Unlocker: Disentangle the Deadlock of Learning from Label-noisy and Long-tailed Data

Shu Chen[1] , **Hongjun Xu**[1] , **Ruichi Zhang**[1] , **Mengke Li**[2] , **Yonggang Zhang**[3]
Yang Lu[1] *, **Bo Han**[4] , **Yiu-ming Cheung**[4] , **Hanzi Wang**[1]
[1]Key Laboratory of Multimedia Trusted Perception and Efficient Computing
Ministry of Education of China, Xiamen University
[2]College of Computer Science and Software Engineering, Shenzhen University
[3]Hong Kong University of Science and Technology
[4]Hong Kong Baptist University
{chenshu, xuhongjun}@stu.xmu.edu.cn, luyang@xmu.edu.cn, zhangyg@ust.hk

Abstract

In real world, the observed label distribution of a dataset often mismatches its true distribution due to noisy labels. In this situation, noisy labels learning (NLL) methods directly integrated with long-tail learning (LTL) methods tend to fail due to a dilemma: NLL methods normally rely on unbiased model predictions to recover true distribution by selecting and correcting noisy labels; while LTL methods like logit adjustment depends on true distributions to adjust biased predictions, leading to a deadlock of mutual dependency defined in this paper. To address this, we propose Unlocker, a bilevel optimization framework that integrates NLL methods and LTL methods to iteratively disentangle this deadlock. The inner optimization leverages NLL to train the model, incorporating LTL methods to fairly select and correct noisy labels. The outer optimization adaptively determines an adjustment strength, mitigating model bias from over- or under-adjustment. We also theoretically prove that this bilevel optimization problem is convergent by transferring the outer optimization target to an equivalent problem with a closed-form solution. Extensive experiments on synthetic and real-world datasets demonstrate the effectiveness of our method in alleviating model bias and handling long-tailed noisy label data. Code is available at https://github.com/ChenShu248/Unlocker.

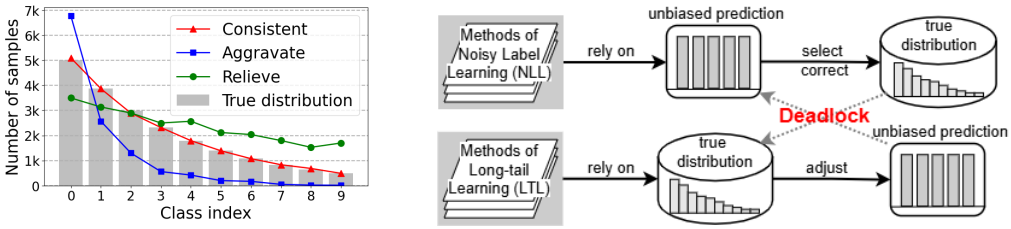
1 Introduction

Long-tailed noisy label learning (LTNLL) focuses on addressing the coexistence of long-tailed distribution and noisy labels in datasets. Existing LTNLL methods [1, 2, 3, 4, 5, 6] are commonly under the potential assumption that the observed long-tailed distribution based on the noisy labels is consistent with the true distribution of the clean labels. However, empirical observations from real world reveal that noisy labels can alter the original true distribution, leading to a deviation between the observed distribution and the true underlying distribution of the dataset [7, 8, 9]. Due to the diverse real-world noise patterns, the deviations of the distribution are various, typically including three situations: *consistent*, *relieve*, and *aggravate*, as illustrated in Figure 1a.

When addressing this issue of distribution deviation, neither noisy label learning (NLL) nor long tail learning (LTL) can be effective due to a *deadlock* dilemma of mutaul dependency. A mainstream in NLL highly rely on the model prediction to select and correct noisy labels to recover clean labels [10, 11, 12, 13, 14]. Yet in long-tailed scenarios, model predictions get biased towards head classes,

*Corresponding Author: Yang Lu

39th Conference on Neural Information Processing Systems (NeurIPS 2025).



(a) Distribution deviation

(b) Deadlock dilemma

Figure 1: (a) Three typical scenarios of distribution deviation between the observed distribution and the true distribution. *Consistent*, *relieve*, and *aggravate* denote that the imbalance ratio (IR) of the observed distribution is identical to, lower than, and higher than the IR of the true distribution, respectively. (b) Deadlock between the noisy label learning (NLL) methods and the long-tail learning (LTL) methods: NLL relies on the unbiased prediction to recover true distribution, while LTL requires the true distribution to recalibrate the biased prediction, creating a circular dependency.

posing challenges for selecting and correcting noisy samples in tail classes [1, 8]. Logit adjustment methods in LTL can recalibrate the biased predictions while relying on the true label distribution [15, 16, 17, 18, 19]. Due to the true distribution deviation, these methods fail to perform accurate adjustments. At this point, we can see a deadlock between NLL and LTL, as shown in Figure 1b. NLL relies on the unbiased predictions to restore a true distribution by selecting and correcting. Conversely, LTL can provide unbiased predictions but relies on a true distribution.

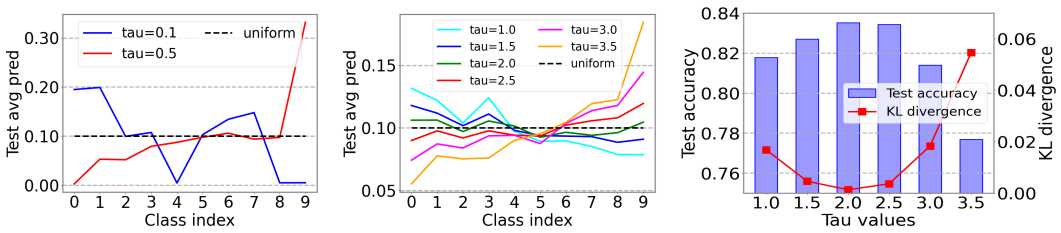
To disentangle the deadlock, we propose a novel method `Unlocker` based on the bilevel optimization framework. The core idea is jointly estimating the true distribution of the training set and optimizing an appropriate parameter τ for the adjusting strength. We define the inner problem as training a model using NLL adjusted by LTL, and the outer problem as optimizing the τ based on the inner problem's solution. In this framework, model predictions are adjusted using both τ and the estimated true distribution. This enables the model to better distinguish noisy labels. Improved noisy label selection and correction enhance distribution estimation accuracy. Accurate distribution estimation facilitates τ optimization, promoting model to reduce bias on the test set. This iterative process continues until convergence, gradually breaking the deadlock. Our main contributions are as follows:

- We define a new and challenging long-tailed noisy label problem of the deviation between the observed distribution based on noisy labels and the true distribution of the clean labels.
- We define a deadlock dilemma between the NLL and LTL, where their mutaul dependency renders both methods ineffecvtive.
- We propose a novel method `Unlocker` which bases on the bilevel optimization to effectively combine NLL and LTL by adaptively optimizing the adjustment strength.

2 Preliminaries

2.1 Noisy Label Selection

Noisy label learning has developed various methods to improve model performance by addressing label noise. A mainstream is to select noisy label samples from clean ones and correct the noisy labels [11, 12, 13, 14], which has been proven effective in noisy label learning tasks. Specifically, the selection of noisy labels relies on logit-based metrics, which are designed to maximize the discrimination between noisy and clean labels, enabling an efficient separation of the two subsets. Based on these metrics, the original dataset $\mathcal{D}$ is partitioned into a clean subset $\mathcal{D}_{clean}$ and a noisy subset $\mathcal{D}_{noisy}$. Then, the noisy labels in the $\mathcal{D}_{noisy}$ are corrected by leveraging techniques such as semi-supervised learning [20, 21]. The overall training objective is formulated as the total loss L_{NLL}, which is the sum of the clean loss L_{clean} computed on $\mathcal{D}_{clean}$ and the noisy loss L_{noisy} calculated on D_{noisy}. Conventionally, the cross-entropy loss is utilized for L_{clean} to ensure accurate classification on reliable data. For L_{noisy}, more robust loss functions are often applied to reduce the impact of noise during training [22].



