCE for training. For WebVision, the Inception-ResNet backbone is utilized. The batch size and the epoch are set to 32 and 200, respectively. For our HMW, we set $\alpha = 100$, and $\beta = 100$ when using CCE for training. They are set to one when using SCE for training. All networks are trained using the Stochastic Gradient Descent (SGD) optimizer with cosine learning rate annealing. The weight decay is set to 1×10^{-3} for ANIMAL-10N and 5×10^{-4} for WebVision. The learning rate is set to 0.1 for ANIMAL-10N and 0.01 for WebVision, respectively. Additionally, Random Crop, Random Horizontal Flip, and CutMix are picked as data augmentation strategies.

Results: The classification accuracy on real-world datasets is reported in Tables 2 and 3. As can be seen, compared to the original CCE, CCE+HMW obviously obtains much greater performance. The gap between CCE and CCE+HMW on ANIMAL-10N is 7.1% (86.5%-79.4%). The gap in top-1 accuracy between CCE and CCE+HMW on ILSVRC12 is 13.00% (71.88%-58.88%). Moreover, our strategy outperforms other SOTA strategies in most cases. Except for the top-1 accuracy on ILSVRC12, CCT+HMW yields the best result in our experiments. These improvements also support the validity of our approach.

Methods	**CCE**	CED	SELEIE	PLC	InstanceGM	**CCE + HMW**
Accuracy	79.4	81.3	81.8	83.4	84.6	**86.5**

Table 2: Test accuracy (%) of different methods on ANIMAL-10N datasets. The best result is in bold.

Methods	WebVision		ILSVRC12	
	top-1	top-5	top-1	top-5
CCE	-	-	58.88	-
Forward	61.12	82.68	57.36	82.36
D2L	62.68	84.00	57.80	81.36
MentorNet	63.00	81.40	57.80	79.92
Co-teaching	63.58	85.20	61.48	84.98
Iterative-CV	65.24	85.34	61.60	84.98
ELR+	77.78	91.68	70.29	89.76
ProtoMix	76.3	91.5	**73.3**	91.2
CCE + HMW	**78.04**	**93.08**	71.88	**92.20**

Table 3: Test accuracy (%) of different methods on WebVision and ILSVRC12 datasets. The best results are in bold.

Ablation Studies

Finally, to assess the influence of hyperparameter settings on the HMW, some ablation studies are conducted. We similarly employ CCE with the HMW and evaluate the symmetric noise on CIFAR-100 as an example.

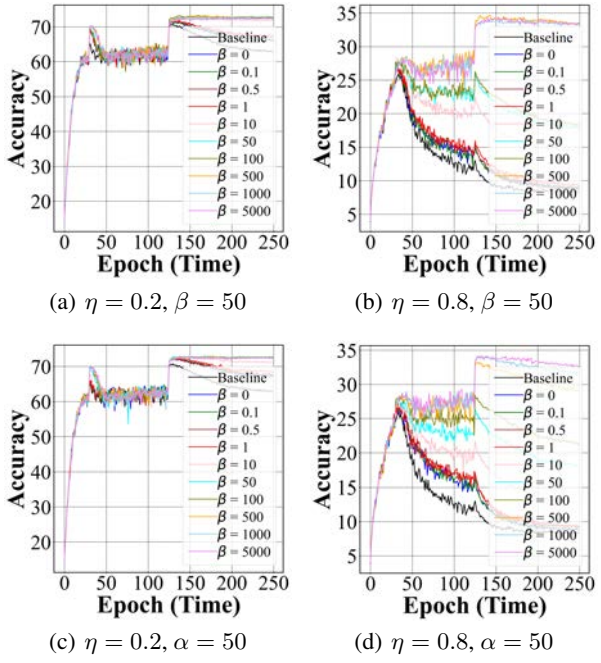
Experimental Setup: The relative settings of the experiments are the same as those used to generate Table 1. When evaluating α, we set it in $\{0, 0.1, 0.5, 1, 10, 50, 100, 500, 1000, 5000\}$. β is set to 50. When evaluating β, we set it in $\{0, 0.1, 0.5, 1, 10, 50, 100, 500, 1000, 5000\}$. α is set to 50. Additionally, we also draw the curve that is not processed by our HMW ($\alpha = 0$, $\beta = 0$) as a baseline to compare.

Results: The accuracy curves under various training settings are shown in Fig. 7. As can be seen, the baseline is limited by an apparent overfitting problem. Its curves present a steep decline in the late stages of training under all tested situations. Nevertheless, the improvement is significant after employing our HMW. In addition, we also discover that α and β seem to exhibit similar trends. The larger values they set, the stronger their abilities are to resist overfitting problems. Moreover, it is obvious that α and β should be set as larger values under $\eta = 0.8$ to limit more serious overfitting.

