WHICH SAMPLES SHOULD BE LEARNED FIRST: EASY OR HARD?

Anonymous authors
Paper under double-blind review

ABSTRACT

An effective weighting scheme for training samples is essential for learning tasks. Numerous weighting schemes have been proposed. Some schemes take the easy-first mode, whereas some others take the hard-first one. Naturally, an interesting yet realistic question is raised. Which samples should be learned first given a new learning task, easy or hard? To answer this question, three aspects of research are carried out. First, a high-level unified weighted loss is proposed, providing a more comprehensive view of existing schemes. Theoretical analysis is subsequently conducted. Some preliminary conclusions are obtained. The optimal weighting scheme is determined by the distribution of training samples' learning difficulties and the prior knowledge for the task. Second, a flexible weighting scheme is proposed to overcome the defects of existing schemes. The three modes, namely, easy/medium/hard-first, can be flexibly switched in the proposed scheme. Third, a wide range of experiments is conducted to further compare the weighting schemes in different modes. On the basis of these works, reasonable answers are obtained. Factors including prior knowledge and data characteristics determine which samples should be learned first in a learning task.

1 INTRODUCTION

It is widely accepted that model training is sensitive to the weights of training samples. Treating each sample unequally can improve the learning performance. The cues and inspirations for the design of the weighting function in a weighting scheme are usually derived from the following aspects:

- Application contexts of the learning task. Tasks such as fraud detection and medical diagnosis are cost-sensitive. Different samples have unequal importance according to their gains or costs. Therefore, samples with high gains/costs will be assigned high weights.
- Characteristics of the training data. Training samples are different from each other in characteristics, such as data quality, sample neighbors, and category distribution. In some tasks, some labels are of low confidence or with high noises, so these samples should be assigned low weights. In some other tasks, samples in the minority categories are usually more difficult to learn well, so these samples should be assigned high weights.

Context-inspired weighting functions are usually defined in a heuristic manner and only used in special applications, whereas characteristics-inspired weighting functions have received increasing attention in recent years due to their effectiveness and universality. Data characteristics are related to an intrinsic property of samples, namely, learning difficulty. Most related studies split training samples into easy/hard or easy/medium/hard according to samples' learning difficulties. In some schemes, hard samples are assigned high weights in what is called the hard-first mode. For example, Lin et al. (2017) proposed Focal loss in object detection, which significantly improves the detection performance. In some other schemes, easy samples have higher weights. Kumar et al. (2010) proposed self-paced learning (SPL), which sets the weights of hard samples to zero with a threshold. The threshold is gradually increased to ensure that more hard samples can participate in the training. These two priority modes, namely, easy-first and hard-first, appear to contradict each other yet both demonstrate effectiveness in certain learning tasks. Consequently, a natural question is raised. Which samples should be learned first facing a new task, easy or hard ones? To answer this question (called the "easy-or-hard" question), this study conducted both theoretical analysis and empirical verification. Reasonable answers are presented. Our contributions are summarized as follows:

Table 1: Several typical weighting schemes.

Paper	Method	Weighting scheme	Domain	Scenario	Criterion	Priority mode	Granularity
Kumar et al. (2010)	SPL_ Binary	$\min_{w\in[0,1]^n} \mathcal{L}(w,\lambda,l)=\sum_{i=1}^n w_i l-\lambda\sum_{i=1}^n w_i$	NLP CV	Noun Phrase Coreference Image classification Object Localization (Standard)	Loss	Easy-first	Sample
Jiang et al. (2014a)	SPL_Log	$\min_{w\in[0,1]^n} \mathcal{L}(w,\lambda,l)=\sum_{i=1}^n w_i l_i+\sum_{i=1}^n\left(\xi w_i-\xi^{w_i}/\log\xi\right),\xi=1-\lambda$	CV	Multimedia Event Detection (Standard)	Loss	Easy-first	Sample
Zieba et al. (2016)	Cost-sensitive SPL	$w_i=\begin{cases}1, \textit{ if } l_i<y_i C_+ +(1-y_i)C_- \\ 0, \textit{ otherwise}\end{cases}$	CV	Image classification (Imbalance)	Loss	Easy-first	Mixture
Lin et al. (2017)	Focal Loss	$\mathcal{L}(\gamma)=-(1-p)^\gamma\log(p)$	CV	Dense Object Detection (Imbalance)	Pred	Hard-first	Sample
Li et al. (2020)	QFL	$\mathcal{L}(\sigma,\beta)=\sum_{i=1}^N\left(-\lvert y_i-\sigma\rvert^\beta\left((1-y_i)\log(1-\sigma)+y_i\log(\sigma)\right)\right)$	CV	Dense Object Detection (Imbalance)	Pred	Hard-first	Sample
Ben-Baruch et al. (2020)	ASL	$l_i(\gamma_+,\gamma_-,m)=\begin{cases}(1-p_i)^{\gamma}\log(p_i), y_i=1 \\ (p_{i,m})^{\gamma}\log(1-p_{i,m}), y_i=0\end{cases}, p_{i,m}=\max(p_i-m,0)$	CV	Dense Object Detection (Imbalance)	Pred	Hard-first Discard mislabeled negative samples	Sample
Li et al. (2019)	GHM	$\mathcal{L}(\beta,l)=(1/N)\sum_{i=1}^N\beta_i l_i$	CV	Dense Object Detection (Imbalance)	Gradient	Medium-first	Sample
Freund & Schapire (1996)	AdaBoost	$w_i^{m+1}=w_i^m\exp(\alpha_m)$	CV	Handwritten Digit Recognition (Standard)	Error (Loss)	Hard-first	Sample
Zhang et al. (2021b)	G-RW	$w^c=(1/r_c)^\rho/\sum_{k=1}^C(1/r_k)^\rho$	CV	Image classification Object detection (Imbalance)	Empirical class frequency	Hard-first	Category
Bengio et al. (2009)	CL	$w_i<w_j,\forall\gamma(x_i)<\gamma(x_j)$	NLP CV	Language Modeling Shape Recognition (Standard)	Prior Knowledge	Easy-first	Sample
Zhang et al. (2021a)	GAIRAT	$w_i=(1+\tanh(\lambda+5\times(1-2\times k(x_i,y_i)/K)))/2$	CV	Image classification (Standard)	Margin	Hard-first	Sample
Cui et al. (2019)	Class-balance	$w^c=(1-\beta)/(1-\beta^{N_c}),\beta\in[0,1)$	CV	Image classification (Imbalance)	Category Proportion	Hard-first	Category
Wang et al. (2021a)	Truncated Loss	$l_i=\begin{cases}0, l_i^{CE}>\tau\wedge y_i=1 \\ l_i^{CE}, \textit{ otherwise}\end{cases}$	Data mining	Recommendation (Noise)	Loss	Easy-first Discard hard positive samples	Mixture
Shin et al. (2020)	FOCI	$w_i(q)=\textit{ Normalize }\sqrt{p_t(y_i\mid x_i)\operatorname{Var}(p_{t-q+1:t}(y_i\mid x_i))}$	CV	Image classification (Noise)	Loss and uncertainty	Medium-first	Sample
Santiagoa et al. (2021)	LOW	$R(w;\lambda)=-w^T\nabla_{\theta_t}+\lambda\lVert w-1\rVert^2$	CV	Image classification (Imbalance)	Gradient	Hard-first	Sample
Liu et al. (2021)	JTT	$\mathcal{L}(l,E)=\left(\lambda_{up}\sum_{(x_i,y_j)\in E}l_i+\sum_{(x_j,y_j)\notin E}l_j\right)$	NLP CV	Image classification Sentiment analysis (Standard)	Loss	Hard-first	Patial data
Castells et al. (2020)	SuperLoss	$\mathcal{L}(l_i,\sigma_i)=(l_i-\tau)w_i+\lambda(\log w_i)^2$	CV	Object detection, Image retrieval (Noise)	Loss	Easy-first	Sample

(1) To theoretically explore the "easy-or-hard" question, a high-level unified weighted loss is constructed. It reveals the underlying principles of how a weighting function is generated and most of the existing weighting functions can be mathematically explained with this weighted loss. Theoretical analysis is then carried out and some preliminary answers are obtained.

(2) A flexible weighting scheme is proposed based on the analysis of the defects of existing strategies with our proposed unified weighted loss. Compared with existing methods, the weighting function in our scheme can be flexibly switched among the three priority modes, namely, easy-first, medium-first, and hard-first. In contrast, existing weighting schemes can achieve only one of the three modes.

(3) Extensive experiments on image classification, graph classification, and object detection tasks are conducted on benchmark data sets. The empirical observations further support our main theoretical conclusions. In addition, our proposed weighting function achieves competitive results in all the above typical learning scenarios.

2 EXISTING WEIGHTING SCHEMES

We define the symbols including the main symbols in Table 1 as follows. Let $T=\{(x_i,y_i)\}_{i=1}^N$ be a set of N training samples, where x_i is the input feature and y_i is the associated label. Let C be the number of categories and $y_i\in\{1,\ldots,C\}$. Let r_c be the empirical class frequency of the c-th category. Let $\mathcal{L}$ is the training loss. w_i and l_i are the weight and the loss of the i-th sample, respectively. Let $p\in[0,1]$ be the predicted probability for the correct category. w^c is the weight of the c-th category when the category-wise weighting strategy is used.

The core of a weighting scheme is its weighting function for the input samples. The weighting functions can be sample-wise, category-wise, or their mixtures. According to the priority mode, the weighting functions can be easy-first, medium-first, hard-first, or their mixtures. Table 1 lists some of the typical weighting functions in previous literature. The application scenarios (i.e., standard, imbalanced, and noisy) of these functions are also presented. The hyper-parameters in most functions are nearly fixed during training, whereas they are dynamic in SPL (Kumar et al., 2010).

The weighting schemes in Table 1 can only implement one mode. Their corresponding modes are selected based on a (partial) particular view of the data characteristics. Focal loss (Lin et al., 2017) is inspired by the observation that "easy samples occupy more than hard ones in object detection data sets". SuperLoss (Castells et al., 2020) is easy-first and effective when the training data is corrupted by noise. GHM (Li et al., 2019) exerts high weights on medium samples as Focal loss is sensitive to noise. LOW (Santiagoa et al., 2021) is hard-first and works well for imbalanced data. Some studies are inspired by other cues such as the human learning mechanism. Curriculum learning (Bengio et al., 2009) is motivated by human learning that easy samples should be learned first.

Table 2: Statistics of measurement criteria

Criterion	Method	Number	Scenario
Loss (pred)	SPL_Binary (2010), SPL_Log (2014), Cost-sensitive SPL (2016), Focal Loss (2017), QFL (2020), ASL (2020), SuperLoss (2020), FOCI (2020), Truncated Loss (2021), JTT (2021)	11	Standard, Noise, Imbalance
Gradient	GHM (2019), LOW (2021)	2	Imbalance
Category proportion	Class-balance (2019), G-RW (2021)	2	Imbalance
Prior knowledge	CL (2009)	1	Standard, Noise, Imbalance
Uncertainty	FOCI (2020)	1	Noise
Margin	GAIRAT (2021)	1	Standard

Existing studies only (explicitly or implicitly) give partial answers to the "easy-or-hard" question on a specific view or scenario. Few studies have attempted to thoroughly discuss the applicable/inapplicable scenarios for a given weighting scheme. Meanwhile, several studies have proposed similar concerns. Wang et al. (2021b) raised a similar question about "easy-first versus hard-first" under the context of curriculum learning. This paper explores this question from a global perspective, obtaining reasonable findings.

3 A UNIFIED WEIGHTED LOSS AND THEORETICAL ANALYSIS

3.1 THE CRITERIA OF LEARNING DIFFICULTY MEASUREMENT

Learning difficulty is an intrinsic attribute of samples. To answer the "easy-or-hard" question, the criteria for learning difficulty measurement should be clarified first. Current criteria for learning difficulty measurement include loss, gradient, category proportion, margin, prior knowledge, and uncertainty. Table 2 statistics the application methods and scenarios of each criterion in Table 1.

Samples' learning difficulty depends on various factors which are related to the samples' characteristics, including data quality (Shin et al., 2020; Li et al., 2019), sample neighbors (Zhang et al., 2021a), and data distribution (Santiagoa et al., 2021; Cui et al., 2019). None of the above measurements can capture all these factors. From Table 2, loss-based measurements are most widely used. Two recent methods including JTT and Truncated Loss still use this criterion. In addition, loss-based measurements are more efficient than others, and it is effective in various learning tasks.

3.2 A UNIFIED WEIGHTED LOSS

Let d_i represent the i-th sample's learning difficulty, our unified weighted loss is defined in Eq. (1):

$$\mathcal{L}_{UW} = \frac{1}{N}\sum\nolimits_i w_i^* l_i, \quad s.t. \quad w^* = \arg\min_w \frac{1}{N}\sum\nolimits_i [(w_i d_i + R(w_i, \Gamma, \lambda))\mathbb{S}(d_i - \tau)], \tag{1}$$

where $R(w_i, \Gamma, \lambda)$ is the regularization term; Γ represents prior knowledge or data characteristics, τ determines the priority mode, λ is a hyper-parameter, $\mathbb{S}$ is a signum function whose value is either 1 or -1, and $w_i > 0$. $R(w_i, \Gamma, \lambda)$ must be convex on w_i. $\mathcal{L}_{UW}$ should satisfy other requirements, which are detailed in the Appendix.

The priority modes of the weighting function (w^*) obtained by Eq. (1) can be easy-first, hard-first, or medium-first when different τ values are taken. Three typical cases are discussed as follows:

(1) Easy-first ($\tau < \min\{d_i\}$): In this case, the objective function for weights becomes

$$\min_w \frac{1}{N}\sum\nolimits_{i=1}^{N} w_i d_i + R(w_i, \Gamma, \lambda), \tag{2}$$

which implies that the weights are in the easy-first mode and thus easy samples have higher weights.

(2) Hard-first ($\tau > \max\{d_i\}$): In this case, the objective function for weights becomes

$$\max_w \frac{1}{N}\sum\nolimits_{i=1}^{N} w_i d_i + R(w_i, \Gamma, \lambda), \tag{3}$$

which implies that the weights are in the hard-first mode where hard samples have higher weights.