(a) (b) (c)

Figure 2: (a) Average prediction on CIFAR-10 test set of a NLL method DivideMix [11] adjusted by the true distribution prior $\mathbb{P}_{\text{train}}(y)$. The training set is CIFAR-10 with imbalance ratio (IR) of 50 and 40% symmetric noise. The model is highly sensitive to the τ, making it prone to under-adjustment or over-adjustment. (b) Test average predictions of models trained using logit adjustment (LA) across different τ on a clean long-tailed CIFAR-10 dataset with IR of 50. As τ increases, the average predictions transition from skewing toward head classes to uniformity and then to skew toward tail classes. (c) Corresponding test accuracies and KL divergences (between test average predictions and a uniform distribution) for varying τ. When $\tau = 2.0$ or $\tau = 2.5$, the trained model attains the highest accuracy and ouputs the most uniform average predictions.

2.2 Logit Adjustment

Logit adjustment [17, 23, 19] is an effective technique for long-tailed learning. In a standard classification with C classes, the class posterior follows Bayes' theorem $\mathbb{P}(y|x) = \frac{\mathbb{P}(x|y)\mathbb{P}(y)}{\mathbb{P}(x)}$. Since $\mathbb{P}(x)$ is constant across all classes, it can be omitted. Thus, cross-entropy training yields $\mathbb{P}_{train}(y|x) \propto \mathbb{P}(x|y)\mathbb{P}_{train}(y)$, where $\mathbb{P}_{train}(y)$ is the class prior in training datasets. For balanced test datasets, a balanced prior $\mathbb{P}_{\text{bal}}(y) = \frac{1}{C}$ is desired, leading to $\mathbb{P}_{\text{bal}}(y|x) \propto \mathbb{P}(x|y)$. Assuming the likelihood $\mathbb{P}(x|y)$ is unchanged between the train and balanced datasets, we have the following relationship:

$$\mathbb{P}_{\text{test}}(y|x) \propto \frac{\mathbb{P}_{\text{train}}(y|x)}{\mathbb{P}_{\text{train}}(y)} \propto \text{softmax}(\boldsymbol{\theta}_y(x) - \log \mathbb{P}_{\text{train}}(y)), \tag{1}$$

where $\boldsymbol{\theta}_y(x)$ is the model's logit for class y. To further improve flexibility, a temperature parameter τ which modulates the adjustment intensity is introduced, leading to the general LA formulation:

$$\arg\max_{y \in [C]} \text{softmax}(\boldsymbol{\theta}_y(x) - \tau \cdot \log \mathbb{P}_{\text{train}}(y)). \tag{2}$$

Typically, $\tau > 0$ is used for post-hoc adjustment, while $\tau < 0$ can be incorporated as logit adjustment loss during training.

3 `Unlocker`: Breaking the Deadlock with Bilevel Optimization

3.1 Motivation

To break the deadlock, we first attempt to adjust the logits of a classical NLL method DivideMix [11] using the true distribution prior $\mathbb{P}_{\text{train}}(y)$. Ideally, incorporating $\mathbb{P}_{\text{train}}(y)$-based adjustments leads to unbiased confidence across all classes, reflected by a uniform average prediction on the test set, as depicted by the black line in Figure 2a. However, directly employing $\mathbb{P}_{\text{train}}(y)$ for LA results in suboptimal model performance due to under-adjustment or over-adjustment [24]. Specifically, as $\mathbb{P}_{\text{train}}(y)$ is the same, when the parameter $\tau = 0.5$ (red line), the tail classes are over-adjusted, causing the average prediction to skew towards the tail and damaging the head, as shown in Figure 2a. Conversely, when $\tau = 0.1$ (blue line), the adjustments are insufficient, remaining the predictions biased towards the heads. Without an appropriate τ to modulate the adjusting strength, even with the precise $\mathbb{P}_{\text{train}}(y)$, the model could not achieve an ideal unbiased state.

To further investigate τ in modulating model bias, we eliminate the influence of noisy label and implement a simple control variable experiment on a clean long-tailed dataset. We test different τ of $\{1.0, 1.5, 2.0, 2.5, 3.0, 3.5\}$ with the same $\mathbb{P}_{train}(y)$. As shown in Figure 2b, when $\tau = 1.0$, the tail classes are under-adjusted. As τ increasing, the test average predictions become more uniform.

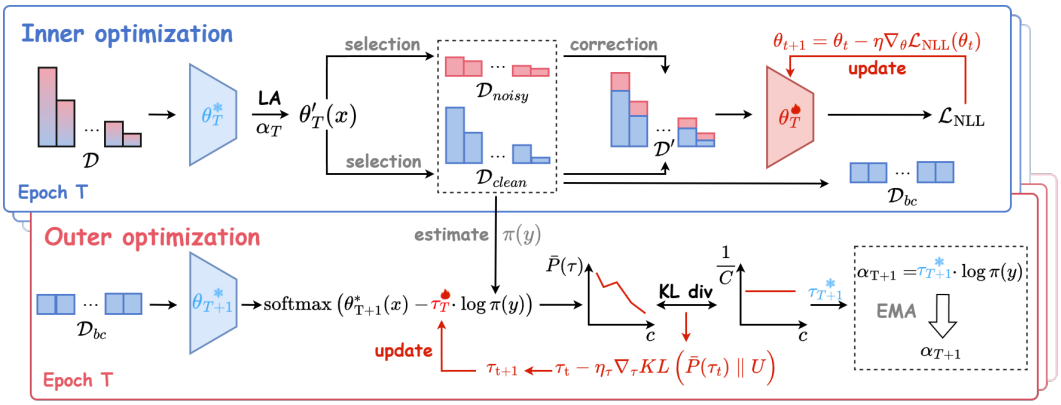


Figure 3: The illustration of the proposed method `Unlocker` based on the bilevel optimization framework, which iteratively disentangle the deadlock between the noisy label learning (NLL) and long-tail learning (LTL). In epoch T, the inner optimization trains the model θ using NLL integrated with logit adjustment in LTL. By adjusting logits as $\theta'_T(x)$, it fairly selects and corrects noisy labels, and trains θ. The outer optimization adaptively updates the adjusting strength parameter τ to modulate model bias towards balance. An EMA strategy is introduced to stabilize the adjustments after the whole optimizations completing. Parameters frozen during optimization are denoted by the snowflake (no gradient flow), while trainable parameters are marked by fire.

When $\tau = 2.0$ or $\tau = 2.5$, the average predictions are closest to uniformity, and the model achieved the highest test accuracy as depicted in Figure 2c. When further increasing τ, the model gets over-adjusted, and the predictions skew towards the tail. This simple experiment demonstrates that an appropriate τ is crucial for achieving unbiased model state and influences the test accuracy.

3.2 Bilevel Optimization

Based on the motivation, we propose `Unlocker` which utilizes the bilevel optimization framework to iteratively disentangle the deadlock. The inner layer optimization trains model using NLL integrating LA, and the outer layer optimizes a learnable parameter τ to regulate the strength of adjustments. Through gradient coupling between layers, this framework maintains model capability against noisy labels while dynamically calibrating class bias.

3.2.1 Inner Optimization

Given a long-tailed noisy label training set $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$, the inner layer employs NLL methods integrating LA to train model θ. At epoch T, for the noisy label selection, an adjustment α_T (post-hoc) is applied to the row logits $\theta_T(x)$ (gradient-free) to achieve fairer noise recognition, particularly enhancing the recognition accuracy of the tail classes:

$$\theta'_T(x) = \theta_T(x) - \alpha_T. \tag{3}$$

Based on the adjusted logits $\theta'_T(x)$, selecting metrics of NLL methods are computed to divide $\mathcal{D}$ into $\mathcal{D}_{clean}$ and $\mathcal{D}_{noisy}$. For the noisy label correction for $\mathcal{D}_{noisy}$, we follow the method in [25] to apply post-hoc adjustment on the generation of corrected labels. With $\mathcal{D}_{clean}$ and corrected $\mathcal{D}_{noisy}$, the model training loss $\mathcal{L}_{NLL}$ can be computed and the model θ_T updated at step t is given by:

$$\theta_{t+1} = \theta_t - \eta \nabla_\theta \mathcal{L}_{NLL}(\theta_t), \tag{4}$$

where η is the learning rate. Upon completing T epochs, the optimal inner parameters θ^*_{T+1} are fixed as the inner solution for the outer optimization.