Furthermore, as shown in Table 4, when we only use AUM to weight examples, the performance is the worst. When adding our CA strategy, it changes to be better. When using our HMW, it is the best. It reveals that our strategy is indeed effective. Meanwhile, as shown in Figs. 7(a) and 7(b), when only using the hyperspherical TKUM ranking ($\alpha = 0$ and $\beta = 50$, the blue curve), we observe that the performance is only better than the baseline. When we use the hyperspherical AUM ranking and the hyperspherical TKUM ranking together ($\alpha > 0$ and $\beta = 50$), the curves are further elevated. A similar phenomenon also appears at the Figs. 7(c) and 7(d). It reveals that the effect of employing the hyperspherical IAM is actually better than that of employing these two sub-metrics individually. *The hyperspherical AUM ranking and the hyperspherical TKUM ranking present good cooperation to deal with noisy labels.*

Methods	Symmetric Noise		Asymmetric Noise	
	$\eta = 0.2$	$\eta = 0.5$	$\eta = 0.1$	$\eta = 0.3$
AUM Only	88.4	67.9	90.9	82.4
AUM + CA	91.7	82.1	92.5	85.1
HMW	**93.2**	**89.8**	**93.7**	**91.0**

Table 4: Test accuracy (%) of different combinations on various noise rates η on CIFAR10. The best results are in bold.



(a) $\eta = 0.2$, $\beta = 50$ (b) $\eta = 0.8$, $\beta = 50$

(c) $\eta = 0.2$, $\alpha = 50$ (d) $\eta = 0.8$, $\alpha = 50$

Figure 7: The convergence curves with different settings to the hyperparameters α and β under various noise rate η.

Conclusion

This study set out to improve the performance of learning with noisy labels. We noticed that many strategies followed the small-loss trick, which assumed the mislabeled example to generate a larger loss value than the clean example. This consideration neglected that some clean examples that were hard to learn also produced large losses. In this paper, we offered a new metric of the IAM to identify the mislabeled examples. It was outstanding at distinguishing clean examples from mislabeled examples. Owing to the IAM, we further proposed the HMW, which reweighted every training example for better performance. The HMW was universal and could also strengthen many SOTA approaches.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (62171123, 62211530112, 62071241, and 62001111) and the National Key Research and Development Program of China (2023YFC3603600). This research work is supported by the Big Data Computing Center of Southeast University.

References

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. ImageNet: A large-scale hierarchical image database. In *CVPR*, 248–255.

Northcutt, C. G.; Athalye, A.; and Mueller, J. 2021. Pervasive Label Errors in Test Sets Destabilize Machine Learning Benchmarks. *arXiv preprint arXiv:2103.14749*.

Song, H.; Kim, M.; Park, D.; Shin, Y.; and Lee, J.-G. 2022. Learning From Noisy Labels With Deep Neural Networks: A Survey. *IEEE Transactions on Neural Networks and Learning Systems*, 1–19.

Han, B.; et al. 2018. Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *NeurIPS*, 8527–8537.

Li, J.; Socher, R.; and Hoi, S. C. 2020a. DivideMix: Learning with noisy labels as semi-supervised learning. In *ICLR*, 1–14.

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

Wei, H.; Feng, L.; Chen, X.; and An, B. 2020. Combating noisy labels by agreement: A joint training method with co-regularization. In *CVPR*, 13726–13735.

Bang, J.; Koh, H.; Park, S.; Song, H.; Ha, J.-W.; and Choi, J. 2022. Online Continual Learning on a Contaminated Data Stream With Blurry Task Boundaries. In *CVPR*, 9275–9284.

Pleiss, G.; Zhang, T.; Elenberg, E.; and Weinberger, K. Q. 2020. Identifying Mislabeled Data using the Area Under the Margin Ranking. In *NeurIPS*, 17044–17056.

Chen, X.; and Gupta, A. 2015. Webly supervised learning of convolutional networks. In *ICCV*, 1431–1439.

Ortego, D.; Arazo, E.; Albert, P.; O'Connor, N. E.; and McGuinness, K. 2021a. Multi-Objective Interpolation Training for Robustness To Label Noise. In *CVPR*, 6606–6615.

Li, S.; Xia, X.; Ge, S.; and Liu, T. 2022. Selective-Supervised Contrastive Learning With Noisy Labels. In *CVPR*, 316–325.

Li, J.; Xiong, C.; and Hoi, S. 2021a. Learning From Noisy Data With Robust Representation Learning. In *ICCV*, 9485–9494.

Jenni, S.; and Favaro, P. 2018. Deep bilevel learning. In *ECCV*, 618–633.

Wei, H.; Tao, L.; Xie, R.; and An, B. 2021. Open-set label noise can improve robustness against inherent label noise. In *NeurIPS*, 1–15.

Patrini, G.; Rozza, A.; Krishna Menon, A.; Nock, R.; and Qu, L. 2017a. Making Deep Neural Networks Robust to Label Noise: A Loss Correction Approach. In *CVPR*, 1944–1952.

Zhang, Z.; and Sabuncu, M. 2018. Generalized cross entropy loss for training deep neural networks with noisy labels. In *NeurIPS*, 8778–8788.

Song, H.; Kim, M.; and Lee, J.-G. 2019. SELFIE: Refurbishing unclean samples for robust deep learning. In *ICML*, 5907–5915.

Liu, S.; Niles-Weed, J.; Razavian, N.; and Fernandez-Granda, C. 2020a. Early-Learning Regularization Prevents Memorization of Noisy Labels. In *NeurIPS*, 20331–20342.