(3) Medium-first ($\min\{d_i\} < \tau < \max\{d_i\}$): The objective function for weights becomes

$$\max_w \frac{1}{N}\sum\nolimits_{i:d_i<\tau} [w_i d_i + R(w_i, \Gamma, \lambda)] + \min_w \frac{1}{N}\sum\nolimits_{j:d_j\geq\tau} [w_j d_j + R(w_j, \Gamma, \lambda)], \tag{4}$$

which implies that the weights are in the medium-first mode.

The value of τ determines the priority mode and the specific weight of each sample is affected by $R(w_i, \Gamma, \lambda)$. In the next subsection, two typical sample weighting methods are explained and analyzed based on $\mathcal{L}_{UW}$.

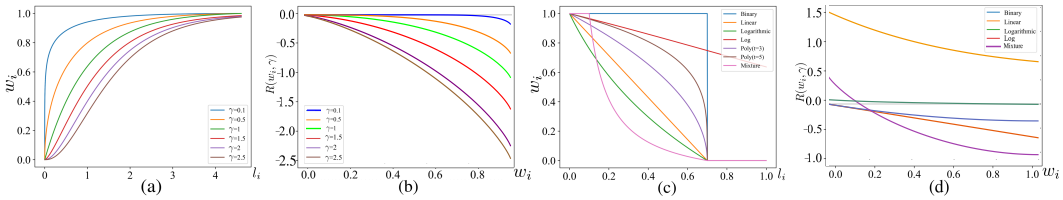

Figure 1: The weight and regularizer curves of Focal loss ((a) and (b)) and SPL ((c) and (d)).

3.3 THEORETICAL ANALYSIS OF TWO TYPICAL METHODS WITH $\mathcal{L}_{UW}$

Focal loss applies l_i to approximate d_i. According to $\mathcal{L}_{UW}$, Focal loss is obtained by defining

$$\tau > \max_i\{l_i\} \quad \text{and} \quad R(w_i,\gamma) = w_i \frac{w_i^{\frac{1}{\gamma}}{}_2F1(1,1+\gamma,2+\gamma,w_i^{\frac{1}{\gamma}})}{1+\gamma} + \log(1-w_i^{\frac{1}{\gamma}}), \tag{5}$$

where ${}_2F1$ refers to the Hypergeometric function (Seaborn, 1991), γ is the hyper-parameter, and Γ is omitted because no other prior knowledge is used. This situation belongs to the hard-first mode as shown in Fig. 1(a). The curves for $R(w_i,\gamma)$ under different values of γ are shown in Fig. 1(b). The inference of $R(w_i,\gamma)$ of Focal loss is detailed in the Appendix.

SPL-series also apply l_i to approximat d_i. Taking the SPL_Binary (Kumar et al., 2010) and SPL_Linear (Jiang et al., 2014a) as two examples, their weighting functions are obtained by defining

$$\tau < \min_i\{l_i\} \quad \text{and} \quad R(w_i,\lambda) = -\lambda\sum_{i=1}^{N} w_i \quad \text{or} \quad R(w_i,\lambda) = 0.5\lambda\sum_{i=1}^{N}(w_i^2 - 2w_i). \tag{6}$$

The primary priority mode of SPL is easy-first. The weight curves of different SPL schemes are shown in Fig. 1(c). The regularizer curves in different schemes are shown in Fig. 1(d). As previously analyzed, $R(w_i,\Gamma,\lambda)$ can be defined to prioritize some particular categories or samples (Yang et al., 2020a). Most existing weighting functions can be explained in a similar manner.

Eqs. (5) and (6) indicate that weighting schemes differ in the settings of τ and $R(w_i,\Gamma,\lambda)$. Therefore, our investigated "easy-or-hard" question can be transformed into a theoretical problem that given an arbitrary learning task, whether there is a universal optimal setting for τ and $R(w_i,\Gamma,\lambda)$.

Obviously, the answer is "No" because no fixed optimal mode can achieve the best performances on arbitrary data sets. Alternately, discussing easy-first or hard-first is futile in the absence of any prior knowledge or useful information. Indeed, the optimal solution of Eq. (1) is determined by the distribution of the training samples' learning difficulties and the prior knowledge. The learning difficulty depends on the data characteristics including data quality, sample neighbors, and data distribution. For example, the lower the quality of a sample is, the larger the learning difficulty of the sample will be; the more heterogeneous samples in the neighborhood of a sample are, the larger the learning difficulty of the sample will be.

Based on partial observations/conclusions of existing studies and the above analysis from our $\mathcal{L}_{UW}$, it is believable that an ideal weighting strategy should satisfy the following requirements:

- The weights for noisy samples should be reduced making the model less disturbed by noise. In other words, the easy-first mode will be more effective on training sets with heavy noise.
- If easy samples are excessive, the hard-first mode is preferred like the application of Focal loss in object detection. Likewise, it is natural to deduce that if hard samples are excessive, the easy-first mode should be applied.
- Reliable prior knowledge for the learning task and the useful information on training data characteristics should be integrated into the regularizer. For example, some categories (e.g., tail categories) should be given more attention.

The conclusion that no universal optimal setting exists and the above three requirements constitute the preliminary answer to the "easy-or-hard" question. Unfortunately, none of the existing weighting schemes can satisfy all three requirements. Section 4 introduces a new weighting scheme.

3.4 MORE ANALYSIS BASED ON THE BIAS-VARIANCE TRADE-OFF

The bias-variance trade-off theory is used to further support the second requirement introduced in the previous subsection. Let T be a random training set and $f(x|T)$ be the trained model on T. The bias-variance trade-off is based on the following learning error (Yang et al., 2020b):

$$Err = E_{x,y}E_T\left[\|y - f(x|T)\|_2^2\right] = Bias^2 + Variance + \delta_e \approx BiasT + VarT. \tag{7}$$

The bias-variance trade-off theory indicates that the bias and variance terms will respectively decrease and increase if the model complexity c increases (Domingos, 2000). Minimum learning error is achieved when the sum of the partial derivatives of two terms with respect to the model complexity c is equal to zero (Fortmann-Roe, 2012). In this study, training samples are divided into easy, medium, and hard according to their learning difficulties. Therefore, we divide the sample space into three corresponding regions, namely, R_{easy}, R_{medium}, and R_{hard}. Similar to Eq. (7), we define:

$$Err_{easy} = E_{(x,y)\in R_{easy}} E_T \left[\|y - f(x|T)\|_2^2 \right] \approx BiasT_{easy} + VarT_{easy}. \tag{8}$$

Likewise, we can define the bias/variance terms for the R_{medium} and R_{hard} regions. Based on the bias-variance trade-off theory on the entire sample space, we propose the following assumption:

Assumption 1: For all the three bias (e.g., $BiasT_{easy}$) and variance terms (e.g., $VarT_{easy}$) of R_{easy}, R_{medium}, and R_{hard}, the bias and variance terms are decreasing and increasing functions of the model complexity c, respectively. Both the partial derivatives of the bias and variance terms with respect to c are increasing functions, respectively.

According to Assumption 1, minimum learning error for each region is achieved when the sum of the partial derivatives of its bias and variance term with respect to c equals to zero.

Let c^* be the optimal model complexity for the entire sample space when the minimum of Err in Eq. (7) is attained. Likewise, let c^*_{easy} and c^*_{hard} be the optimal model complexities for R_{easy} and R_{hard}, respectively. The following assumption is proposed:

Assumption 2: $c^*_{easy} < c^* < c^*_{hard}$.

With Assumption 2, we have the following proposition.

Proposition 1: If weights higher than one are exerted on the samples in R_{hard}, and the weights in the other regions remain one, then the new optimal model complexity c^*_{new} over the entire space will be larger than c^*. Alternatively, the complexity of the optimal model is increased.

A theoretical analysis for Proposition 1 is shown in the Appendix. Proposition 1 supports the second requirement. When easy samples are excessive in a training set, the model will become quite simple and under-fitting. Thus, hard samples should be assigned high weights to increase the complexity of the final model. The analysis for the second requirement when hard samples are excessive is presented in the Appendix.

4 A NEW WEIGHTING SCHEME

Inspired by $\mathcal{L}_{UW}$, we propose a flexible weighting scheme which can achieve all three priority modes. Its weighting function is shown in Eq. (9).

$$w_i = (d_i + \alpha)^\gamma e^{-\gamma(d_i+\alpha)}, \tag{9}$$

where γ (the shape parameter) and α (the translation parameter) are two hyper-parameters. Inspired by Focal loss, d_i is approximated by $1 - p_i$. Thus, the weighting function becomes:

$$w_i = (1 - p_i + \alpha)^\gamma e^{-\gamma(1-p_i+\alpha)}. \tag{10}$$

In the easy-first ($\tau < \min(l_i)$) and hard-first ($\tau > \max(l_i)$) modes, Eq. (10) can be obtained by solving Eq. (1) with the following regularizer:

$$R(w_i, \gamma, \alpha) = \int -\log \frac{1}{1 + \alpha + W(-(w_i)^{\frac{1}{\gamma}})} dw_i, \tag{11}$$

where W represents the Lambert W function (Corless et al., 1996). The complete derivation process and curve examples of $R(w_i, \gamma, \alpha)$ can be seen in the Appendix.

When different values of γ and α are chosen, different priority modes can be produced by FlexW. Fig. 2 shows the exemplar weighting curves in different modes including "easy-first" (Fig. 2(a)), "medium-first" (Fig. 2(b)) and "hard-first" (Fig. 2(c)). Therefore, we only need to change the values of γ and α of FlexW instead of the entire weighting scheme when facing different learning tasks. Our experiments show that FlexW is effective in different scenarios including noise, long-tail, and their mixtures. The appendix contains more examples of weight curves.

The dynamic weighting manner of SPL can also be incorporated into FlexW. Adding a scale parameter c_{y_i} can further improve the performance in some cases. The new weighting function is:

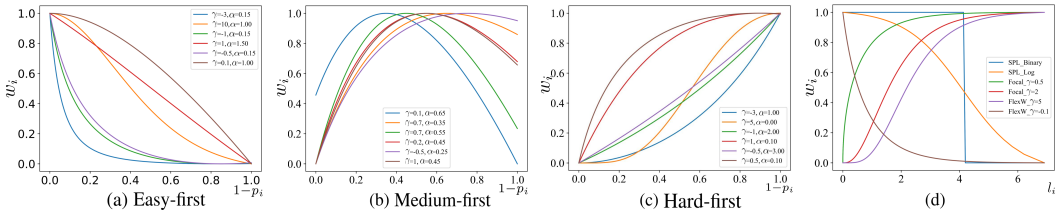

Figure 2: (a-c) show the weight curves of FlexW under three priority modes. (d) shows the weight curves of SPL, Focal loss, and FlexW.

$$w_i = \begin{cases} c_{y_i}(1 - p_i + \alpha)^{\gamma} e^{-\gamma(1-p_i+\alpha)}, & l_i \leq \lambda \\ 0, & l_i > \lambda \end{cases} . \tag{12}$$

When γ is set to 0, the binary scheme of SPL can be realized as shown in Fig. 2(d). To better understand FlexW, its loss gradient is analyzed and details are presented in the Appendix.

5 Experimental Results

To answer the question from the perspective of empirical verification, we conduct extensive experiments for various tasks under different scenarios.

5.1 Image Classification with Noisy Labels

Two benchmark data sets, namely, CIFAR10 and CIFAR100 (Krizhevsky, 2009), are used. Flip and uniform label noises are simulated following the manners in (Shu et al., 2019). Wide ResNet-28-10 (WRN-28-10) (Zagoruyko & Komodakis, 2016) and ResNet-32 (He et al., 2016) are adopted for the flip and uniform noises, respectively. Each experimental run is repeated five times with different seeds for parameter initialization and label noise generation. The introduction for compared methods and other details are presented in the Appendix. Due to lack of space, only the results under flip noise are presented and analyzed here. The results of uniform noise are presented in the Appendix.

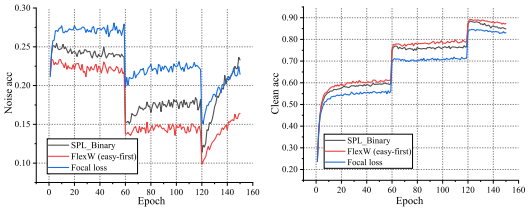

Figure 3: Accuracies of the three methods on noisy (left) and clean (right) samples under 40% flip noise.

To analyze the performances of the hard-first and easy-first modes on the noise data, the specific accuracies of SPL_Binary (easy-first), Focal loss (hard-first), and FlexW (easy-first) on noisy and clean samples are analyzed which is shown in Fig. 3. The schemes with the easy-first mode (including SPL_Binary have lower accuracies on noisy samples (errors) before 140 epochs as shown in the left figure. Thus, the easy-first methods are less affected by noises. From the right figure, SPL_Binary and FlexW consistently outperform Focal loss on clean samples. In addition, medium-first also achieves good results as presented in the Appendix. Therefore, the easy/medium-first modes are more suitable than hard-first ones on noisy data. Under different noise rates, we compare various advanced methods, as shown in Table 3. The best two methods are SPL_log and FlexW(easy-first) which are both under the easy-first mode. In some cases, the performances of Focal loss and FlexW (hard-first) can approach or even exceed SPL_binary. The reason is that using only loss to distinguish noise samples from hard ones is not completely accurate. However, under the same method, the performances of the easy-first mode always exceed that of the hard-first.

Table 3: Accuracies (%) under flip noises. The best and the second best results are bold and underlined, respectively.

Data set	Noise	Baseline	Reed Hard	SPL_Binary	SPL_Log	Focal loss	S-model	Co-teaching	D2L	Fine-tuning	MentorNet	FlexW (hard-first)	FlexW (easy-first)
CIFAR10	20%	76.83±2.30	88.28±0.36	87.03±0.34	<u>89.50±0.48</u>	86.45±0.19	79.25±0.30	82.83±0.85	87.66±0.40	82.47±3.64	86.36±0.31	88.50±0.85	**90.96±0.12**
	40%	70.77±2.31	81.06±0.76	81.63±0.52	<u>84.01±0.51</u>	80.45±0.97	75.73±0.32	75.41±0.21	83.89±0.46	74.07±1.56	81.76±0.28	83.28±0.45	**85.64±0.11**
CIFAR100	20%	50.86±0.27	60.27±0.76	63.63±0.30	<u>63.82±0.27</u>	61.87±0.30	45.45±0.25	54.13±0.55	63.48±0.53	56.98±0.50	61.97±0.47	62.65±0.75	**65.48±0.82**
	40%	43.01±1.16	50.40±1.01	53.51±0.53	53.20±0.11	<u>54.13±0.40</u>	43.81±0.15	44.85±0.81	51.83±0.33	46.37±0.25	52.66±0.56	52.78±0.44	**55.50±0.25**

5.2 Image Classification with Imbalanced Data Sets

In this experiment, long-tailed versions of CIFAR benchmarks with different imbalance factors as defined by Cui et al. (2019) are used. ResNet-32 (He et al., 2016) is used as the basic model. The introduction for compared methods and other details are presented in the Appendix.