Before diving into the outer optimization, two critical components are required in addition to support the outer object: (i) the construction of a balanced and clean subset $\mathcal{D}_{bc}$, and (ii) the estimation of the training set class prior $\pi(y)$.

Construct $\mathcal{D}_{bc}$. During the process of noisy label selection, we further filter the top $q\%$ of samples with the highest confidence of clean as a high-purity clean subset. As the clean subset follows a long-tail distribution, we apply over-sampling and form the final balanced clean subset $\mathcal{D}_{bc}$.

Estimate $\pi(y)$. As $\mathbb{P}_{train}(y)$ is unknown in our long-tailed noisy labels problem, we estimate a proxy distribution prior $\pi(y)$ to approximate it. After the correction for $\mathcal{D}_{noisy}$ completing, we estimate $\pi(y)$ based on the labels of the $\mathcal{D}_{clean}$ and $\mathcal{D}_{noisy}$.

3.2.2 Outer Optimization

The outer optimizes a learnable parameter τ to dynamically regulate the adjusting strength to modulate model bias towards balance. The core objective is to minimize the discrepancy between the model's class confidence and a uniform distribution. To measure model bias, we leverage the balanced clean validation subset $\mathcal{D}_{bc}$ constructed in 3.2.1. Given the fixed inner parameters θ^*_{T+1}, the model's logits on $\mathcal{D}_{bc}$ are $\theta^*_{T+1}(x)$ (gradient-free), and the τ_T-regulated prediction distribution is:

$$P(y \mid x; \tau_T) = softmax\left(\theta^*_{T+1}(x) - \tau_T \cdot \log(\pi(y)\right), \tag{5}$$

where $\log(\pi)$ is the fixed class prior term estimated in Sec. 3.2.1. Averaging $P(y \mid x; \tau_T)$ over $\mathcal{D}_{bc}$ yields the class-averaged prediction $\bar{P}(\tau_T) = \mathbb{E}_{x \sim \mathcal{D}_{bc}} P(y \mid x; \tau_T)$, leading to the outer objective:

$$\min_{\tau} KL\left(\bar{P}(\tau_T) \parallel U\right), \quad U_c = \frac{1}{C}\ (\forall c \in \{1, \ldots, C\}), \tag{6}$$

where U is a uniform prediction tensor, C is the number of classes. Minimizing this divergence adapts τ_T to drive $\bar{P}(\tau_T)$ toward uniformity, forcing balanced class confidence on the test set. The update of τ_T at step t is derived via gradient descent:

$$\tau_{t+1} = \tau_t - \eta_\tau \nabla_\tau KL\left(\bar{P}(\tau_t) \parallel U\right). \tag{7}$$

After completing the optimization of τ_T, the adjustments are calculated as $\alpha_{\mathrm{T+1}} = \tau_{\mathrm{T+1}} \cdot \log \pi(y)$. To ensure the stability of the adjustments during the training, especially when the $\pi(y)$ fluctuates in the early training, we adopt an Exponential Moving Average (EMA) strategy to update the adjustments. At the T-th epoch, we update the adjustments as follows:

$$\alpha_{T+1} = \beta \cdot \alpha_{T+1} + (1 - \beta) \cdot \alpha_T, \tag{8}$$

where β is a hyperparameter that controls the decay rate of the EMA. $\alpha_{\mathrm{T+1}}$ is then passes into epoch $T + 1$ to adjust the inner-model training.

3.2.3 Training Overview

As illustrated in Figure 3, through the bilevel optimization, we effectively combine the NLL method and LA to handle long-tailed noisy label training data, breaking the mutual dependency. First, in inner optimization, we perform post-hoc logit adjustment on the selection of the NLL methods, enabling a more fair selection of noisy labels in tail classes. Based on the selection results, we construct a balanced clean subset $\mathcal{D}_{bc}$ and partition the training set into $\mathcal{D}_{clean}$ and $\mathcal{D}_{noisy}$. Next, we use the NLL methods to correct the noisy labels in $\mathcal{D}_{noisy}$, and estimate the $\pi(y)$. Then, the model is trained and adjusted using the corrected labels. In outer optimization, the τ is optimized to minimize the discrepancy between model preference and uniform distribution based on the $\mathcal{D}_{bc}$. After the optimization of τ is completed, we update the adjustments through EMA to ensure the stability of the training. The detailed pseudocode of the training process can be found in the Appendix A.1.

3.3 Theoretical Analysis

In this section, we establish the convergence of our bilevel optimization framework by deriving a closed-form approximation for the outer optimization of τ. Specifically, we first prove that the outer objective is differentiable and exists a global minimum. Then, leveraging the uniform target distribution, we show that the optimal τ can be approximated via least-squares minimization, reducing the bi-level problem to a normal optimization. This reduction guarantees the convergence of our method. The detailed processes of proof are provided in the Appendix A.2.

Theorem 1 (Differentiability and Gradient Descent Applicability for Outer Optimization)**.** *Under the assumptions that $\theta^*(x)$ is continuously differentiable with respect to τ and $\pi_c(y) > 0$ for all classes $c \in \{1, 2, \ldots, C\}$, the outer objective function $J(\tau) = KL\left(\bar{P}(\tau) \parallel U\right)$ is continuously differentiable with respect to τ. The gradient descent update rule $tau_{t+1} = \tau_t - \eta_\tau \nabla_\tau J(\tau_t)$ ensures that $J(\tau)$ decreases monotonically along the negative gradient direction.*

Based on the Theorem 1, we can further prove the existence of a global minimum.

Lemma 1 (KL Divergence Bounds)**.** *By the definition of KL divergence, the objective function* $J(\tau) = KL(\bar{P}(\tau)\|U)$ *establishes the following lower bound and upper bound:*

$$0 \leq J(\tau) = \log C - H(\bar{P}(\tau)) \leq \log C.$$

Theorem 2 (Existence of Global Minimum)**.** *Given the continuity of* $J(\tau)$ *and its lower bound* $J(\tau) \geq 0$ *and upper bound* $\lim_{\tau\to\pm\infty} J(\tau) = \log C$ *in Lemma 1, and* $\lim_{\tau\to\pm\infty} J(\tau) = \log C$*, there exists at least one global minimum point* $\tau^* \in \mathbb{R}$*.*

In summary, the objective function $J(\tau)$ is bounded and has a global minimum, providing a theoretical guarantee for the optimization process. Leveraging the uniform prior U, we derive a closed-form approximation for τ via least-squares minimization of the residual function.

Proposition 1 (Least-Squares Closed-Form Solution)**.** *Define* $\mu_c = \frac{1}{N}\sum_x \theta_c(x)$*, the least-squares closed-form solution for* τ *is:*

$$\tau^{LS} = \frac{\sum_c (\log \pi_c(y) - \log \pi_1)(\mu_c - \mu_1)}{\sum_c (\log \pi_c(y) - \log \pi_1)^2}. \tag{9}$$

This τ^{LS} reduces the bi-level optimization problem to a single-level optimization, establishing the theoretical convergence guarantee for our framework.

4 Experiments

4.1 Experimental Setup

Datasets. We conduct simulated experiments on CIFAR-10/100[26], including three cases: *consistent*, *relieve*, and *aggravate* which we summarize in Figure 1a . CIFAR-10/100 contains 50,000 training images and 10,000 test images of size 32 × 32 pixels, where CIFAR-10 contains 10 classes and CIFAR-100 contains 100 classes. To simulate all scenarios, we first construct a long-tailed dataset with an imbalance ratio (IR) by exponential decay. $IR = N_{max}/N_{min}$ represents the ratio of the number of samples of the majority class to the number of samples of the minority class.

Before generating label noise, due to the real-world noise patterns are complex and diverse, we systematically explore the effects of existing noise addition methods on the imbalance ratio of the long-tailed dataset. The detailed analysis results refer to the Appendix A.4. We find that the joint noise hardly changes the original long-tailed distribution in the case of low imbalance ratio, the symmetry noise alleviates the imbalance ratio, and the t2h noise aggravates. Based on these findings, we introduce the joint noise to the clean long-tailed dataset to simulate the *consistent* case, use the symmetry noise to simulate the *relieve* case, and inject the t2h noise to simulate the *aggravate* case. The detailed noise addition methods refer to the Appendix A.3. In the following experiments, the imbalance rate is selected in $\{10, 50, 100\}$, and the noise rate η which describes the proportion of label noise in the training dataset is set to $\{0.4, 0.6\}$.