Wen, Y.; Liu, W.; Weller, A.; Raj, B.; and Singh, R. 2022. SphereFace2: Binary Classification is All You Need for Deep Face Recognition. In *ICLR*.

Deng, J.; Guo, J.; Xue, N.; and Zafeiriou, S. 2019. ArcFace: Additive Angular Margin Loss for Deep Face Recognition. In *CVPR*.

Wang, H.; Wang, Y.; Zhou, Z.; Ji, X.; Gong, D.; Zhou, J.; Li, Z.; and Liu, W. 2018. CosFace: Large Margin Cosine Loss for Deep Face Recognition. In *CVPR*.

Ke, B.; Zhu, Y.; Li, M.; Shu, X.; Qiao, R.; and Ren, B. 2022. Hyperspherical Learning in Multi-Label Classification. In *ECCV*, 38–55.

Zhang, H.; Cisse, M.; Dauphin, Y. N.; and Lopez-Paz, D. 2018. mixup: Beyond Empirical Risk Minimization. In *ICLR*.

Krizhevsky, A.; Nair, V.; and Hinton, G. 2014. *CIFAR-10 and CIFAR-100 Datasets*.

Karim, N.; Rizve, M. N.; Rahnavard, N.; Mian, A.; and Shah, M. 2022a. UniCon: Combating Label Noise Through Uniform Selection and Contrastive Learning. In *CVPR*, 9676–9686.

Van der Maaten, L.; and Hinton, G. 2008. Visualizing data using t-SNE. *Journal of machine learning research*, 9(11).

Li, W.; Wang, L.; Li, W.; Agustsson, E.; and Gool, L. V. 2017. Web vision database: Visual learning and understanding from web data. *arXiv preprint arXiv:1802.05300*.

Wang, Y.; Ma, X.; Chen, Z.; Luo, Y.; Yi, J.; and Bailey, J. 2019. Symmetric cross entropy for robust learning with noisy labels. In *ICCV*, 322–330.

Karim, N.; Rizve, M. N.; Rahnavard, N.; Mian, A.; and Shah, M. 2022b. UniCon: Combating Label Noise Through Uniform Selection and Contrastive Learning. In *CVPR*, 9676–9686.

Ma, X.; Huang, H.; Wang, Y.; Romano, S.; Erfani, S.; and Bailey, J. 2020. Normalized loss functions for deep learning with noisy labels. In *ICML*, 6543–6553.

Wang, Y.; Sun, X.; and Fu, Y. 2022. Scalable Penalized Regression for Noise Detection in Learning With Noisy Labels. In *CVPR*, 346–355.

Patrini, G.; Rozza, A.; Krishna Menon, A.; Nock, R.; and Qu, L. 2017b. Making Deep Neural Networks Robust to Label Noise: A Loss Correction Approach. In *CVPR*, 1944–1952.

Yi, K.; and Wu, J. 2019. Probabilistic End-To-End Noise Correction for Learning With Noisy Labels. In *CVPR*.

Li, J.; Socher, R.; and Hoi, S. C. 2020b. DivideMix: Learning with Noisy Labels as Semi-supervised Learning. In *ICLR*.

Liu, S.; Niles-Weed, J.; Razavian, N.; and Fernandez-Granda, C. 2020b. Early-Learning Regularization Prevents Memorization of Noisy Labels. In *NeurIPS*, volume 33, 20331–20342.

Ortego, D.; Arazo, E.; Albert, P.; O'Connor, N. E.; and McGuinness, K. 2021b. Multi-Objective Interpolation Training for Robustness To Label Noise. In *CVPR*, 6606–6615.

Chen, Y.; Shen, X.; Hu, S. X.; and Suykens, J. A. K. 2021. Boosting Co-Teaching With Compression Regularization for Label Noise. In *CVPR*, 2688–2692.

Zhang, Y.; Zheng, S.; Wu, P.; Goswami, M.; and Chen, C. 2021. Learning with Feature-Dependent Label Noise: A Progressive Approach. In *ICLR*.

Garg, A.; Nguyen, C.; Felix, R.; Do, T.-T.; and Carneiro, G. 2023. Instance-Dependent Noisy Label Learning via Graphical Modelling. In *WACV*, 2288–2298.

Ma, X.; et al. 2018. Dimensionality-Driven Learning with Noisy Labels. In *ICML*, 3355–3364.

Jiang, L.; Zhou, Z.; Leung, T.; Li, L.-J.; and Fei-Fei, L. 2018. MentorNet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *ICML*, 2304–2313.

Chen, P.; Liao, B. B.; Chen, G.; and Zhang, S. 2019. Understanding and Utilizing Deep Neural Networks Trained with Noisy Labels. In *ICML*, 1062–1070.

Li, J.; Xiong, C.; and Hoi, S. 2021b. Learning From Noisy Data With Robust Representation Learning. In *ICCV*, 9485–9494.

Li, J.; Xiong, C.; and Hoi, S. 2021c. MoPro: Webly Supervised Learning with Momentum Prototypes. In *ICLR*.