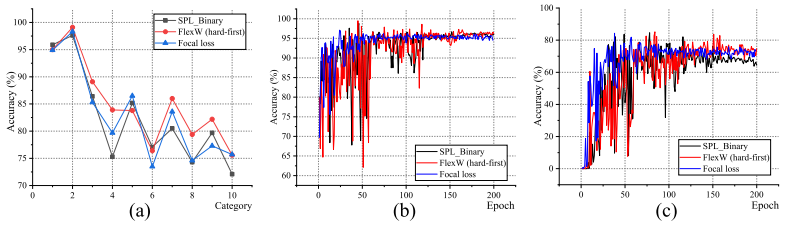

Figure 4: (a) shows the accuracies of ten categories in their respective optimal epochs; (b) and (c) show the accuracies of Categories 1 and 10, respectively. More results are presented in the Appendix.

Table 4: Test accuracies (%) on imbalanced CIFAR10 and CIFAR100 with different imbalance factors ("-" means there is no record of the results in the original paper.)

Data set	Long-tailed CIFAR10					Long-tailed CIFAR100				
Imbalance factor	200	100	50	20	10	200	100	50	20	10
CE (Baseline)	65.68	70.36	74.81	82.23	86.39	34.84	38.32	43.85	51.14	55.71
Focal loss_γ=1	65.29	70.38	76.71	82.76	86.66	35.62	38.41	44.32	51.95	55.78
Focal loss_γ=0.5	64.00	70.33	76.72	82.89	86.81	35.00	38.69	44.12	51.10	55.70
SPL_Binary	65.64	70.94	76.82	82.41	87.09	35.56	38.16	42.77	50.91	56.70
SPL_Log	62.05	70.46	75.64	82.66	86.62	33.08	38.51	41.71	49.71	54.79
L2RW	66.25	72.23	76.45	81.35	82.12	33.00	38.90	43.17	50.75	52.12
Class-balance CE loss	68.77	72.68	78.13	84.56	87.90	35.56	38.77	44.79	51.94	57.57
Class-balance Fine-tuning	66.24	71.34	77.44	83.22	83.17	**38.66**	41.50	46.12	52.30	57.57
Class-balance Focal loss	68.15	74.57	79.22	83.78	87.48	36.23	39.60	45.21	52.59	57.99
Equalised	-	73.98	-	-	-	-	**42.74**	-	-	-
Mixup	-	73.06	77.82	-	87.10	-	39.54	44.99	-	58.02
Meta-weight net	67.20	73.57	79.10	84.45	87.55	36.62	41.61	45.66	53.04	58.91
LDAM	66.75	73.55	78.83	83.89	87.32	36.53	40.60	46.16	51.59	57.29
FlexW (easy-first)	66.20	73.79	79.11	84.51	88.07	37.21	39.23	44.80	52.11	57.73
FlexW (hard-first)	**69.40**	**75.33**	**80.05**	**85.46**	**88.50**	37.54	41.69	**47.18**	**53.10**	**58.98**

To study the performances of the hard-first and easy-first modes on each category, the accuracy for each category is analyzed where the imbalance factor equals to 20. Fig. 4(a) indicates that methods under hard-first mode (i.e., FlexW(hard-first) and Focal loss) increase the accuracies of most tail categories compared with those under easy-first mode (i.e., SPL_Binary). Fig. 4(c) shows that the methods of hard-first mode significantly improve the accuracy of the last tail category. Table 4 compares the performances of some advanced methods under different imbalance factors. Two typical hard-first methods (i.e., FlexW (hard-first) and Class-balance) perform well. Furthermore, the performances of FlexW are ranking first or second in all cases. In some cases, the performances of SPL are approaching Focal loss which is because easy-first methods can improve the accuracies of the head categories. However, these methods further enlarge the gap between head and tail categories.

Figs. 5(a) and (b) show the average weights of samples in the five head (a) and tail (b) categories, reflecting the contribution of samples in each category to the model. The weights of the head categories drop quickly, whereas those of the tail categories remain high during the entire training process. It indicates hard-first mode increases the influence of the tail categories on the model.

Fig. 5(c) shows the proportion of hard samples (with $l_i \geq \log 10$) in each category. Tail categories have larger proportions of hard samples than head ones, which supports the common sense that samples in the tail categories are harder to learn than those in the head on average. The confusion matrices for CIFAR10 under varying imbalance factors are shown in the Appendix.

5.3 Node classification for graph data sets

Five benchmark graph data sets are used, namely, Cora, Citeseer, Pubmed, Coauthor CS, and Coauthor Physics (Yang et al., 2016; Shchur et al., 2018). The basic model in this experiment is an eight-layer GCN (Bruna et al., 2014). In GCN, the heterogeneous nodes around a node negatively affect the representation of that node. To study the effect of the hard-first and easy-first modes on graph data sets, FlexW (both easy/hard-first) is compared with several variations of SPL and Focal loss. More experimental information is shown in the Appendix.

Table 5 shows the results of the competing methods. In general, the easy-first schemes (i.e., SPL and FlexW (easy-first)), are better than the hard-first ones (i.e., Focal loss and FlexW (hard-first)). As the hard samples in a graph are mostly those with a large proportion of heterogeneous adjacent nodes, easy-first schemes can reduce the negative influence of the information exchange among heterogeneous nodes. To investigate it, the over-smoothing degree is measured by computing the

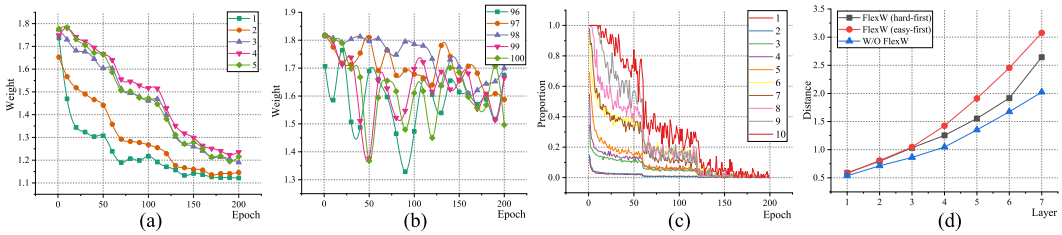

Figure 5: (a) and (b) show the average weights of the first/last five head/tail categories on CIFAR100. (c) shows the proportion of hard samples contained in each category on CIFAR10. (d) shows the effect of relieving over-smoothing with different priority modes on the Cora data set.

Table 5: Accuracies (%) of the competing methods on five graph data sets.

Method\Data set	Cora	Citeseer	Pubmed	Coauthor CS	Coauthor Physics
Original	86.50	78.70	90.90	90.70	94.00
SPL_Poly	87.10	78.30	90.40	92.07	95.78
SPL_Log	87.10	78.30	90.20	93.44	95.65
SPL_Binary	86.50	78.90	89.90	93.16	94.48
Focal loss	86.10	78.70	89.70	89.43	93.03
FlexW (hard-first)	86.60	78.10	90.00	89.34	93.90
FlexW (easy-first)	**87.50**	**79.50**	**91.30**	**93.71**	**95.85**

Euclidean distance between the output of the current layer and that of the previous one (Rong et al., 2020). The smaller the distance is, the more serious the over-smoothing is (Chen et al., 2020). Fig. 5(d) indicates that the distances of the model with FlexW (easy-first) are larger than those of the baseline and hard-first. Thus, the easy-first mode can relieve the over-smoothing phenomenon.

5.4 Dense object detection

Dense object detection is a typical application where the distribution of hard and easy samples is imbalanced. PASCAL VOC (Mark et al., 2010; 2015) is used in this experiment. The original VOC data has excessive easy samples, which is abbreviated as VOC-e. To investigate other data distribution cases, we compiled two training data sets based on VOC: data set with excessive hard samples (denoted by VOC-h) and data set with 8,000 medium hard samples (denoted by VOC-m). The YOLOv4 (Bochkovskiy et al., 2020) model, which utilizes Focal loss (FL) to calculate the loss of confidence, is used. More details of these experiments are shown in the Appendix.

In this experiment, we reveal an interesting fact that Focal loss can also implement the easy-first mode when its hyper-parameter γ is negative. In Table 6, we discuss the four weighting schemes (FL (easy-first), FL (hard-first), FlexW (easy-first), FlexW (hard-first)) for the three data sets.

The two hard-first schemes obtain better results on VOC-e which contains excessive easy samples. In contrast, when the data set has excessive hard samples, the easy-first methods get better results, and the same is true for VOC-m. FlexW achieves the highest accuracies in all three cases.

5.5 Standard CIFAR data sets

The standard CIFAR10 and CIFAR100 data sets are experimented. Detailed information on this experiment is shown in the Appendix. Focal loss (hard-first) (Lin et al., 2017), SPL (easy-first) (Kumar et al., 2010), importance sampling (Katharopoulos & Fleuret, 2018), MentorNet (Jiang et al., 2018), LOW (hard-first) (Santiagoa et al., 2021), and FlexW (easy-first and hard-first) are compared on the basic network WRN-28-2 (Zagoruyko & Komodakis, 2016) as shown in Table 7.

From Table 7 and results in the Appendix, a clear judgement between the easy-first and hard-first modes on standard data can not be obtained. In practice, FlexW can achieve three priority modes by adjusting its parameters, exhibiting flexibility when facing a new data set. More results in the Appendix suggest that FlexW obtains better results than other compared methods on standard data.

5.6 More experimental analysis for FlexW

More analyses are conducted for FlexW. In the first scenario, different levels of prior knowledge are combined into FlexW. In the second scenario, varied priority modes during the training process are experimented. In the third scenario, data sets that contain both imbalance and noise are studied. The conclusion is that when both imbalance and noise exist in a data set, a strategy with a more serious deviation should be selected. The details are presented in the Appendix.

Table 6: mAPs (%) of the four learning schemes on the three VOC data sets.

Scheme	FL (hard-first)	FL (easy-first)	FlexW (hard-first)	FlexW (easy-first)
VOC-e	75.21	66.96	**76.84**	71.70
VOC-h	66.62	68.30	67.67	**69.25**
VOC-m	55.74	62.36	60.14	**62.71**

Table 7: Accuracies (%) of different methods on CIFAR10 and CIFAR100.

Method	Baseline	Focal loss	SPL	IS	MentorNet	LOW	FlexW (easy-first)	FlexW (hard-first)
CIFAR10	92.80	92.40	92.30	92.10	91.50	93.20	92.68	**94.15**
CIFAR100	72.00	71.40	71.80	68.00	70.90	72.30	**72.72**	70.22

5.7 The selection of hyper-parameters in FlexW

During the experiments, we find that for easy-first and hard-first modes, parameters $\{\gamma = -0.5, \alpha = 0.15\}$ and $\{\gamma = 0.5, \alpha = 0.15\}$ can achieve good results in most cases. Under the medium-first mode, the preference for easy or hard samples still exists. When $\alpha = 0.58$ and γ is chosen arbitrarily, the preference for easy and hard samples is approximately equal. Also, the value of γ can be set to 0.5. More detailed parameter value ranges for stable performances in different modes are shown in Section B.5. Grid search can be used to select parameters within given intervals.

6 Answers and discussion

According to the aforementioned theoretical analysis and empirical observations, a comprehensive answer is obtained for our investigated "easy-or-hard" question:

- No universal fixed optimal priority mode exists for an arbitrary learning task.
- The priority mode depends heavily on the distribution character for the easy, medium, and hard samples. If easy/hard samples are excessive in the training set, hard/easy-first mode is the primary choice when no prior knowledge exists.
- For long-tail data, hard-first is preferred; for noisy data, easy/medium-first are more appropriate; for graph data, easy-first can alleviate over-smoothing. For other scenarios, FlexW can implement different modes and the best one can be selected based on validation data.
- The priority mode need not remain unchanged during training. Using varied priority modes in different training stages may achieve better results.

The above answer indicates that the inference of the learning difficulty of samples is crucial. In most existing studies (including ours), the learning difficulty is approximated by the loss (or the predicted probability) (Lin et al., 2017; Kumar et al., 2010; Ben-Baruch et al., 2020). Nevertheless, an ideal solution should fully consider factors such as loss, the sample's neighborhood, category distribution, and noise level. This study will be the focus of our future work.

Another important issue is the judgement of whether easy/hard samples are excessive in the training set. The excess of easy/hard samples can be judged according to the difference between the distributions of the entire sample space and the training data set. However, it is impractical to utilize the distribution of the entire sample space. A feasible way is to take the distribution of the validation set as a reference. The detailed analysis is shown in Appendix.

7 Conclusions

This study focused on an interesting and important question about the choices of priority modes on easy, medium, and hard samples for learning tasks. A deep investigation for this question facilitates the understanding of various existing weighting schemes and the choice of an appropriate scheme for a new learning task. First, a unified weighted loss is proposed which can mathematically explain most existing weighting functions. This unified loss provides a comprehensive view to theoretically analyze the "easy-or-hard" question. The defects of existing weighting functions can be clearly summarized by this loss. Second, a flexible weighting scheme is proposed inspired by the unified weighted loss and the defects of existing weighting schemes. Third, extensive experiments in image classification, graph classification, and object detection are conducted under different data characteristics including standard, noisy, long-tail, and different difficulty distributions. A comprehensive answer is obtained according to the theoretical analysis and the empirical verification. In addition, our proposed scheme FlexW achieves competitive results under nearly all the experimental tasks.