We also evaluate the performance of our method on real-world datasets, including Red Mini-ImageNet [27], Clothing1M [28] and WebVision-50 [29]. Red Mini-ImageNet contains 100 classes with 50,000 training images and 5,000 testing images, annotated via controlled noise protocols. Clothing1M contains 1 million training images obtained from online shopping websites, with 50k, 14k, and 10k images split into clean labels for training, validation, and testing across 14 classes. WebVision consists of 2.4 million images from Google and Flickr, sharing the same 1,000 categories as ImageNet, and includes 50,000 human-annotated validation and test images. Following the experimental setting of [11], we use the first 50-class subset (WebVision-50) for training and evaluate on both WebVision validation set and ILSVRC12 validation set for the same 50 classes.

Baselines. We compare our method with the following four categories of methods: (1) Noisy label learning (NLL) methods: including DivideMix [11], UNICON [12] and DPC [14]. (2) NLL methods with post-hoc LA based on real distribution (NLL+LA post-hoc). (3) NLL methods with LA based on real distribution during training (NLL+LA). (4) Long-tailed noisy label learning methods (LTNLL), including RoLT [1], TABASCO [8] and DaSC [4]. We select the baseline NLL methods to combine with our method. Additionally, for real-world dataset, we further include baselines: MentorNet [30], Co-teaching [10], HAR [31], UCL [2], RCAL [3], GSS [5].

Table 1: Test accuracy (%) comparison of different methods on the CIFAR-100 dataset under varying imbalance ratios (IR), noise types and noise rates η, involving three scenarios of true distribution shifts. * denotes results from the original papers. Green numbers indicate improvements of our method combined with the NLL method over the original NLL method. Boldface represents the best performance in each case.

Dataset	CIFAR-100									
Types	consistent		relieve				aggravate			
IR	10		50		100		10		50	
η	$joint.40\%$	$joint.60\%$	$sym.40\%$	$sym.60\%$	$sym.40\%$	$sym.60\%$	$t2h.40\%$	$t2h.60\%$	$t2h.40\%$	$t2h.60\%$
RoLT [1]	32.57	21.44	22.68	14.76	23.51*	16.61*	32.65	22.12	20.28	14.08
TABASCO [8]	55.31	45.40	40.91	31.28	36.91*	26.25*	45.37	43.60	32.18	17.53
DaSC [4]	58.06	45.64	41.96	32.17	36.40	29.59	55.67	39.93	34.03	13.47
DivideMix [11]	68.96	62.54	50.75	39.56	45.13	37.15	63.24	65.76	51.53	48.68
DivideMix+LA (post-hoc)	68.53	62.74	32.37	17.53	23.63	11.89	70.18	68.95	52.41	52.68
DivideMix+LA	68.53	63.87	16.30	5.31	7.31	2.51	68.67	63.94	46.56	42.93
DivideMix+`Unlocker`	71.95	68.97	60.38	53.34	54.26	45.07	72.20	69.05	62.59	58.07
vs. DivideMix	↑2.99	↑6.43	↑9.63	↑16.98	↑9.13	↑7.92	↑8.96	↑3.29	↑11.06	↑9.39
UNICON [12]	63.63	62.42	52.16	44.53	46.82	39.83	60.37	58.61	49.60	40.53
UNICON+LA (post-hoc)	67.15	63.13	53.18	45.27	46.84	40.44	65.41	58.81	50.32	42.78
UNICON+LA	62.39	61.40	8.53	50.19	2.75	1.80	67.58	58.64	55.94	41.67
UNICON+`Unlocker`	69.11	65.53	54.83	52.95	48.38	44.87	65.97	60.49	56.18	44.70
vs. UNICON	↑5.48	↑3.11	↑2.67	↑8.42	↑1.56	↑5.04	↑5.60	↑1.88	↑6.58	↑4.17
DPC [14]	70.81	54.91	44.66	34.84	39.43	23.71	70.03	11.32	36.73	1.04
DPC+LA (post-hoc)	69.97	55.13	39.77	30.88	39.59	21.62	70.05	22.85	40.67	2.44
DPC+LA	70.01	50.24	18.16	5.37	5.33	2.38	69.88	61.76	28.36	3.21
DPC+`Unlocker`	71.86	66.78	57.19	50.01	50.18	45.31	72.09	65.78	55.34	49.92
vs. DPC	↑1.05	↑11.84	↑12.53	↑15.17	↑10.75	↑21.60	↑2.06	↑54.46	↑18.61	↑48.88

Table 2: Top 1 and Top 5 test accuracy on Webvision and ImageNet validation sets. The best results are bolded. The experimental results of other methods are from [5].

Test	WebVision		ILSVRC12	
	Top-1	Top-5	Top-1	Top-5
Standard	62.5	80.8	58.50	81.8
Co-teaching [10]	63.58	85.20	61.48	84.70
MentorNet [30]	63.00	81.40	57.80	79.92
HAR [31]	75.5	90.7	70.3	90.0
RoLT+ [1]	77.64	92.44	74.64	92.48
RCAL [3]	76.24	92.83	73.60	93.16
GSS [5]	83.64	95.86	**74.17**	95.22
DivideMix+`Unlocker`	**83.79**	**96.55**	73.21	**96.37**

Table 3: Test accuracy (%) on Red-Mini-Imagenet dataset. The best results are bolded. The experimental results of other methods are from [5].

IR	10		100	
η	20%	40%	20%	40%
ERM	40.42	31.46	30.88	31.46
DivideMix [11]	48.76	48.96	33.00	34.72
UNICON [12]	40.18	41.64	31.86	31.12
HAR [31]	46.61	38.71	32.60	31.30
ULC [2]	48.12	47.06	34.24	34.84
TABASCO [8]	50.20	49.68	37.20	37.12
GSS [5]	52.33	50.91	40.25	36.58
DivideMix+`Unlocker`	**53.29**	**51.48**	**41.51**	**37.14**

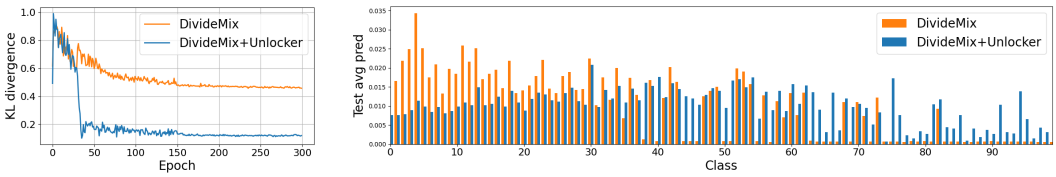
Implementation Details. To ensure a fair comparison with existing methods, we keep the training configurations consistent with the baseline NLL methods. Specifically, we employ the 18-layer ResNet as the backbone architecture. The mini-batch size is fixed at 256. All models are optimized using SGD with a momentum of 0.9. A random seed of 123 is used across all experiments to ensure reproducibility. For NLL methods combined with LA in baselines, the τ is set to 1.0. For our proposed method `Unlocker`, the learnable parameter τ is initialized to 1.0 and optimized using SGD with a momentum of 0.9. The initial learning rate for τ is set to 0.1, which is adjusted to 0.01 at the 150-th epoch. The β in EMA to update adjustments is set to 0.9. All experiments are executed on a GeForce RTX 3090 GPU using the PyTorch 1.8.0 framework to maintain hardware consistency.

4.2 Results on Simulated Scenarios

We conduct a comprehensive performance comparison across three scenarios of the distribution deviations in Table 1. Our method combines with the NLL methods (NLL+`Unlocker`) demonstrates superiority over the LTNLL methods under all three conditions. Regarding NLL methods, integrating `Unlocker` yields substantial accuracy improvements. On CIFAR-100, the combination achieves performance boosts ranging from 1.05% to 54.46%. In contrast, NLL methods directly combined

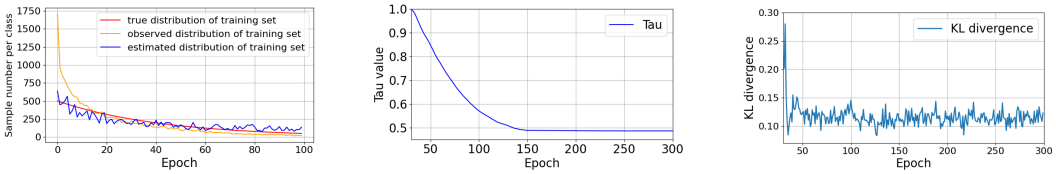
Table 4: Test accuracy (%) comparison of different methods on the Clothing1M dataset The best results are in bold. The experimental results of other methods are from [5].

Methods	CE	MentorNet [30]	Co-teaching [10]	DivideMix [11]	ULC [2]	RCAL+ [3]	GSS [5]	DivideMix+Unlocker
Accuracy	65.42	67.25	67.94	74.76	74.87	74.97	75.83	**76.94**



(a) (b)

Figure 4: Bias comparison of different models trained on CIFAR-100 (IR=100, sym.40%) (a) KL divergence between the average predictions on CIFAR-100 test set and a uniform prediction during training. (b) The average predictions in the last epoch on CIFAR-100 test set.



(a) (b) (c)

Figure 5: (a) The estimated training set distribution at the last epoch on CIFAR-100 with an imbalance ratio (IR) of 10 and t2h.60%. (b) The convergence trajectory of the parameter τ during training. (c) The KL divergence between the average predictions on the balanced clean subset $\mathcal{D}_{bc}$ and the CIFAR-100 test set. (b) and (c) are conducted on CIFAR-100 with IR=100 and sym.40%.

with true-distribution-based LA (both post-hoc and training-time LA) often suffer from the problem of over-adjustment, causing training collapse where predictions concentrate on a single class. Our method adaptively adjusts the parameter τ during training, effectively alleviating the issues of over- or under-adjustment. This mechanism enables a collaboration with NLL methods, jointly addressing the challenges of long-tailed noisy labels data. As a result, our method achieves the best performance among all methods, showing its robust adaptability to varying imbalance ratios and noise types. The results of the comparison experiment on CIFAR-10 are presented in the Appendix A.5.

4.3 Results on Real-world Datasets

We performed comparative experiments on three real-world datasets to validate the robustness of our method in practical setting. As shown in Table 2, DivideMix+`Unlocker` achieves 83.79% Top-1 and 96.55% Top-5 accuracy on the WebVision-50 validation set, outperforming all baselines. On the ILSVRC12 validation set, it achieves the highest 96.37% Top-5 accuracy and competitive 73.21% Top-1 accuracy. For Red-Mini-Imagenet, as shown in Table 3, our method attains 53.29% accuracy under IR=10 and η=20%, outperforming SOTA (52.33%) and improving upon DivideMix by 4.5%. Under conditions (IR=100, η=40%), it maintains 36.37% accuracy. On Clothing1M, as shown in Table Table 4, our method outperforms other methods by achieving 76.94% accuracy. These experiments fully verified the ability of our method in real-world long-tailed noisy label scenarios, enabling NLL methods to achieve a more uniform state against long-tail bias.

4.4 Effectiveness Study and Discussions

Average Prediction Analysis. To validate the effectiveness of our method in reducing models bias to achieve a more uniform state, we conduct experiments by training the models of different methods on CIFAR-100 training set with $IF = 100$ and $\eta = sym.40\%$, and outputting models' average predictions on the CIFAR-100 test set. In Figure 4a, during training, the KL divergence between the

test average prediction of the DivideMix and the uniform prediction stabilizes at a relatively high value, as depicted by the orange line. This means the model's average prediction is still far away from uniform, indicating persistent bias of the model. In contrast, by combining with our method `Unlocker`, the KL divergence down to approximately 0.08, signifying that the model's test average predictions approach uniformity and the model gets more unbiased. Figure 4b demonstrates the average prediction in the last epoch of the different model. As the orange bars showing, the test average predictions of the DivideMix skew toward head classes, neglecting tail classes. Conversely, our method's prediction (blue bars) distributes evenly across classes. These results collectively validate that our method effectively mitigates long-tailed impact and promotes balancer model state.

Effectiveness of Distribution Estimation. To verify the effectiveness of the distribution estimation module in our method, we conduct experiments on CIFAR-100 with IR = 10 and t2h.60%. As shown in the Figure 5a, the red line denotes the true distribution of the training set, the orange line represents the observed distribution based on noisy labels, and the blue line is the training set distribution estimated by our method. Notably, the imbalance ratio of the observed distribution is significantly higher than that of the true distribution, indicating that noisy labels aggravate the imbalance ratio of the true distribution. Our method estimates the distribution (blue line) that closely aligns with the true distribution (red line), demonstrating its capability to accurately capture the training set distribution.

Convergence of τ. We demonstrate the convergence of the parameter τ for our method trained on the CIFAR-100 dataset with IR = 100 and sym.40%. As shown in the Figure 5b, our method achieves the stable convergence of the parameter τ. Specifically, the optimization of τ starts after the warm-up of 30 epochs. In the first 100 epochs, the value of τ decreases rapidly. After 100 epochs, the descent rate slows down, and it converges to a value near 0.48 around 150 epochs.

Effectiveness of $\mathcal{D}_{bc}$. The balanced clean subset $\mathcal{D}_{bc}$ serves as a validation set to measure the model bias. Thus it is critical that $\mathcal{D}_{bc}$ can reflect the model's class confidence. To validate this, we conduct experiments on CIFAR-100 with IR = 100 and sym.40%, tracking the average predictions of the model on $\mathcal{D}_{bc}$ and the test set at each epoch, By computing their KL divergence, as shown in the Figure 5c, the discrepancy remains at a relatively low level of approximately 0.12. While there are slight fluctuations, they are in an acceptable small range. This result indicates that the average predictions on $\mathcal{D}_{bc}$ closely align with those on the test set, demonstrating that $\mathcal{D}_{bc}$ reflects model bias.

5 Related Work

Long-tailed and noisy label learning faces challenges in distinguishing clean samples from noisy ones in tail classes. RoLT [1] proposes a prototype-based noise detection method using class centroid distances to select noisy samples. ULC [2] combines class-specific noise modeling with uncertainty quantification to account for cognitive and incidental uncertainties. TABASCO [8] employs a weighted JS divergence and adaptive centroid distance to distinguish. These methods often integrate semi-supervised learning to refine predictions after noise detection. HAR [31] applies heteroscedastic adaptive regularization to high-uncertainty and low-density data points. RCAL [3] leverages unsupervised contrastive learning to eliminate noisy samples and restore representation distributions, enhancing model generalization. Detailed related work on long-tail learning and noisy label learning individually is provided in the Appendix A.6.

6 Conclusion

In this paper, we address the challenging problem of long-tailed noisy label learning where the observed distribution based on noisy label deviates from the true distribution. When addressing this issue, a deadlock dilemma between noisy label learning (NLL) and long tail learning (LTL) arises. To disentangle the deadlock and tackle the long-tailed noisy label problem, we propose `Unlocker`, a bilevel optimization framework that iteratively optimizes an adjustment strength parameter τ to effectively combine the NLL methods and LTL methods. Extensive experiments on synthetic and real-world datasets demonstrate that `Unlocker` significantly outperforms SOTA methods. However, our method has limitations: (i) potential impurity of $\mathcal{D}_{bc}$, and (ii) approximation errors in distribution estimation. Our future research will focus on address these limitations.

Acknowledgements

This study was supported in part by the National Natural Science Foundation of China under Grants 62376233, 62306181 and 62376235; in part by the Natural Science Foundation of Fujian Province under Grant 2024J09001; in part by the RGC Young Collaborative Research Grant C2005-24Y; in part by the NSFC / Research Grants Council (RGC) Joint Research Scheme under Grant N_HKBU214/21; in part by the General Research Fund of RGC under the Grants 12201323 and 12200725; in part by the RGC Senior Research Fellow Scheme under the Grant SRFS2324-2S02; and in part by Xiaomi Young Talents Program. YGZ was funded by Inno HK Generative AI R&D Center.

References

[1] Tong Wei, Jiang-Xin Shi, Wei-Wei Tu, and Yu-Feng Li. Robust long-tailed learning under label noise. *arXiv preprint arXiv:2108.11569*, 2021.