REFERENCES

Emanuel Ben-Baruch, Tal Ridnik, Nadav Zamir, Asaf Noy, Itamar Friedman, Matan Protter, and Lihi Zelnik-Manor. Asymmetric loss for multi-label classification. *arXiv preprint arXiv:2009.14119*, 2020.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th International Conference on Machine Learning*, pp. 41–48, 2009.

Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934*, 2020.

Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and deep locally connected networks on graphs. In *2nd International Conference on Learning Representations*, pp. 1–14, 2014.

Kaidi Cao, Colin Wei, Adrien Gaidon, Nikos Arechiga, and Tengyu Ma. Learning imbalanced datasets with label-distribution-aware margin loss. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pp. 1567–1578, 2019.

Thibault Castells, Philippe Weinzaepfel, and Jerome Revaud. Superloss: A generic loss for robust curriculum learning. In *34th Conference on Neural Information Processing Systems*, pp. 1–12, 2020.

Deli Chen, Yankai Lin, Wei Li, Peng Li, Jie Zhou, and Xu Sun. Measuring and relieving the over-smoothing problem for graph neural networks from the topological view. In *34th AAAI Conference on Artificial Intelligence*, pp. 3438–3445, 2020.

Jie Chen, TengfeiMa, and Cao Xiao. Fastgcn: Fast learning with graph convolu-tional networks via importance sampling. In *6th International Conference on Learning Representations*, pp. 1–15, 2018.

Hao Cheng, Dongze Lian, Bowen Deng, Shenghua Gao, Tao Tan, and Yanlin Geng. Local to global learning: Gradually adding classes for training deep neural networks. In *32nd IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4743–4751, 2019.

R. M. Corless, G. H. Gonnet, D. E. G. Hare, D. J. Jeffrey, and D. E. Knuth. On the lambertw function. *Advances in Computational Mathematics*, 5(1):329–359, 1996.

Yin Cui, Yang Song, Chen Sun, Andrew Howard, and Serge Belongie. Large scale fine-grained categorization and domain-specific transfer learning. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4109–4118, 2018.

Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. Class-balanced loss based on effective number of samples. In *2019 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 9260–9269, 2019.

Pedro Domingos. A unified bias-variance decomposition for zero-one and squared loss. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence*, pp. 564–569, 2000.

Yang Fan, Fei Tian, Tao Qin, Xiang-Yang Li, and Tie-Yan Liu. Learning to teach. In *6th International Conference on Learning Representations*, pp. 1–16, 2018.

Scott Fortmann-Roe. Understanding the bias-variance tradeoff. `http://scott.fortmann-roe.com/docs/BiasVariance.html`, 2012.

Yoav Freund and Robert E. Schapire. Experiments with a new boosting algorithm. In *Machine Learning: Proceedings of the Thirteenth International Conference*, pp. 1–9, 1996.

Jacob Goldberger and Ehud Ben-Reuven. Training deep neural-networks using a noise adaptation layer. In *5th International Conference on Learning Representations*, pp. 1–9, 2017.

Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor W. Tsang, and Masashi Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *32nd Conference on Neural Information Processing Systems*, pp. 8536–8546, 2018.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, 2016.

Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2261–2269, 2017.

Wenbing Huang, Tong Zhang, Yu Rong, and Junzhou Huang. Adaptive sampling towards fast graph representation learning. In *32nd Conference on Neural Information Processing Systems*, pp. 4558–4567, 2018.

Muhammad Abdullah Jamal, Matthew Brown, Ming-Hsuan Yang, Liqiang Wang, and Boqing Gong. Rethinking class-balanced methods for long-tailed visual recognition from a domain adaptation perspective. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7607–7616, 2020.

Lu Jiang, Deyu Meng, Teruko Mitamural, and Alexander G. Hauptmann. Easy samples first: Self-paced reranking for zero-example multimedia search. In *2014 ACM Conference on Multimedia*, pp. 547–556, 2014a.

Lu Jiang, Deyu Meng, Shoou-I Yu, Zhenzhong Lan, Shiguang Shan, and Alexander G. Hauptmann. Self-paced learning with diversity. In *28th Annual Conference on Neural Information Processing Systems*, pp. 2078–2086, 2014b.

Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fe. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 3601–3620, 2018.

Maya Kabkab, Azadeh Alavi Wang, and Rama Chellappa. Dcnns on a diet: Sampling strategies for reducing the training set size. *arXiv preprint arXiv:1606.04232*, 2016.

Simonyan Karen and Zisserman Andrew. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Shyamgopal Karthik, Jérome Revaud, and Boris Chidlovskii. Learning from long-tailed data with noisy labels. *arXiv preprint arXiv:2108.11096*, 2021.

Angelos Katharopoulos and François Fleuret. Not all samples are created equal: Deep learning with importance sampling. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 1–13, 2018.

Salman H. Khan, Munawar Hayat, Mohammed Bennamoun, Ferdous Sohel, and Roberto Togneri. Cost-sensitive learning of deep feature representations from imbalanced data. *IEEE Transactions on Neural Networks and Learning Systems*, 29(8):3573–3587, 2018.

Diederik P. Kingma and Jimmy Lei Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations*, pp. 1–15, 2015.

Alex Krizhevsky. *Learning multiple layers of features from tiny images*. MIT Press, 2009.

M. Pawan Kumar, Ben Packer, and Daphne Koller. Self-paced learning for latent variable models. In *24th Annual Conference on Neural Information Processing Systems*, pp. 1–9, 2010.

Buyu Li, Yu Liu, and Xiaogang Wang. Gradient harmonized single-stage detector. In *33rd AAAI Conference on Artificial Intelligence*, pp. 8577–8584, 2019.

Shuang Li, Kaixiong Gong, Chi Harold Liu, Yulin Wang, Feng Qiao, and Xinjing Cheng. Metasaug: Meta semantic augmentation for long-tailed visual recognition. *arXiv preprint arXiv:2103.12579*, 2021.

Xiang Li, Wenhai Wang, Lijun Wu, Shuo Chen, Xiaolin Hu, Jun Li, Jinhui Tang, and Jian Yang. Generalized focal loss: Learning qualified and distributed bounding boxes for dense object detection. *arXiv preprint arXiv:2006.04388*, 2020.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollar. Focal loss for dense object detection. In *2017 IEEE International Conference on Computer Vision*, pp. 2999–3007, 2017.

Evan Zheran Liu, Behzad Haghgoo, Annie S. Chen, Aditi Raghunathan, Pang Wei Koh, Shiori Sagawa, Percy Liang, and Chelsea Finn. Just train twice: Improving group robustness without training group information. *arXiv preprint arXiv:2107.09044*, 2021.

Ilya Loshchilov and Frank Hutter. Online batch selection for faster training of neural net- works. In *Proceedings of the 33th International Conference on Machine Learning*, pp. 1–20, 2016.

Xingjun Ma, Yisen Wang, Michael E. Houle, Shuo Zhou, Sarah Erfani, Shutao Xia, Sudanthi Wijewickrema, and James Bailey. Dimensionality-driven learning with noisy labels. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 3361–3370, 2018.

Everingham Mark, Gool Luc, Williams Christopher K. I., Winn John, and Zisserman Andrew. The pascal visual object classes(voc)challenge. *International Journal of Computer Vision*, 88(2):303–338, 2010.

Everingham Mark, Eslami S. M. AliVan, Gool Luc, Williams Christopher K. I., Winn John, and Zisserman Andrew. The pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision*, 111(1):98–136, 2015.

Deanna Needell, Nathan Srebro, and Rachel Ward. Stochastic gradient descent, weighted sampling, and the randomized kaczmarz algorithm. In *28th Annual Conference on Neural Information Processing Systems 2014*, pp. 1017–1025, 2014.

Scott Reed, Honglak Lee, Dragomir Anguelov, Christian Szegedy, Dumitru Erhana, and Andrew Rabinovich. Training deep neural networks on noisy labels with bootstrapping. In *3rd International Conference on Learning Representations*, pp. 1–11, 2015.

Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. Learning to reweight examples for robust deep learning. In *6th International Conference on Learning Representations*, pp. 1–13, 2018a.

Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. Learning to reweight examples for robust deep learning. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 6900–6909, 2018b.

Yu Rong, Wenbing Huang hwenbing, Tingyang Xu, and Junzhou Huang. Dropedge: Towards deep graph convolutional networks on node classification. In *8th International Conference on Learning Representations*, pp. 1–17, 2020.

Carlos Santiagoa, Catarina Barataa, Michele Sasdellib, Gustavo Carneirob, and Jacinto C.Nascimento. Low: Training deep neural networks by learning optimal sample weights. *Pattern Recognition*, 110(1):1–12, 2021.

James B. Seaborn. *Hypergeometric functions and their applications*. Springer-Verlag, 1991.

Oleksandr Shchur, Maximilian Mumme, Aleksandar Bojchevski, and Stephan Günnemann. Pitfalls of graph neural network evaluation. In *32nd Conference on Neural Information Processing Systems*, pp. 1–11, 2018.

Wonyoung Shin, Shengzhe Li Jung-Woo Ha, Yongwoo Cho, Hoyean Song, and Sunyoung Kwon. Which strategies matter for noisy label classification? insight into loss and uncertainty. *arXiv preprint arXiv:2008.06218*, 2020.

Jun Shu, Qi Xie, Lixuan Yi, Qian Zhao, Sanping Zhou, Zongben Xu, and Deyu Meng. Meta-weight-net: Learning an explicit mapping for sample weighting. In *33rd Annual Conference on Neural Information Processing Systems*, pp. 1–23, 2019.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–15, 2015.

Jingru Tan, Changbao Wang, Buyu Li, Quanquan Li, Wanli Ouyang, Changqing Yin, and Junjie Yan. Equalization loss for long-tailed object recognition. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11659–11668, 2020.

Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(2605):2579–2605, 2008.

Wenjie Wang, Fuli Feng, Xiangnan He, Liqiang Nie, and Tat-Seng Chua. Denoising implicit feedback for recommendation. In *Proceedings of the Fourteenth ACM International Conference on Web Search and Data Mining*, pp. 373–381, 2021a.

Xin Wang, Yudong Chen, and Wenwu Zhu. A survey on curriculum learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1(1):33788677, 2021b.

Xiaoxia Wu, Ethan Dyer, and Behnam Neyshabur. When do curricula work? In *9th International Conference on Learning Representations*, pp. 1–23, 2021.

Jufeng Yang, Xiaoping Wu, Jie Liang, Xiaoxiao Sun, Ming-Ming Cheng, Paul L. Rosin, , and Liang Wang. Self-paced balance learning for clinical skin disease recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(8):2832–2846, 2020a.

Zhilin Yang, William W. Cohen, and Ruslan Salakhutdinov. Revisiting semi-supervised learning with graph embeddings. In *Proceedings of the 33th International Conference on Machine Learning*, pp. 40–48, 2016.

Zitong Yang, Yaodong Yu, Chong You, Jacob Steinhardt, and Yi Ma. Rethinking bias-variance trade-off for generalization of neural networks. In *Proceedings of the 37the International Conference on Machine Learning*, pp. 10698–10708, 2020b.

Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *Proceedings of the British Machine Vision Conference*, pp. 87.1–87.12, 2016.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *5th International Conference on Learning Representations*, pp. 1–15, 2017.

Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *6th International Conference on Learning Representations*, pp. 1–13, 2018.

Jingfeng Zhang, Jianing Zhu, Gang Niu, Bo Han, Masashi Sugiyama, and Mohan Kankanhalli. Geometry-aware instance-reweighted adversarial training. In *9th International Conference on Learning Representations*, pp. 1–29, Online, 2021a.

Songyang Zhang, Zeming Li, Shipeng Yan, Xuming He, and Jian Sun. Distribution alignment: A unified framework for long-tail visual recognition. *arXiv preprint arXiv:2103.16370*, 2021b.

Zizhao Zhang and Tomas Pfister. Learning fast sample re-weighting without reward data. *arXiv preprint arXiv:2109.03216*, 2021.

Peilin Zhao and Tong Zhang. Stochastic optimization with importance sampling. In *Proceedings of the 32th International Conference on Machine Learning*, pp. 1–9, 2015.

Maciej Zieba, Jakub M. Tomczak, and Jerzy Świątek. Self-paced learning for imbalanced data. In *Asian Conference on Intelligent Information and Database Systems*, pp. 564–573, 2016.

A OTHER WEIGHTING METHODS

Apart from weighting schemes investigated in our study, there are also other sample weighting techniques including meta-optimization, teacher-student strategies and sampling methods.

Meta-optimization leverages an additional unbiased data set to optimize sample weight (Shu et al., 2019; Khan et al., 2018). Ren et al. (2018a) proposed the first meta-optimization method, which assigns weights to training samples on the basis of their gradient directions. Meta-class-weight (Jamal et al., 2020) exploits meta-learning to estimate class-wise weights. However, meta-optimization methods heavily rely on unbiased data sets which are unavailable in many scenarios. By comparison, our proposed FlexW is easier to implement because an additional data set is not necessary.

The teacher-student strategy uses an additional network as the teacher, with the help of the teacher network's performance to assign weights to samples in the student network (Fan et al., 2018). MentorNet (Jiang et al., 2018) uses the teacher network to assign weights to samples in the student network. Samples that are quite hard for the student network will be dropped (weights are set to 0 for these samples) in this case. However, this strategy is computationally expensive and requires an additional network.

Importance sampling aims to reduce the variance of gradient estimates by selecting samples with an adaptive sampling distribution, instead of the traditional uniform sampling (Needell et al., 2014; Zhao & Zhang, 2015). However, it needs to know the gradient of loss with respect to each of the network's parameters and for each training sample before each single gradient descent step. In the context of deep learning, this is computationally infeasible. Alternatively to importance sampling, other types of sample selection strategies have also been investigated. These are based on ranking samples according to the corresponding loss (Loshchilov & Hutter, 2016), or using more complex metrics that combine classifier uncertainty, class balance, and sample representativeness (Kabkab et al., 2016). Compared to the sampling-based methods, our proposed weighting scheme can process all samples each epoch, thus guaranteeing that we always know how the network is performing on each sample.

B THEORETICAL ANALYSIS

B.1 SUPPLEMENT TO SECTION 3.2 (REQUIREMENTS THAT THE UNIFIED WEIGHTED LOSS NEEDS TO SATISFY)

In addition to the non-negative and convex constraints introduced in Section 3.1, The unified weighted loss should also satisfy the following requirement.