[2] Yingsong Huang, Bing Bai, Shengwei Zhao, Kun Bai, and Fei Wang. Uncertainty-aware learning against label noise on imbalanced datasets. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 6960–6969, 2022.

[3] Manyi Zhang, Xuyang Zhao, Jun Yao, Chun Yuan, and Weiran Huang. When noisy labels meet long tail dilemmas: A representation calibration method. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15890–15900, 2023.

[4] Jae Soon Baik, In Young Yoon, Kun Hoon Kim, and Jun Won Choi. Distribution-aware robust learning from long-tailed data with noisy labels. In *Proceedings of the European Conference on Computer Vision*, pages 160–177. Springer, 2024.

[5] Yuan Wang, Yakun Chang, Ying Qin, Yao Zhao, and Shikui Wei. Unbiased sample selection and label improvement for mitigating noisy labels in class-imbalanced datasets. *IEEE Transactions on Circuits and Systems for Video Technology*, 2025.

[6] Zhuo Li, He Zhao, Anningzhe Gao, Dandan Guo, Tsung-Hui Chang, and Xiang Wan. Prototype-oriented clean subset extraction for noisy long-tailed classification. *IEEE Transactions on Circuits and Systems for Video Technology*, 2025.

[7] Jiaheng Wei, Zhaowei Zhu, Hao Cheng, Tongliang Liu, Gang Niu, and Yang Liu. Learning with noisy labels revisited: A study using real-world human annotations. *arXiv preprint arXiv:2110.12088*, 2021.

[8] Yang Lu, Yiliang Zhang, Bo Han, Yiu-ming Cheung, and Hanzi Wang. Label-noise learning with intrinsically long-tailed data. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1369–1378, 2023.

[9] MingCai Chen, Yuntao Du, Wenyu Jiang, Baoming Zhang, Shuai Feng, Yi Xin, and Chongjun Wang. Robust logit adjustment for learning with long-tailed noisy data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 15830–15838, 2025.

[10] Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. *Advances in Neural Information Processing Systems*, 31, 2018.

[11] Junnan Li, Richard Socher, and Steven CH Hoi. Dividemix: Learning with noisy labels as semi-supervised learning. *arXiv preprint arXiv:2002.07394*, 2020.

[12] Nazmul Karim, Mamshad Nayeem Rizve, Nazanin Rahnavard, Ajmal Mian, and Mubarak Shah. Unicon: Combating label noise through uniform selection and contrastive learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9676–9686, 2022.

[13] Yifan Li, Hu Han, Shiguang Shan, and Xilin Chen. Disc: Learning from noisy labels via dynamic instance-specific selection and correction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24070–24079, 2023.

[14] Chen-Chen Zong, Ye-Wen Wang, Ming-Kun Xie, and Sheng-Jun Huang. Dirichlet-based prediction calibration for learning with noisy labels. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 17254–17262, 2024.

[15] Kaidi Cao, Colin Wei, Adrien Gaidon, Nikos Arechiga, and Tengyu Ma. Learning imbalanced datasets with label-distribution-aware margin loss. *Advances in Neural Information Processing Systems*, 32, 2019.

[16] Jiawei Ren, Cunjun Yu, Xiao Ma, Haiyu Zhao, Shuai Yi, et al. Balanced meta-softmax for long-tailed visual recognition. *Advances in Neural Information Processing Systems*, 33:4175–4186, 2020.

[17] Aditya Krishna Menon, Sadeep Jayasumana, Ankit Singh Rawat, Himanshu Jain, Andreas Veit, and Sanjiv Kumar. Long-tail learning via logit adjustment. In *Proceedings of the 9th International Conference on Learning Representations*, 2021.

[18] Mengke Li, Yiu-ming Cheung, and Yang Lu. Long-tailed visual recognition via gaussian clouded logit adjustment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6929–6938, 2022.

[19] Fan Zhang, Wei Qin, Weijieying Ren, Lei Wang, Zetong Chen, and Richang Hong. Gradient-aware logit adjustment loss for long-tailed classifier. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 3190–3194. IEEE, 2024.

[20] David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. Mixmatch: A holistic approach to semi-supervised learning. *Advances in Neural Information Processing Systems*, 32, 2019.

[21] Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin A Raffel, Ekin Dogus Cubuk, Alexey Kurakin, and Chun-Liang Li. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *Advances in Neural Information Processing Systems*, 33:596–608, 2020.

[22] Zhilu Zhang and Mert Sabuncu. Generalized cross entropy loss for training deep neural networks with noisy labels. *Advances in Neural Information Processing Systems*, 31, 2018.

[23] Youngkyu Hong, Seungju Han, Kwanghee Choi, Seokjun Seo, Beomsu Kim, and Buru Chang. Disentangling label distribution for long-tailed visual recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6626–6636, 2021.

[24] Feng Hong, Jiangchao Yao, Zhihan Zhou, Ya Zhang, and Yanfeng Wang. Long-tailed partial label learning via dynamic rebalancing. *arXiv preprint arXiv:2302.05080*, 2023.

[25] Tong Wei and Kai Gan. Towards realistic long-tailed semi-supervised learning: Consistency is all you need. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3469–3478, 2023.

[26] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[27] Lu Jiang, Di Huang, Mason Liu, and Weilong Yang. Beyond synthetic noise: Deep learning on controlled noisy labels. In *Proceedings of the International Conference on Machine Learning*, pages 4804–4815. PMLR, 2020.

[28] Tong Xiao, Tian Xia, Yi Yang, Chang Huang, and Xiaogang Wang. Learning from massive noisy labeled data for image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2691–2699, 2015.

[29] Wen Li, Limin Wang, Wei Li, Eirikur Agustsson, and Luc Van Gool. Webvision database: Visual learning and understanding from web data. *arXiv preprint arXiv:1708.02862*, 2017.

[30] Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *Proceedings of the International Conference on Machine Learning*, pages 2304–2313. PMLR, 2018.

[31] Kaidi Cao, Yining Chen, Junwei Lu, Nikos Arechiga, Adrien Gaidon, and Tengyu Ma. Heteroskedastic and imbalanced deep learning with adaptive regularization. *arXiv preprint arXiv:2006.15766*, 2020.

[32] Pierre Baldi and Peter J Sadowski. Understanding dropout. In *Advances in Neural Information Processing Systems*, volume 26, 2013.

[33] Chen Shu, Mengke Li, Yiqun Zhang, Yang Lu, Bo Han, Yiu-ming Cheung, and Hanzi Wang. Classifying long-tailed and label-noise data via disentangling and unlearning. *arXiv preprint arXiv:2503.11414*, 2025.

[34] Yu Yao, Tongliang Liu, Bo Han, Mingming Gong, Jiankang Deng, Gang Niu, and Masashi Sugiyama. Dual t: Reducing estimation error for transition matrix in label-noise learning. *Advances in Neural Information Processing Systems*, 33:7260–7271, 2020.

[35] Xuefeng Li, Tongliang Liu, Bo Han, Gang Niu, and Masashi Sugiyama. Provably end-to-end label-noise learning without anchor points. In *Proceedings of the International Conference on Machine Learning*, pages 6403–6413. PMLR, 2021.

[36] De Cheng, Tongliang Liu, Yixiong Ning, Nannan Wang, Bo Han, Gang Niu, Xinbo Gao, and Masashi Sugiyama. Instance-dependent label-noise learning with manifold-regularized transition matrix estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16630–16639, 2022.

[37] LIN Yong, Renjie Pi, Weizhong Zhang, Xiaobo Xia, Jiahui Gao, Xiao Zhou, Tongliang Liu, and Bo Han. A holistic view of label noise transition matrix in deep learning and beyond. In *The 11th International Conference on Learning Representations*, 2022.

[38] Rui Zhao, Bin Shi, Jianfei Ruan, Tianze Pan, and Bo Dong. Estimating noisy class posterior with part-level labels for noisy label learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22809–22819, 2024.

[39] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357, 2002.

[40] Andrew Estabrooks, Taeho Jo, and Nathalie Japkowicz. A multiple resampling method for learning from imbalanced data sets. *Computational Intelligence*, 20(1):18–36, 2004.

[41] Xu-Ying Liu, Jianxin Wu, and Zhi-Hua Zhou. Exploratory undersampling for class-imbalance learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 39(2):539–550, 2008.

[42] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2980–2988, 2017.

[43] Seulki Park, Jongin Lim, Younghan Jeon, and Jin Young Choi. Influence-balanced loss for imbalanced visual classification. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 735–744, 2021.

[44] Jiaan Luo, Feng Hong, Jiangchao Yao, Bo Han, Ya Zhang, and Yanfeng Wang. Revive reweighting in imbalanced learning by density ratio estimation. *Advances in Neural Information Processing Systems*, 37:79909–79934, 2024.

[45] Boyan Zhou, Quan Cui, Xiu-Shen Wei, and Zhao-Min Chen. Bbn: Bilateral-branch network with cumulative learning for long-tailed visual recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9719–9728, 2020.

[46] Jun Li, Zichang Tan, Jun Wan, Zhen Lei, and Guodong Guo. Nested collaborative learning for long-tailed visual recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6949–6958, 2022.

[47] Yan Jin, Mengke Li, Yang Lu, Yiu-ming Cheung, and Hanzi Wang. Long-tailed visual recognition via self-heterogeneous integration with knowledge excavation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 23695–23704, 2023.

[48] Zhe Zhao, HaiBin Wen, Zikang Wang, Pengkun Wang, Fanfu Wang, Song Lai, Qingfu Zhang, and Yang Wang. Breaking long-tailed learning bottlenecks: A controllable paradigm with hypernetwork-generated diverse experts. *Advances in Neural Information Processing Systems*, 37:7493–7520, 2024.

[49] Zhe Zhao, Pengkun Wang, HaiBin Wen, JingXin Han, Zhenkun Wang, Qingfu Zhang, and Yang Wang. Balancing model efficiency and performance: Adaptive pruner for long-tailed data. In *Proceedings of the 42nd International Conference on Machine Learning*, 2025.

[50] Bingyi Kang, Saining Xie, Marcus Rohrbach, Zhicheng Yan, Albert Gordo, Jiashi Feng, and Yannis Kalantidis. Decoupling representation and classifier for long-tailed recognition. *arXiv preprint arXiv:1910.09217*, 2019.

[51] Mengke Li, HU Zhikai, Yang Lu, Weichao Lan, Yiu-ming Cheung, and Hui Huang. Feature fusion from head to tail for long-tailed visual recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 13581–13589, 2024.

[52] Pengkun Wang, Zhe Zhao, HaiBin Wen, Fanfu Wang, Binwu Wang, Qingfu Zhang, and Yang Wang. Llm-autoda: Large language model-driven automatic data augmentation for long-tailed problems. *Advances in Neural Information Processing Systems*, 37:64915–64941, 2024.

[53] Binwu Wang, Pengkun Wang, Wei Xu, Xu Wang, Yudong Zhang, Kun Wang, and Yang Wang. Kill two birds with one stone: Rethinking data augmentation for deep long-tailed learning. In *The 12th International Conference on Learning Representations*, 2024.

[54] Tianjiao Zhang, Huangjie Zheng, Jiangchao Yao, Xiangfeng Wang, Mingyuan Zhou, Ya Zhang, and Yanfeng Wang. Long-tailed diffusion models with oriented calibration. In *Proceedings of the 12th International Conference on Learning Representations*, 2024.

[55] Zhe Zhao, Pengkun Wang, HaiBin Wen, Wei Xu, Song Lai, Qingfu Zhang, and Yang Wang. Two fists, one heart: Multi-objective optimization based strategy fusion for long-tailed learning. In *Proceedings of the International Conference on Machine Learning*, 2024.

[56] Jiaan Luo, Feng Hong, Qiang Hu, Xiaofeng Cao, Feng Liu, and Jiangchao Yao. Long-tailed recognition with model rebalancing. *arXiv preprint arXiv:2510.08177*, 2025.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction clearly state the problem of the deadlock between NLL and LTL, the proposed bilevel optimization framework `Unlocker`, and its effectiveness on synthetic/real datasets, which aligns with the experimental results in Section 5.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The conclusion mentions limitations: reliance on uniform test distribution assumption, potential impurity of $\mathcal{D}_{\text{bc}}$ affecting τ optimization, and room for improving $\pi(y)$ estimation.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: Theorem 1 in Section 4.4 provides assumptions and a detailed proof via equivalence transformation to a convex problem with a closed-form solution.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We provide details on each parameter in the datasets, baselines, and implementations in our experimental setup.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The experimental code is available and the data sources are publicly accessible.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Details including dataset split (IR=10/50/100, noise type: joint/sym/t2h), baseline (NLL, LTNLL) and optimizer (SGD) are described in Section 5.1.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [NA]

   Justification: We do not report error bars or statistical tests.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Experiments used a GeForce RTX 3090 GPU with PyTorch 1.8.0.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We conduct the research project following NeurIPS Code of Ethics faithfully.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: The work does not seem to raise any (negative) societal impacts.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: Our method does not involve data or models that might lead to intended misuse.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [NA]

    Justification: Datasets (CIFAR, Clothing1M, WebVision) and baselines (DivideMix, UNICON) are cited with their original licenses.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: We provide pseudocode and parameters to facilitate reproduction.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: No human subjects research is conducted.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: LLM was not a component of this study

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

A Appendix / Supplemental Material

A.1 Detailed Training Process

The following is the process of our proposed method `Unlocker`, which leverages a bilevel optimization framework. Within each epoch, the inner optimization employs NLL methods with LA to train model, while the outer loop optimizes the learnable parameter τ to dynamically scale the strength of LA. This process adaptively tunes τ to integrate NLL methods and LA, iteratively disentangling the NLL-LTL deadlock and enhancing model robustness against long-tailed noisy label data.

Algorithm 1 Detailed training process of `Unlocker`

Input: Training data $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$, Number of epochs N_{epoch}, NLL algorithm $\mathcal{A}$, learning ratesη_τ, EMA decay β
Output: Trained model θ, Optimized τ
Initialize: Model Parameters θ_0, Learnable Parameter $\tau_0 \leftarrow 1.0$, Class Prior $\pi(y) \leftarrow$ EstimatePrior($\mathcal{D}$), adjustments $\alpha \leftarrow \tau_0 \cdot \log(\pi(y))$
for $epoch = T$ from 1 to N_{epoch} **do**
  // Inner Optimization
  for $k = 1$ to K **do**
    $\mathcal{D}_k \leftarrow$ GetBatch($\mathcal{D}, k$)
    Inference Mode: $\theta'_T \leftarrow \theta_T(\mathcal{D}_k) - \alpha_T$ ▷ Post-hoc logit adjustment
  end for
  $\mathcal{D}_{clean}, \mathcal{D}_{noisy}, \mathcal{D}_{bc} \leftarrow \mathcal{A}(\theta'_T)$ ▷ Select noisy labels according to $\mathcal{A}$
  $\mathcal{D}_{noisy} \leftarrow \mathcal{A}(\theta'_T, \mathcal{D}_{noisy})$ ▷ Correct noisy labels according to $\mathcal{A}$
  $\pi(y) \leftarrow$ EstimatePrior($\mathcal{D}_{clean}, \mathcal{D}_{noisy}$) ▷ Estimate class prior of $\mathcal{D}$
  for $k = 1$ to K **do**
    $\mathcal{D}_k \leftarrow$ GetBatch($\mathcal{D}, k$)
    $\theta'_T \leftarrow \begin{cases} \theta_T + \alpha_T, & \text{if } \mathcal{D}_k \subset \mathcal{D}_{\text{clean}} \\ \theta_T, & \text{otherwise} \end{cases}$ ▷ logit adjustment
    $\mathcal{L}_{NLL} \leftarrow \mathcal{A}(\theta'_T(x), \mathcal{D}_k)$ ▷ Compute model loss according to $\mathcal{A}$
    $\theta_{k+1} \leftarrow \theta_k - \eta \nabla_\theta \mathcal{L}_{NLL}(\theta_k)$ ▷ Update model
  end for
  // Outer Optimization
  for $k = 1$ to B **do**
    $\mathcal{D}_k \leftarrow$ GetBatch($\mathcal{D}_{bc}, k$)
    Inference Mode: $\theta_{T+1}(\mathcal{D}_k)$
    $\theta'_{T+1}(\mathcal{D}_k) \leftarrow \theta_{T+1}(\mathcal{D}_k) - \tau_k \cdot \log \pi(y)$ ▷ Post-hoc logit adjustment
    $\mathcal{L}_\tau \leftarrow KL\left(\mathbb{E}_{x \sim D_k} \text{softmax}(\theta'_{T+1}(\mathcal{D}_k)) \parallel U\right)$ ▷ Compute outer loss according to Eq. 6
    $\tau_{k+1} = \tau_k - \eta_\tau \nabla_\tau \mathcal{L}_\tau(\tau_k)$ ▷ Optimize τ
  end for
  $\alpha_{T+1} \leftarrow \tau_{T+1} \cdot \log(\pi(y))$ ▷ Compute adjustments
  $\alpha_{T+1} \leftarrow \beta \cdot \alpha_{T+1} + (1 - \beta) \cdot \alpha_T$ ▷ EMA update for adjustments
end for
return θ, τ