- If the easy-first mode is adopted in an epoch, then the weighting function decreases with respect to the sample's learning difficulty.
- If the medium-first mode is adopted, then the weighting function increases firstly and then decreases with respect to the sample's learning difficulty.
- If the hard-first mode is adopted, then the weighting function increases with respect to the sample's learning difficulty.

The three conditions guarantee that $\mathcal{L}_{UW}$ can implement the three different priority modes, namely, easy-first, medium-first, and hard-first. $\mathcal{L}_{UW}$ also satisfies the following properties (Castells et al., 2020):

- Translation-invariance: Adding a constant to the input loss should have no effect on $\mathcal{L}_{UW}$'s gradient, i.e. $\forall K, \exists K' \mid \mathcal{L}_{UW}(w_i, l_i + K, R(w_i, \Gamma, \lambda)) = K' + \mathcal{L}_{UW}(w_i, l_i, R(w_i, \Gamma, \lambda))$, where K and $K^{'}$ are constant.
- Homogeneity: $\mathcal{L}_{UW}$ should have a multiplicative scaling behavior: $\exists \lambda, \lambda' \mid \forall\ K > 0, \mathcal{L}_{UW}(w_i, Kl_i, R(w_i, \Gamma, \lambda)) = K\mathcal{L}_{UW}(w_i, l_i, R(w_i, \Gamma, \lambda^{'}))$, where K is a constant. With this property, we can handle losses of any amplitude.

B.2 SUPPLEMENT TO SECTION 3.3 (THE DERIVATION OF FOCAL LOSS'S REGULARIZATION FUNCTION)

Because Focal loss applies l_i to approximate d_i, the optimization problem for the weights of Focal loss is shown in Eq. (13).

$$\frac{1}{N}\sum_{i=1}^{N}\max_{w_i}(w_i l_i + R(w_i,\gamma)), \tag{13}$$

where the hard-first mode is used. To implement the maximization, the following equation should be satisfied:

$$\frac{\partial(w_i l_i + R(w_i,\gamma))}{\partial w_i} = l_i + \frac{\partial R(w_i,\gamma)}{\partial w_i} = 0. \tag{14}$$

Thus, we have

$$\frac{\partial R(w_i,\gamma)}{\partial w_i} = -l_i. \tag{15}$$

We know that the weight function of Focal loss is as follows:

$$w_i = (1-p_i)^{\gamma}. \tag{16}$$

The loss can be subsequently expressed as

$$l_i = -\log p_i = -\log(1 - {w_i}^{\frac{1}{\gamma}}). \tag{17}$$

Then we have

$$\frac{\partial R(w_i,\gamma)}{\partial w_i} = \log(1 - {w_i}^{\frac{1}{\gamma}}). \tag{18}$$

By solving the above differential equation, $R(w_i,\gamma)$ of Focal loss can be obtained as follows:

$$\begin{aligned} R(w_i,\gamma) &= \textstyle\int \log(1 - {w_i}^{\frac{1}{\gamma}})dw_i \\ &= w_i \frac{{w_i}^{\frac{1}{\gamma}}\,{}_2F1(1,1+\gamma,2+\gamma,{w_i}^{\frac{1}{\gamma}})}{1+\gamma} + \log(1 - {w_i}^{\frac{1}{\gamma}}) \end{aligned}, \tag{19}$$

where ${}_2F1$ is the Hypergeometric function (Seaborn, 1991) and γ is the hyper-parameter of Focal loss.

B.3 SUPPLEMENT TO SECTION 3.4 (THEORETICAL ANALYSIS FOR PROPOSITION 1)

A strict proof for Proposition 1 is challenging. We give a proof under a special case that the weights exerted on R_{easy} are identical. Without loss of generality, the weights on each sample in R_{hard} are denoted as $(1+\epsilon)$, where $\epsilon > 0$.

Let $BiasT(c)$ and $VarT(c)$ be the values of bias and variance terms defined in Eq. (7) in Section 3.4, respectively, when the model complexity is c. First, we have

$$\left.\frac{\partial Err}{\partial c}\right|_{c^*} = \left.\frac{\partial BiasT(c)}{\partial c}\right|_{c^*} + \left.\frac{\partial VarT(c)}{\partial c}\right|_{c^*} = 0. \tag{20}$$

According to Assumptions 1 and 2, we have

$$\begin{aligned} \left.\frac{\partial BiasT_{easy}(c)}{\partial c}\right|_{c^*} + \left.\frac{\partial VarT_{easy}(c)}{\partial c}\right|_{c^*} &> 0 \\ \left.\frac{\partial BiasT_{hard}(c)}{\partial c}\right|_{c^*} + \left.\frac{\partial VarT_{hard}(c)}{\partial c}\right|_{c^*} &< 0 \end{aligned}. \tag{21}$$

Let p_{easy}, p_{medium}, p_{hard} be the proportions of samples in R_{easy}, R_{medium}, R_{hard}, respectively. We have

$$\begin{aligned} BiasT(c^*) &= p_{easy}BiasT_{easy}(c^*) + p_{medium}BiasT_{medium}(c^*) + p_{hard}BiasT_{hard}(c^*) \\ VarT(c^*) &= p_{easy}VarT_{easy}(c^*) + p_{medium}VarT_{medium}(c^*) \\ &\quad + p_{hard}VarT_{hard}(c^*) \end{aligned}. \tag{22}$$

When the weights $(1+\epsilon)$ are exerted on R_{hard}, then $BiasT(c^*)$ and $VarT(c^*)$ become

$$\begin{aligned} BiasT_\epsilon(c^*) &= p_{easy}BiasT_{easy}(c^*) + p_{medium}BiasT_{medium}(c^*) + p_{hard}BiasT_{hard} \\ &\quad + \epsilon p_{hard}BiasT_{hard}(c^*) \\ VarT_\epsilon(c^*) &= p_{easy}VarT_{easy}(c^*) + p_{medium}VarT_{medium}(c^*) \\ &\quad + p_{hard}VarT_{hard}(c^*) + \epsilon p_{hard}VarT_{hard}(c^*) \end{aligned} . \tag{23}$$

Based on Eqs. (21) and (23), we have

$$\left.\frac{\partial BiasT_\epsilon(c)}{\partial c}\right|_{c^*} + \left.\frac{\partial VarT_\epsilon(c)}{\partial c}\right|_{c^*} < 0. \tag{24}$$

Accordingly, in order to attain the new balance between the bias and variance terms, the model complexity should be increased. Alternatively, the new optimal model complexity c^*_{new} will be larger than c^*. Likewise, we have the following proposition with the similar inference manner.

***Proposition 2**:* If weights higher than one are exerted on samples in the R_{easy}, and the weights in other regions remain one, the new optimal model complexity c^*_{new} over the entire space will be smaller than c^*.

Proposition 2 supports the situation in the second requirement when easy samples are excessive.

B.4 Supplement to Section 4 (The derivation of FlexW's regularization function)

Similar to the derivation process of the Focal loss's regularizer, the underlying regularizer for FlexW when d_i is approximated by l_i can also be inferred.

In the easy-first mode, the optimization for the weights is in the following form:

$$\frac{1}{N}\sum_{i=1}^{N}\min_{w_i}(w_i l_i + R(w_i,\gamma,\alpha)). \tag{25}$$

Note that the weighting function of FlexW is as follows:

$$w_i = (1-p_i+\alpha)^\gamma e^{-\gamma(1-p_i+\alpha)}. \tag{26}$$

We also express the loss function with the weight, which is

$$l_i = -\log p_i = \log\frac{1}{1+\alpha+W(-(w_i)^{\frac{1}{\gamma}})}. \tag{27}$$

Then we solve the differential equation $\frac{\partial R(w_i,\gamma,\alpha)}{\partial w_i} = -l_i$, and the expression of the FlexW's regularizer is as follows:

$$R(w_i,\gamma,\alpha) = \int -\log\frac{1}{1+\alpha+W(-(w_i)^{\frac{1}{\gamma}})}dw_i, \tag{28}$$

where W is the Lambert W function (Corless et al., 1996) which is the inverse function of

$$f(x) = xe^x. \tag{29}$$

Likewise, the regularizer in the hard-first mode can also be inferred with the above steps.

B.5 Supplement to Section 4 (Analysis of the three priority modes achieved by FlexW)

The priority modes that the weight function can achieve are determined by the attributes of the function (e.g. extreme point). What's more, the value of the function on $p \in [0,1]$ needs to be greater than 0. The FlexW weight function we give is an example that can implement all the three priority modes, there are also other functions that can achieve this. Below we analyze FlexW.

The weight function of FlexW when d_i is approximated by $1-p_i$ is:

$$w_i = (1-p_i+\alpha)^\gamma e^{-\gamma(1-p_i+\alpha)}, \tag{30}$$

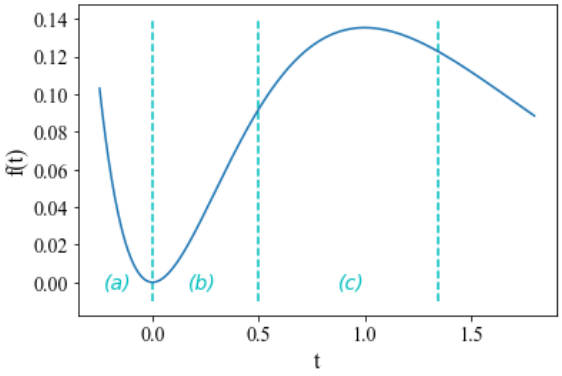

Figure A-1: The intervals of the weight function under the three priority modes.

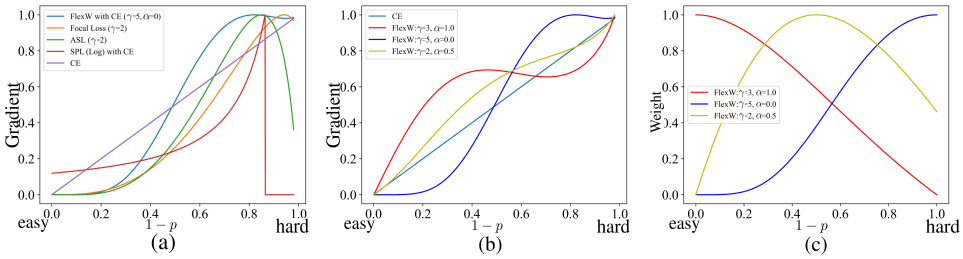

Figure A-2: Gradients of different losses.

where γ is the shape parameter and α is the translation parameter. α controls the translation of the curve so that different segments of the curve can be taken when $p \in [0, 1]$.

To simplify the form, we let $t = 1 - p + \alpha$. So that, the weight function becomes:

$$f(t) = t^{\gamma} e^{-\gamma t}. \tag{31}$$

Taking the derivative of Eq. (31), we get:

$$f^{'}(t) = \gamma t^{\gamma - 1} e^{-\gamma t}(1 - t). \tag{32}$$

Regardless of the value of γ, the function either has both a local maximum point and a local minimum point, or only contains a local maximum point or a local minimum point. Taking the case of containing both a local maximum point and a local minimum point as an example, we analyze how this function implements three priority modes:

Without considering translation, $t = 1 - p$. Fig. A-1 shows the function's curve when $\gamma = 2$. When the segment of (a) is selected, the priority mode is the easy-first; when the (b) segment is selected, the priority mode is the hard-first; when the segment of (c) is selected, the priority mode is the medium-first. Because of $p \in [0, 1]$, the curve can be translated by the parameter α, so that when $p \in [0, 1]$, different segments of the curve can be taken.

B.6 Supplement to Section 4 (Gradient analysis of FlexW)

To better understand FlexW, its loss gradient is analyzed. The loss gradient of FlexW function is:

$$\frac{d\mathcal{L}}{dz} = \frac{\partial \mathcal{L}}{\partial p} \times \frac{\partial p}{\partial z} = p(1-p)(1-p+\alpha)^{\gamma-1} e^{-\gamma(1-p+\alpha)} (\gamma \log p(p-\alpha) - \frac{1-p+\alpha}{p}), \tag{33}$$

where $p = \frac{1}{1+e^{-z}}$. The gradient of FlexW is in comparison to the gradients of cross entropy (CE) loss, Focal loss (Lin et al., 2017), SPL_Log (Jiang et al., 2014a), and ASL (Ben-Baruch et al., 2020). Fig. A-2(a) shows the gradients of different losses. Under CE loss, harder samples have larger gradients than easier ones. Focal loss increases the gradients of hard samples. However, it is sensitive to noise. ASL decreases the gradients of quiet-hard samples. Fig. A-2(b) shows the gradients of three variants of FlexW with CE loss. The weight curves of the three variants are shown in Fig. A-2(c). When the easy/medium/hard-first mode of FlexW is used, the loss gradients of easy/medium/hard samples are increased compared with those under CE loss shown in Fig. A-2(b).

Table A-1: Hyper-parameter value intervals in which the performance is stable.

Priority mode	Easy-first	Medium-first	Hard-first
Intervals	[-0.6,-0.2]×[0.1,0.4] [0.2,0.6]×[0.9,1.2]	[0.2,0.6]×[0.4,0.8] [-0.6,-0.2]×[0.4,0.8]	[0.2,0.6]×[0.1,0.4] [-0.6,-0.2]×[0.9,1.2]

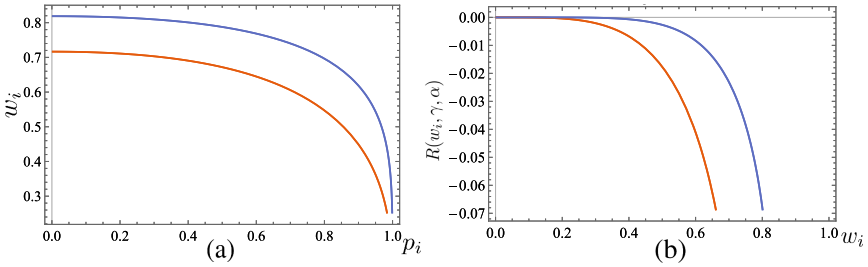

Figure A-3: Weighting function (a) and regularizer (b) curves of the FlexW in two cases.

B.7 Supplement to Section 4 (The parameter value ranges under different modes.)

For ease of use, we give the parameter value ranges with stable effects corresponding to the three priority modes which are shown in Table A-1.In the three priority modes, the recommended values of the parameters are shown in Section 4. In practical applications, grid search can be used to search parameters within the given parameter value ranges. We have verified that the performances of FlexW are stable within these ranges.

B.8 Supplement to Section 6 (Analysis of the excess of easy and hard samples)

In our point of view, the judgement whether easy or hard samples are excessive should be based on a reference or reliable prior knowledge. In other words, it is nearly impossible to judge which parts are excessive without a reference or reliable prior knowledge. The proportions of easy and hard samples on validation data can be used as the reference. Assuming that there is an effective measure of samples' learning difficulty, if the proportion of easy samples on the training set is larger than that of the validation set, then we can conclude that easy samples are excessive. If the proportion of hard samples on the training set is larger than that of the validation set, then we can conclude that hard samples are excessive.

C Experimental details and more experiments

C.1 Supplement to Section 4 (Curves of FlexW)

Fig. A-3 shows the curves of the weighting function and the regularizer $R(w_i, \gamma, \alpha)$ when ($\gamma = 1/3, \alpha = 0$) (red) and ($\gamma = 1/5, \alpha = 0$) (blue), respectively. Hard-first is used in these two cases shown in Fig. A-3. Hence, a smaller value of p_i (larger value of loss l_i) indicates a larger value of the weight w_i. Notably, the solving of w is a maximum optimization problem in this mode. The regularization function monotonically decreases with respect to the weight to prevent all weights from taking the maximum value of the weighting function.

C.2 Supplement to Section 4 (More curves of FlexW)

C.2.1 Priority mode of both-ends-first

Apart from the three modes mentioned in Section 4, FlexW can implement the priority mode of both-ends-first when both (a) and (b) segments are selected in Fig. A-1. This mode can be achieved when the weighting function has a local minimum point. The curves of both-ends-first under different

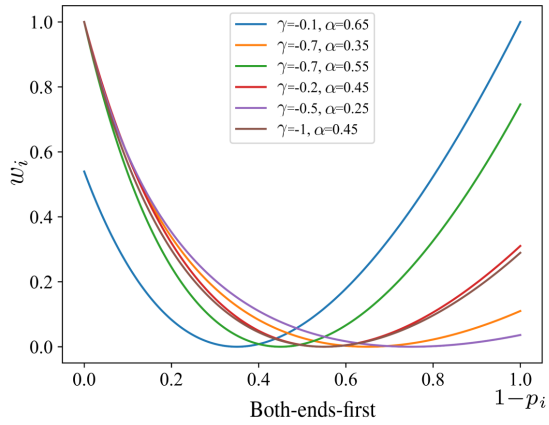

Figure A-4: Both-ends-first mode implemented by FlexW.

parameters are shown in Fig. A-4. Only a few tasks utilize this priority mode. For example, Yang et al. (2020a) proposed the self-paced balance learning that considers the priority as a combination of the easy-first and hard-category-first modes which is an approximation of the both-ends-first mode.

C.2.2 Curves of FlexW with dynamic weighting manner

As mentioned in Section 4, the dynamic weighting manner can be integrated into FlexW. Fig. A-5 shows the FlexW (easy-first and hard-first) weight curves using the dynamic weighting manner. The first and the second rows present the easy-first (the smaller the $1 - p_i$ is, the larger the weight w_i is.) and hard-first (the larger the $1 - p_i$ is, the larger the weight w_i is.) modes, respectively. In the easy-first mode, the easier the sample is, the larger the weight will be; and in the hard-first mode, the harder the sample is, the larger the weight will be. In the dynamic weighting manner, hard samples gradually participate in the training process.

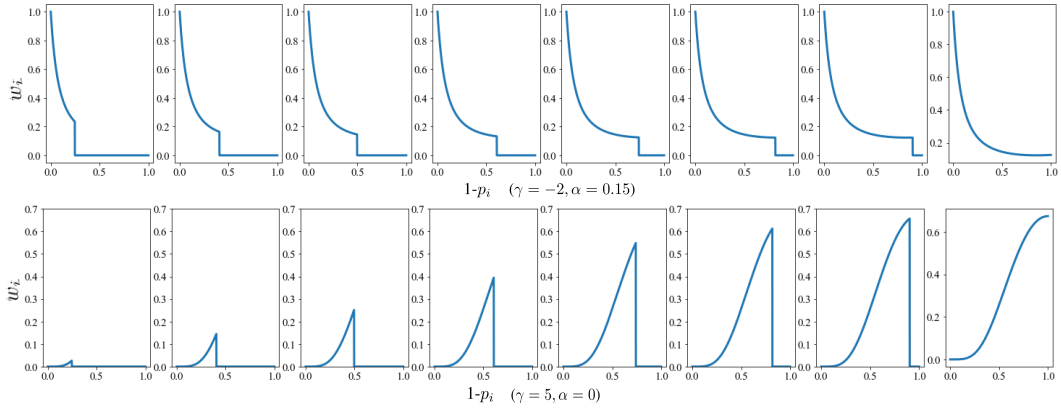

Figure A-5: Weight curves of FlexW (easy-first ($\gamma = -2, \alpha = 0.15$) and hard-first ($\gamma = 5, \alpha = 0$)) with the dynamic weighting manner.

C.2.3 The influence of different hyper-parameters on the FlexW weighting function

We can obtain different weight curves by gradually changing the value of the parameter γ in FlexW, as shown in Fig. A-6. The left four curves are in the hard-first mode, while the right four curves are in the easy-first mode. Fig. A-7 shows additional examples including easy-first ((1)), medium-first ((4), (5), (7), and (8)), and hard-first ((2), (6) and (9)) modes as well as the equal weights in (3). We obtain several weight curves by adjusting the translation parameter α while γ remains unchanged as shown in Fig. A-8. The results indicate that our FlexW weighting function can also achieve

switching from the hard-first mode to the medium-first model and then to the easy-first only when α is adjusted.

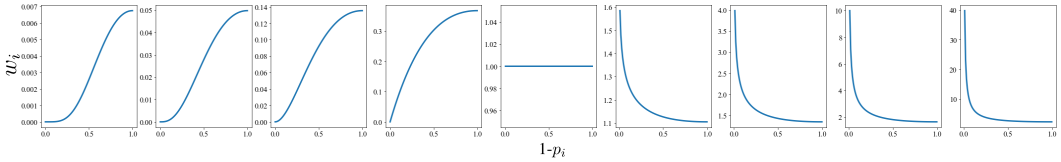

Figure A-6: Different weight curves obtained by only changing the parameter γ.

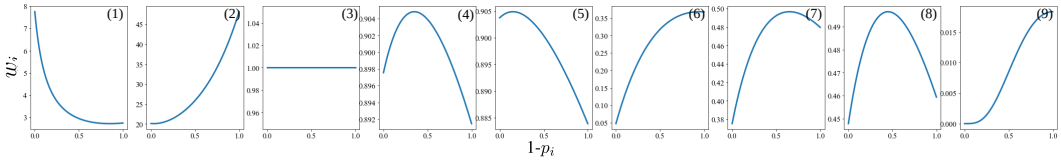

Figure A-7: Different weight curves of FlexW.

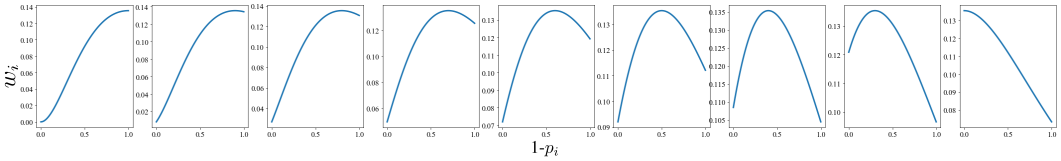

Figure A-8: Different weight curves obtained by only changing the translation parameter α.

C.3 Supplement to Section 5.1 (More details for experiments on noisy CIFAR data sets)

C.3.1 Experimental setup

CIFAR10 and CIFAR100 are two benchmark image data sets. Both data sets consist of 50,000 training samples and 10,000 test samples. Each sample is a 32×32 image from 1 out of 10 or 100 categories, respectively. They are balanced data sets where each category holds the same number of images. There are two settings of corrupted labels on the training set that are used: uniform noise and flip noise. Uniform noise is simulated according to the manner that the label of each sample is independently changed to a random category. Flip noise is simulated according to the manner that the label of each sample is independently flipped to similar classes. The settings of both types of noise follow the same setting in (Zhang et al., 2017). Wide ResNet-28-10 (WRN-28-10) (Zagoruyko & Komodakis, 2016) and ResNet-32 (He et al., 2016) are used as the basic network in the experiments under uniform and flip noises, respectively.

The comparison methods include Baseline which refers to the basic classifier network with CE loss; the robust learning methods including Reed (Reed et al., 2015), S-Model (Goldberger & Ben-Reuven, 2017), SPL (Kumar et al., 2010; Jiang et al., 2014a), Focal Loss (Lin et al., 2017), Co-teaching (Han et al., 2018), D2L (Ma et al., 2018), and MentorNet (Jiang et al., 2018); and Fine-tuning (Shu et al., 2019) which refers to fine-tuning the result of Baseline on the meta-data with clean labels to further enhance its performance.

In this experiment, the networks are trained using SGD with a momentum 0.9, a weight decay $5 \times e^{-4}$, and an initial learning rate 0.1. At the 60th, 120th, and 160th epochs, the learning rate is reduced to one-fifth of that in the previous epoch. The values of the batch size and epoch are set to 32 and 200, respectively. All the results are the average of five experiments with different seeds.

C.3.2 Results under uniform noise

In addition to the comparison under flip noise shown in Section 5.1, the test accuracies under uniform noise are shown in Table A-2. The performances of FlexW(easy-first) are better than that of FlexW(hard-first). In some cases, the performances of FlexW (hard-first) and Focal loss are close to or even better than SPL_log which is because that using loss as the criterion to distinguish noise samples form hard ones is not completely accurate. What is certain is that for the same method,

the easy-first mode will always achieve better results than the hard-first mode. What's more, the performance of FlexW is the best or the second-best in all cases under uniform noise.

Table A-2: Test accuracies (%) of the competing methods under uniform noise.

Data set	Noise ratio	Baseline	Reed Hard	S-Model	Co-teaching	SPL_Binary	SPL_Log	D2L	Focal loss	Fine-tuning	MentorNet	FlexW (hard-first)	FlexW (easy-first)
CIFAR10	0	95.60±0.22	94.38±0.14	83.79±0.11	88.67±0.25	90.81±0.34	94.94±0.22	94.64±0.33	95.70±0.15	95.65±0.15	94.35±0.42	95.26±0.42	**95.85**±0.31
	40%	68.07±1.23	81.26±0.51	79.58±0.33	74.81±0.34	86.41±0.29	77.50±0.50	85.60±0.13	75.96±1.31	80.47±0.25	87.33±0.22	86.16±0.85	**88.15±0.22**
	60%	53.12±3.03	73.53±1.54	-	73.06±0.25	53.10±1.78	53.40±0.38	68.02±0.41	51.87±1.19	78.75±2.40	**82.80±1.35**	77.96±1.11	81.87±1.23
CIFAR100	0	79.95±1.26	64.45±1.02	52.86±0.99	61.80±0.25	78.31±0.26	75.60±0.56	66.17±1.42	81.04±0.24	80.88±0.21	73.26±1.23	78.70±0.58	**81.15±0.42**
	40%	51.11±0.42	51.27±1.18	42.12±0.99	46.20±0.15	55.11±0.75	54.94±0.21	52.10±0.97	51.19±0.46	52.49±0.74	**61.39±3.99**	54.25±0.34	57.72±0.36
	60%	30.92±0.33	26.95±0.98	-	35.67±1.25	36.56±0.57	37.17±0.32	41.11±0.30	27.70±3.77	38.16±0.38	36.87±1.47	39.40±1.55	**42.50±0.87**

C.3.3 THE AVERAGE LOSSES OF CLEAN AND NOISY SAMPLES

The average losses of clean samples and noise samples under 40% flip noise are shown in Fig. A-9. The average loss of the noise samples is always higher than that of the clean ones during the training process. Therefore, using loss as the criterion to distinguish clean and noisy samples is reasonable.

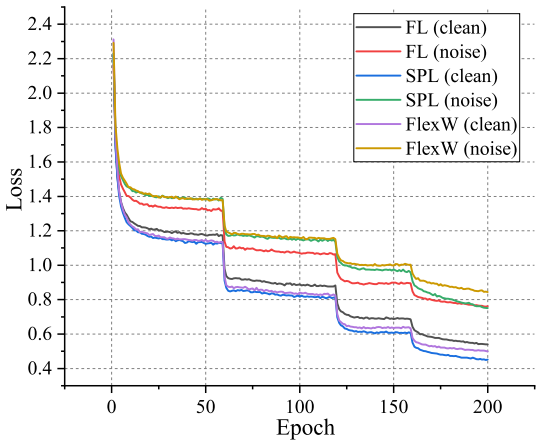

Figure A-9: The average losses of clean and noise samples of the three methods.

C.3.4 RESULTS OF MORE ROBUST METHODS UNDER THE UNIFORM NOISE

The accuracies of more robust methods under the uniform noise on CIFAR10 are shown in Fig. A-10. FlexW (easy-first) achieves the highest accuracies under most noise rates. Another easy-first method SuperLoss (Castells et al., 2020) also achieves good performance. The results indicate that the easy-first mode is more suitable for noisy data sets than the hard-first mode.

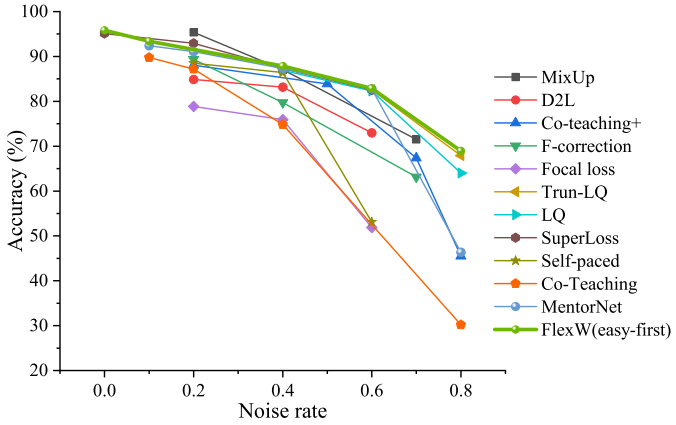

Figure A-10: Accuracies of different methods under varied uniform noise rates. The accuracies of other methods are from Castells et al. (2020).

Table A-3: The performances of different parameter settings on CIFAR10 under 40% flip noise.

Scheme	Parameters	Accuracy (%)
Easy-first	γ = -0.4, α = 0.15	85.64±0.11
Hard-first	γ = 0.4, α = 0.15	83.28±0.45
Medium-first	γ = 0.7, α = 0.45	85.81±0.45

C.3.5 THE PERFORMANCE OF FLEXW UNDER MEDIUM-FIRST MODE ON NOISY DATA

In this part, we use FlexW to implement the medium-first mode and compare its performance with those of easy-first and hard-first modes. Table A-3 indicates that both the medium-first and easy-first modes obtain better results and the hard-first performs poorly on noisy data. It verifies the conclusion that the easy-first and medium-first modes are more suitable for noisy data than the hard-first mode.

C.3.6 CONFUSION MATRICES OF FLEXW UNDER VARIED NOISE RATES

Confusion matrices of the true labels and predictions generated by FlexW on CIFAR10 under different noise rates of flip and uniform noise are shown in Fig. A-11. The results indicate that even when the data set contains heavy noise, FlexW can also achieve good results.

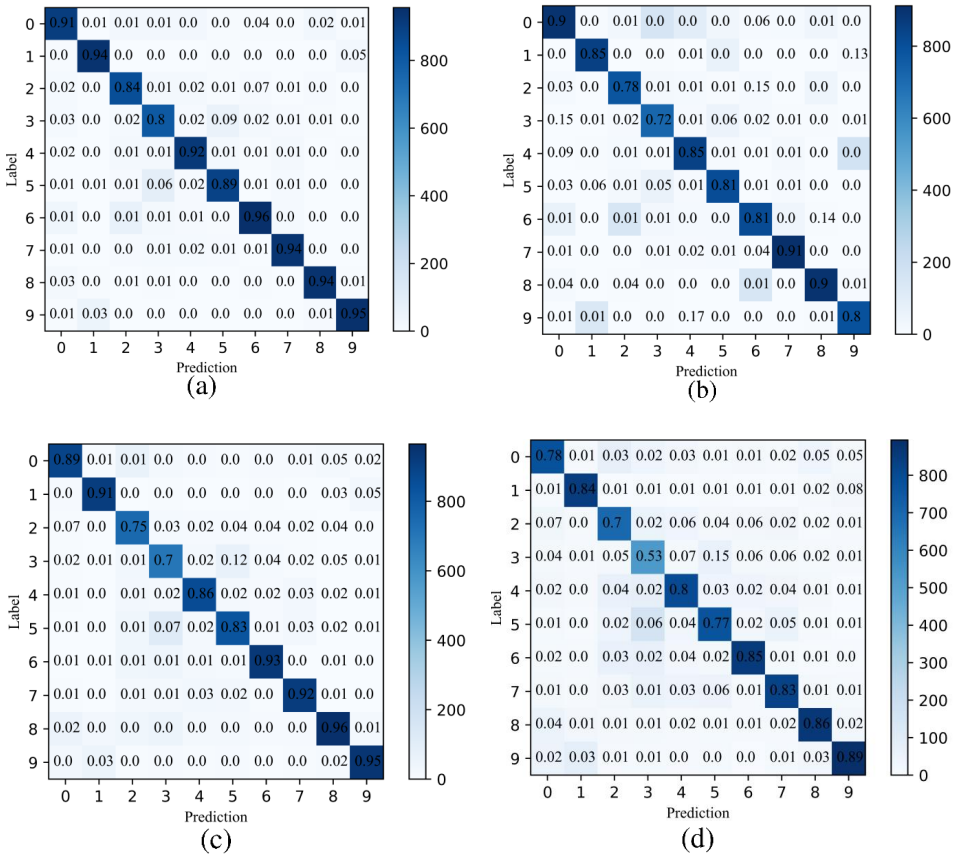

Figure A-11: Confusion matrices of the true labels and prediction results under varied noise conditions on CIFAR10. (a) 20% flip noise rate, (b) 40% flip noise rate, (c) 40% uniform noise rate, and (d) 60% uniform noise rate.

Table A-4: Details of the five graph data sets.

Data set	Categories	Features	Nodes	Edges	Label rate	Edge density
Cora	7	1,433	2,485	5,069	0.0563	0.0004
Citeseer	6	3,703	2,110	3,668	0.0569	0.0004
Pubmed	5	500	19,717	44,324	0.0030	0.0001
Coauthor CS	15	6,805	18,333	81,894	0.0164	0.0001
Coauthor Physics	5	8,415	34,493	247,962	0.0029	0.0001

C.4 Supplement to Section 5.2 (More details to experiments on imbalanced CIFAR data sets)

C.4.1 Experimental setup

Following Cui et al. (2019), we discard some training samples to construct imbalanced data sets. We build ten training sets with a varied imbalance factor $\mu \in \{200, 100, 50, 20, 10\}$. The factor μ denotes the image amount ratio between the largest and the smallest categories. It is calculated by

$$\mu = \max_i(n_i)/\min_j(n_j), \tag{34}$$

where n_i is the number of samples in the i-th category. The new sample size of Category c is calculated by the following equation (Li et al., 2021):

$$N_{\text{cs}} = N_c \times (1/\mu)^{\frac{c}{C-1}}, \tag{35}$$

where N_{cs} is the number of samples after discarding some samples. C is the number of categories. N_c is the original number of samples in category c.The compared methods include the Baseline model which uses a cross-entropy loss to train ResNet-32 on the training set, Focal loss (Lin et al., 2017), SPL (Kumar et al., 2010; Jiang et al., 2014a), Mix up (Zhang et al., 2018), LDAM (Cao et al., 2019), Class-balanced (Cui et al., 2019), L2RW (Ren et al., 2018b) which leverages an additional meta-data to adaptively assign weights for training samples, Equalised (Tan et al., 2020), and Class-balanced Fine-tuning (Cui et al., 2018) which means that the model is fine-tuned using the meta-data.

In this experiment, the optimizer used is SGD. The momentum and weight decay are set to 0.9 and 5×10^{-4}, respectively. The values of the epoch and batch size are set to 32 and 200, respectively. The initial learning rate is 0.1. At the 60th, 120th, and 160th epochs, the learning rate is reduced to one-fifth of the original. The experimental results of FlexW are the average of five repeated experiments with different seeds.

C.4.2 The accuracies of the three methods on Categories 2-9

Fig. A-12 shows the accuracies during the training process of the three methods (i.e. SPL_Binary, FlexW (hard-first), and Focal loss) on Categories 2-9. The accuracy curves during the training process of Categories 1 and 10 are shown in Section 5.2. Methods under the hard-first mode increase the accuracies of most tail categories.

C.4.3 The average weight of samples in each category on CIFAR10

Fig. A-13 shows the average weight of samples in each category on CIFAR10 data. The average weights of the two head categories (Categories 1 and 2) are much lower than those of the rest eight categories before the 150th epoch.

C.5 Supplement to Section 5.3 (More details to experiments on graph data sets)

C.5.1 Experimental setup

The details of the five graph data sets Cora, Citeseer, Pubmed (Yang et al., 2016), Coauthor CS, and Coauthor Physics (Shchur et al., 2018) are shown in Table A-4: Transudative training is used and all node features are accessible during training. We apply the full-supervised training setting used

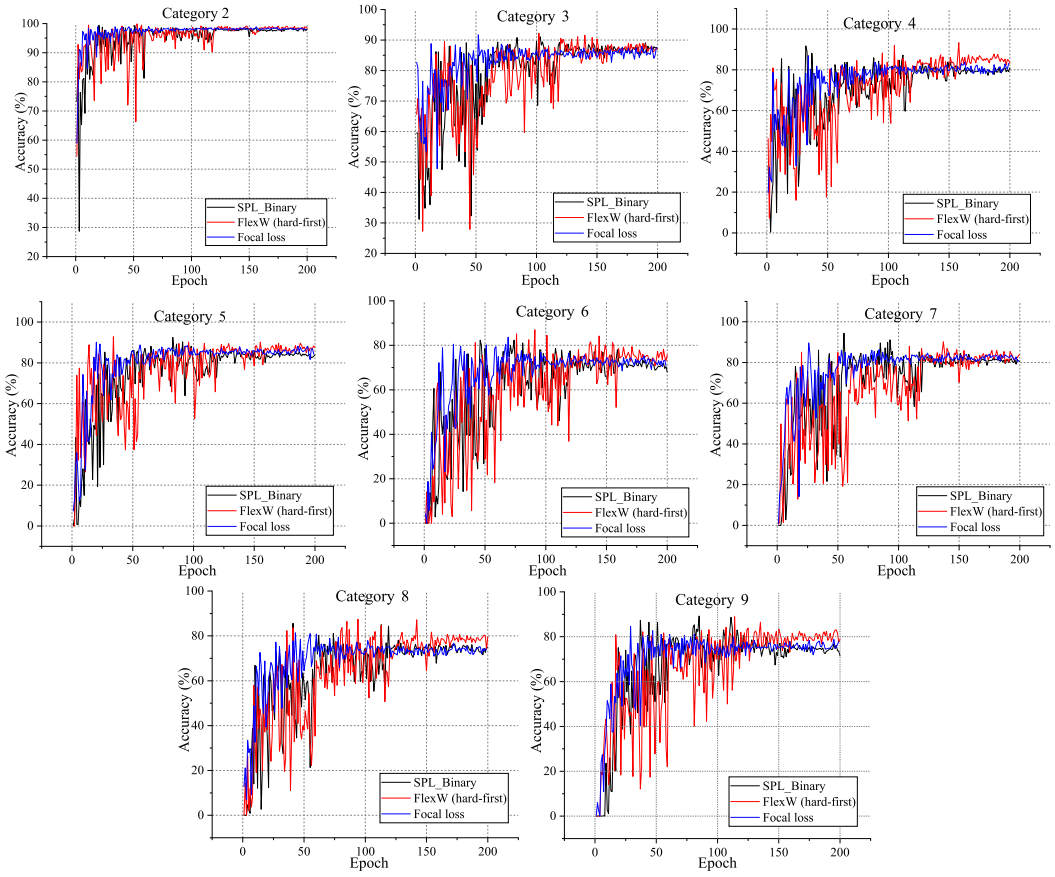

Figure A-12: Accuracies during the training process of Categories 2-9 on CIFAR10 when the imbalance factor equals to 20.

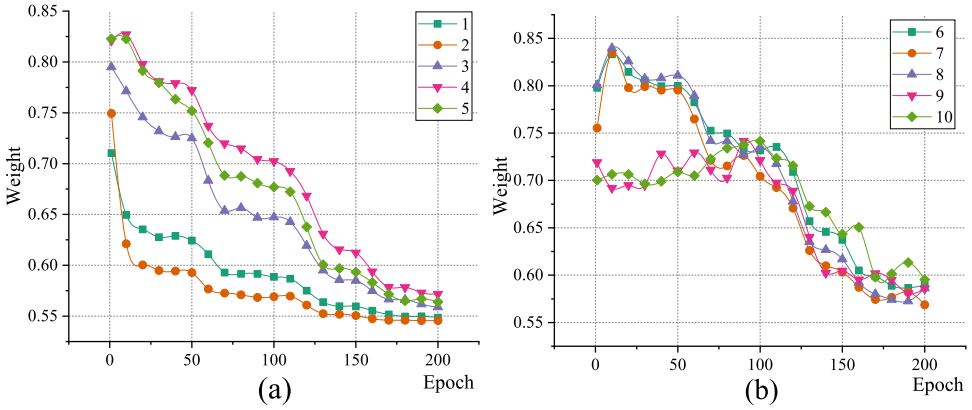

Figure A-13: The average weights of samples in the Categories 1-5 (a) and the (rest) Categories 6-10 (b) on CIFAR10.

in Huang et al. (2018) and Chen et al. (2018) on all data sets in our experiments. During training, Adam (Kingma & Ba, 2015) is used as the optimizer. The value of the learning rate is 0.001. The weight decay is set to 5×10^{-4}. The value of the epoch is set to 400. The dimension of hidden layers is 128.

C.6 Supplement to Section 5.4 (More details for the experiments on object detection)

C.6.1 Experimental setup

The PASCAL VOC (Mark et al., 2010; 2015) data set contains 20 sub-categories. The training set consists of VOC2007 and VOC2012 train+val with a total of 16,551 samples. As the training set contains excessive easy samples, it is abbreviated as VOC-e. Both the two artificially constructed training sets contain 8,000 images. For VOC-h, 7,000 images are from the images with the largest loss-conf in the original VOC training set, and the remaining 1,000 images are randomly selected from training data except for the hardest 7,000 ones. The other training set VOC-m is composed of 8,000 images with moderate loss-conf values. VOC2007 test is used as the test set with a total of 4,952 samples.

YOLOv4 (Bochkovskiy et al., 2020) is used as the basic model. The optimizer we used is SGD where the momentum and weight decay are set to 0.9 and 5×10^{-4}, respectively. The value of epoch is set to 50 and the batch size is set to 4. The initial learning rate is 1×10^{-4}, and the final learning rate is 1×10^{-6}. The value of the warm-up epoch is set to 2.

C.6.2 mAPs on each category of the three data sets

MAPs of the four weighting schemes (FL(easy-first), FL(hard-first), FlexW(easy-first), and FlexW (hard-first)) for all the 20 categories in the original VOC data (VOC-e) are shown in Table A-5. In this case, the accuracies of the hard-first weighting schemes (i.e. FL (hard-first) and FlexW (hard-first)) are relatively higher than those of easy-first on most categories. Furthermore, the performances of the FlexW (hard-first) are better than those of the FL (hard-first). This experiment supports the conclusion in Section 3.4 that when a data set contains excessive easy samples, the hard-first mode is a better choice.

The mAPs of the four weighting schemes on each category in the VOC-h are shown in Table A-6. The performances of the easy-first weighting schemes exceed those of hard-first on most categories. This comparison corroborates the conclusion in Section 3.4 that when a data set contains excessive hard samples, the easy-first mode is the primary choice.

The mAPs of the four weighting schemes for each category in the VOC-m are shown in Table A-7. In indicates that the performances of the easy-first mode exceed those of hard-first for most categories.

Table A-5: mAPs (%) of the four weighting schemes for 20 categories in VOC-e.

Category	FL (hard-first)	FL (easy-first)	FlexW (hard-first)	FlexW (easy-first)
Aeroplane	82.38	72.54	82.71	75.31
Bicycle	85.50	76.57	84.49	84.80
Bird	70.88	63.67	73.46	73.70
Boat	65.35	56.06	71.72	61.94
Bottle	72.73	67.00	75.34	67.73
Bus	81.13	76.47	81.28	76.21
Car	90.30	81.64	91.66	90.61
Cat	76.11	67.52	80.69	72.78
Chair	62.06	55.03	63.52	62.40
Cow	66.34	54.02	66.81	52.74
Diningtable	71.03	62.38	70.58	63.45
Dog	71.19	62.62	77.15	64.56
Horse	82.02	72.63	81.81	78.56
Motorbike	85.59	72.64	86.95	83.88
Person	89.22	81.92	89.88	89.39
Pottedplant	54.94	50.02	58.52	53.80
Sheep	69.56	64.62	70.17	62.75
Sofa	71.44	63.11	68.61	70.46
Train	81.34	73.22	83.08	74.80
Tvmonitor	75.15	65.50	78.43	74.08

Table A-6: mAPs (%) of the four weighting schemes for 20 categories in VOC-h.

Category	FL (hard-first)	FL (easy-first)	FlexW (hard-first)	FlexW (easy-first)
Aeroplane	69.12	62.51	75.88	69.03
Bicycle	74.60	81.02	81.53	81.27
Bird	54.83	61.03	57.06	59.49
Boat	52.14	61.60	59.19	60.23
Bottle	67.08	69.15	69.03	69.22
Bus	70.64	75.81	75.63	79.93
Car	86.98	86.90	87.60	88.77
Cat	61.19	59.01	59.31	63.58
Chair	53.77	61.91	59.47	57.70
Cow	52.15	56.11	48.27	53.01
Diningtable	68.65	69.40	67.26	67.66
Dog	60.49	47.54	46.88	58.57
Horse	70.43	68.00	73.76	72.35
Motorbike	82.23	79.55	84.27	80.88
Person	86.87	89.05	87.90	88.29
Pottedplant	52.80	57.73	51.55	53.74
Sheep	64.64	66.30	54.78	65.70
Sofa	62.73	64.37	67.08	65.07
Train	71.89	73.09	75.30	76.31
Tvmonitor	69.21	76.01	71.65	74.12

C.7 Supplement to Section 5.5 (More details to experiments on benchmark CIFAR data sets)

C.7.1 Experimental setup

The following networks are used: GoogLeNet (Szegedy et al., 2015), VGG (Karen & Andrew, 2014), ResNet (He et al., 2016), MobileNet (Howard et al., 2017), MobileNetV2 (Howard et al., 2017), DenseNet (Huang et al., 2017), and Wide ResNet (Zagoruyko & Komodakis, 2016). General

Table A-7: mAPs (%) of the four weighting schemes on 20 categories in VOC-m.

Category	FL (hard-first)	FL (easy-first)	FlexW (hard-first)	FlexW (easy-first)
Aeroplane	59.80	68.74	70.51	70.31
Bicycle	63.22	73.51	68.82	75.23
Bird	41.85	54.05	52.86	55.15
Boat	40.86	56.11	52.88	54.14
Bottle	34.65	53.94	46.63	54.24
Bus	64.31	73.68	73.52	75.21
Car	81.12	86.57	85.93	85.74
Cat	62.65	67.44	61.48	63.74
Chair	34.59	40.94	41.21	46.84
Cow	54.33	46.85	44.09	51.81
Diningtable	45.26	45.48	50.68	51.79
Dog	56.12	52.61	53.58	55.78
Horse	72.25	73.98	59.98	61.09
Motorbike	68.28	76.35	72.48	76.69
Person	78.45	83.04	80.22	82.82
Pottedplant	30.32	36.19	38.53	34.69
Sheep	49.29	54.14	51.03	56.65
Sofa	52.28	63.41	59.51	62.67
Train	67.68	76.02	76.28	77.17
Tvmonitor	57.58	64.17	62.58	62.35

Table A-8: Accuracies (%) of different methods on CIFAR10 and CIFAR100.

Data set	Baseline	SPL	Inverse-SPL	SPLD	LGL	Focal loss	FlexW (easy-first)	FlexW (hard-first)
CIFAR10	93.03	92.60	92.96	92.85	93.97	93.45	93.73	**94.00**
CIFAR100	71.11	70.30	70.50	70.25	74.17	74.13	**74.96**	73.01

pre-processing steps are used in training including zero-padding with four pixels, random crops with size 32×32, random flips, and standardizing the data.

All the networks are trained to converge from scratch utilizing an SGD optimizer. The weight decay and momentum are set to $5 \times e^{-4}$ and 0.9, respectively. The value of the epoch is 200. The learning rate is set to 0.05. The value of batch size is set to 32. The experimental results of FlexW are the average of five repeated experiments with different initialization.

C.7.2 Results of different methods on the standard data sets

Apart from the results presented in Section 5.5, FlexW (easy-first and hard-first) is compared with SPL (Kumar et al., 2010) (easy-first), Inverse-SPL (Cheng et al., 2019) (hard-first), SPLD (Jiang et al., 2014b) (easy-first), LGL (Cheng et al., 2019), and Focal loss (Lin et al., 2017) (hard-first). VGG-16 (Karen & Andrew, 2014) is used and the results are shown in Table A-8.

It indicates that neither the easy-first mode nor the hard-first mode is consistently better on both standard data sets. The hard-first mode is better on CIFAR10, while the easy-first mode is better on CIFAR100. As stated in Section 5.5, FlexW can implement different priority modes by adjusting its parameters. The optimal mode can be selected by comparing their validation performances.

C.7.3 The performances of easy-first and hard-first modes on the standard CIAFR10 data.

We visualize the feature spaces of test samples under the three priority modes including easy-first, baseline, and hard-first using the t-SNE algorithm (van der Maaten & Hinton, 2008). Fig. A-14 shows the visualized feature spaces under different weighting schemes. The priority modes of the three pictures from left to right are easy-first mode, baseline, and hard-first mode, respectively. Baseline means that all samples have equal weights. A universal optimal setting can not be obtained

Table A-9: Accuracies (%) of the three methods under different models on CIFAR10.

Model	Method	Acc	Model	Method	Acc
VGG-16	Baseline	92.71	ResNet-50	Baseline	94.70
	SPL_Binary	92.99		SPL_Binary	93.50
	FlexW(easy-first)	**94.00**		FlexW(easy-first)	**94.73**
ResNet-110	Baseline	93.41	ResNet-32	Baseline	92.56
	SPL_Binary	92.69		SPL_Binary	92.84
	FlexW(easy-first)	**93.47**		FlexW(easy-first)	**93.26**
ResNet-34	Baseline	93.85	GoogLeNet	Baseline	94.18
	SPL_Binary	92.48		SPL_Binary	94.24
	FlexW(easy-first)	**94.15**		FlexW(easy-first)	**95.02**
MobileNet	Baseline	90.86	MobileNetV2	Baseline	93.35
	SPL_Binary	91.00		SPL_Binary	93.47
	FlexW(easy-first)	**92.18**		FlexW(easy-first)	**93.48**
DenseNet	Baseline	94.68	Wide ResNet	Baseline	92.54
	SPL_Binary	94.33		SPL_Binary	92.48
	FlexW(easy-first)	**94.88**		FlexW(easy-first)	**92.61**

on standard data set, as indicated by the minimal differences among the feature spaces under the three weighting schemes. This finding is consistent with the analysis in Section 7.2.

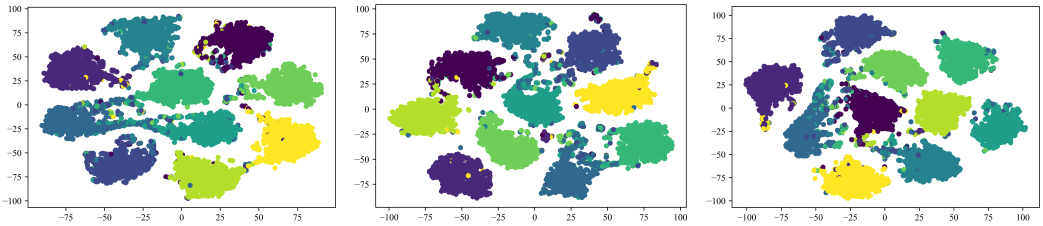

Figure A-14: Feature spaces under different priority modes. Left: easy-first; middle: baseline; right: hard-first.

C.7.4 The performances of FlexW on more basic networks

On CIFAR10, the performances of three different schemes (FlexW (easy-first), SPL_Binary, and baseline) with different basic networks are shown in Table A-9. When different basic networks are used, FlexW (easy-first) consistently outperforms the baseline model and SPL_Binary. The results on CIFAR100 are shown in Table A-10. FlexW still achieves the best results. In some cases of Tables A-9 and A-10, SPL_Binary outperforms the baseline, whereas it is inferior to the baseline in some other cases. Therefore, SPL_Binary has only marginal benefits on the standard data sets, which is consistent with the conclusion in (Wu et al., 2021).

C.8 Supplement to Section 5.6 (More experimental analysis for FlexW)

C.8.1 Experiments on adding prior knowledge

Prior knowledge can be encoded in the regularizer as previously stated. We consider two types of prior knowledge, including the sample-level and the category-level, where an eight-layer GCN (Bruna et al., 2014) is used. The category-level prior knowledge used is the proportion of each category. A smaller proportion indicates more hard samples in this category as shown in Fig. 5(c). The weighting scheme should also assign large weights for samples in tail categories. We use the following proportion of heterogeneous nodes around each node π_i for sample-level prior knowledge:

$$\pi_i = h_i / a_i, \tag{36}$$

where h_i is the number of heterogeneous nodes around Node i and a_i is the number of adjacent nodes of Node i. A large value of π_i indicates a high learning difficulty. We assign small weights for these nodes to further alleviate the over-smoothing phenomenon.

Table A-10: Accuracies (%) of the three methods under different models on CIFAR100.

Model	Method	Acc	Model	Method	Acc
VGG-16	Baseline	71.42	ResNet-50	Baseline	75.23
	SPL_Binary	70.69		SPL_Binary	75.32
	FlexW(easy-first)	**74.26**		FlexW(easy-first)	**75.61**
ResNet-110	Baseline	71.67	ResNet-32	Baseline	70.50
	SPL_Binary	70.13		SPL_Binary	70.05
	FlexW(easy-first)	**71.89**		FlexW(easy-first)	**70.82**
ResNet-34	Baseline	74.13	GoogLeNet	Baseline	76.51
	SPL_Binary	73.46		SPL_Binary	73.54
	FlexW(easy-first)	**74.65**		FlexW(easy-first)	**76.68**
MobileNet	Baseline	65.35	MobileNetV2	Baseline	72.64
	SPL_Binary	65.15		SPL_Binary	73.16
	FlexW(easy-first)	**67.05**		FlexW(easy-first)	**73.27**
DenseNet	Baseline	76.97	Wide ResNet	Baseline	68.82
	SPL_Binary	76.99		SPL_Binary	70.98
	FlexW(easy-first)	**77.38**		FlexW(easy-first)	**72.72**

Table A-11: Accuracies (%) when different levels of prior knowledge are considered.

Method	Acc
Original	90.70
SPL_Log	93.44
SPL_Binary	93.16
FlexW(easy-first)	93.71
FlexW(easy-first+sample-level prior)	**95.65**
FlexW(easy-first+category-level prior)	**95.08**

Before and after adding different prior knowledge, the performance comparison on the Coauthor CS data set is shown in Table A-11 which indicates that adding prior knowledge to FlexW further improves the performance.

C.8.2 Experiments on varied modes

Unlike existing weighting schemes that remain fixed priority mode during the training process, FlexW can flexibly switch the priory mode during the training process. For example, in the early training stages on imbalanced data, the easy-first mode can be leveraged to ensure the performance of the head categories, and then the hard-first mode can be leveraged to improve the performance of the tail categories in later periods. Table A-12 shows the performance of FlexW with varied modes during training. "Varied modes" means that easy-first is used in the first 100 epochs and hard-first is used in the rest of the epochs. This strategy achieves good results in some cases (imb200, imb100, and imb50).

C.8.3 Experiments on imbalanced and noisy data sets

Although label noise and imbalance are usually studied as independent research, these two label deviations may happen simultaneously in real-world applications. Few studies on this kind of data set exist (Karthik et al., 2021; Zhang & Pfister, 2021). We discussed how to select the optimal

Table A-12: Accuracies (%) under the hard-first mode and varied modes of FlexW on CIFAR10.

Imbalance factor	200	100	50	20	10
FlexW (hard-first)	69.40	75.33	80.05	85.46	88.50
FlexW (varied modes)	**69.59**	**75.63**	**80.43**	85.03	88.00

learning strategy when both types of deviations exist. Tables A-13 and A-14 show the results of different priority modes in different cases.

Table A-13: Accuracies (%) on the imbalanced CIFAR10 under 20% flip noise.

Imbalance factor	FlexW (easy-first)	FlexW (hard-first)	FlexW (medium-first)
200	56.30	**57.37**	55.61
50	70.59	**73.92**	73.73

Table A-14: Accuracies (%) on the imbalanced CIFAR10 under 40% flip noise.

Imbalance factor	FlexW (easy-first)	FlexW (hard-first)	FlexW (medium-first)
200	**46.79**	44.73	45.31
50	**58.63**	53.29	55.82

The main deviation of the data set is the imbalance when the data set contains relatively less (e.g., 20%) noise. Thus, increasing the weights of samples in the tail categories is preferred, that is, to take the hard-first mode. Meanwhile, the influence of noise is strong when the data set contains relatively larger (e.g., 40%) noisy labels. At this time, the weighting scheme of the hard-first mode will yield poor performance, and thus the easy-first mode is preferred. Therefore, the priority mode adopted depends on which deviation is more serious when both types of deviations exist in the data set.