A.2 Theoretical Proof

Proof of Theorem 1. We prove the differentiability of the outer optimization with the following steps:
Step 1: Objective Function Expansion Substituting $\bar{P}(\tau_T) = \mathbb{E}_{x \sim \mathcal{D}_{bc}} P(y \mid x; \tau_T)$ into the KL

divergence $J(\tau) = KL\left(\bar{P}(\tau) \parallel U\right), U_c = \frac{1}{C}$, we get:

$$
\begin{aligned}
J(\tau) &= KL\left(\bar{P}(\tau) \parallel U\right) \\
&= \sum_{c=1}^{C} \bar{P}_c(\tau) \log \frac{\bar{P}_c(\tau)}{U_c} \\
&= \sum_{c=1}^{C} \left[\mathbb{E}_{x \sim \mathcal{D}_{bc}} P(y=c \mid x;\tau)\right] \log \frac{\mathbb{E}_{x \sim \mathcal{D}_{bc}} P(y=c \mid x;\tau)}{1/C} \\
&= \sum_{c=1}^{C} \left[\mathbb{E}_{x \sim \mathcal{D}_{bc}} P(y=c \mid x;\tau)\right] \left(\log \left[\mathbb{E}_{x \sim \mathcal{D}_{bc}} P(y=c \mid x;\tau)\right] + \log C\right) \\
&= \mathbb{E}_{x \sim \mathcal{D}_{bc}} \sum_{c=1}^{C} P(y=c \mid x;\tau) \log \left[\mathbb{E}_{x' \sim \mathcal{D}_{bc}} P(y=c \mid x';\tau)\right] + \log C \\
&= \mathbb{E}_{x \sim D_{\text{bc}}} \sum_{c=1}^{C} P_c \log P_c + \log C, \qquad (10)
\end{aligned}
$$

where $P_c = P(y=c \mid x;\tau)$ denotes the conditional probability that a sample x belongs to class c under the adjusting of τ.

Step 2: Gradient Derivation Taking the derivative of (10) with respect to τ using the chain rule:

$$
\nabla_\tau \mathcal{J}(\tau) = \mathbb{E}_{x \sim D_{\text{bc}}} \sum_{c=1}^{C} \frac{\partial P_c}{\partial \tau} \left(\log P_c + 1\right). \qquad (11)
$$

Given the component $\log P_c + 1 = \theta_c(x) - \tau \cdot \log \pi_c(y) - \log \sum_{k=1}^{C} e^{\theta_k(x) - \tau \cdot \log \pi_k} + 1$ is differentiable in τ, we focus on analyzing the differentiability of $\frac{\partial P_c}{\partial \tau}$. For the softmax function $P_c = \frac{e^{z_c}}{\sum_{k=1}^{C} e^{z_k}}$ with $z_c = \theta_c(x) - \tau \cdot \log \pi_c(y)$, we derive $\frac{\partial P_c}{\partial \tau}$ as follows:

$$
\begin{aligned}
\frac{\partial P_c}{\partial \tau} &= \frac{\partial}{\partial \tau} \left(\frac{e^{z_c}}{\sum_{k=1}^{C} e^{z_k}} \right) \\
&= \frac{\frac{\partial e^{z_c}}{\partial \tau} \cdot \sum_{k=1}^{C} e^{z_k} - e^{z_c} \cdot \frac{\partial}{\partial \tau} \sum_{k=1}^{C} e^{z_k}}{\left(\sum_{k=1}^{C} e^{z_k}\right)^2} \\
&= \frac{e^{z_c} \cdot \frac{\partial z_c}{\partial \tau} \cdot \sum_{k=1}^{C} e^{z_k} - e^{z_c} \cdot \sum_{k=1}^{C} e^{z_k} \cdot \frac{\partial z_k}{\partial \tau}}{\left(\sum_{k=1}^{C} e^{z_k}\right)^2} \\
&= \frac{e^{z_c}}{\sum_{k=1}^{C} e^{z_k}} \cdot \frac{\frac{\partial z_c}{\partial \tau} \cdot \sum_{k=1}^{C} e^{z_k} - \sum_{k=1}^{C} e^{z_k} \cdot \frac{\partial z_k}{\partial \tau}}{\sum_{k=1}^{C} e^{z_k}} \\
&= P_c \cdot \left(\frac{\partial z_c}{\partial \tau} - \sum_{k=1}^{C} \frac{e^{z_k}}{\sum_{k=1}^{C} e^{z_k}} \cdot \frac{\partial z_k}{\partial \tau} \right) \\
&= P_c \cdot \left(\frac{\partial z_c}{\partial \tau} - \sum_{k=1}^{C} P_k \cdot \frac{\partial z_k}{\partial \tau} \right). \qquad (12)
\end{aligned}
$$

Differentiate $z_c = \theta_c(x) - \tau \cdot \log \pi_c(y)$ with respect to τ, we have $\frac{\partial z_c}{\partial \tau} = -\log \pi_c(y)$ and $\frac{\partial z_k}{\partial \tau} = -\log \pi_k$. Substituting $\frac{\partial z_c}{\partial \tau}$ and $\frac{\partial z_k}{\partial \tau}$ into (12), we have:

$$\frac{\partial P_c}{\partial \tau} = P_c \cdot \left(-\log \pi_c(y) - \sum_{k=1}^{C} P_k \cdot (-\log \pi_k) \right)$$
$$= P_c \cdot \left(-\log \pi_c(y) + \sum_{k=1}^{C} P_k \log \pi_k \right). \tag{13}$$

Given the continuous differentiability of P_c and the the constancy of $\pi_c(y)$, the gradient $\frac{\partial P_c}{\partial \tau}$ exists and is differentiable. Substituting (13) back into (11), the gradient $\nabla_\tau J(\tau)$ is thus continuously differentiable, and simplified to:

$$\nabla_\tau J(\tau) = \mathbb{E}_{x \sim D_{\text{bc}}} \sum_{c=1}^{C} P_c \left(-\log \pi_c(y) + \sum_{k=1}^{C} P_k \log \pi_k \right) (\log P_c + 1). \tag{14}$$

Thus, by the differentiability of $\mathcal{J}(\tau)$, gradient descent guarantee convergence to a local minimum. □

Proof of Lemma 1. We establish the bounds of $J(\tau)$ as follows. We first expand the outer optimization function according to the definition of KL divergence:

$$J(\tau) = \text{KL}\left(\bar{P}(\tau) \| U\right) = \sum_{c=1}^{C} \bar{P}_c(\tau) \log \left(\frac{\bar{P}_c(\tau)}{U_c} \right)$$
$$= \sum_{c=1}^{C} \bar{P}_c(\tau) \log \left(\frac{\bar{P}_c(\tau)}{\frac{1}{C}} \right)$$
$$= \sum_{c=1}^{C} \bar{P}_c(\tau) \log \left(\bar{P}_c(\tau) \cdot C \right)$$
$$= \sum_{c=1}^{C} \bar{P}_c(\tau) \left[\log \bar{P}_c(\tau) + \log C \right]$$
$$= \sum_{c=1}^{C} \bar{P}_c(\tau) \log \bar{P}_c(\tau) + \log C \cdot \sum_{c=1}^{C} \bar{P}_c(\tau). \tag{15}$$

Note that $\sum_{c=1}^{C} \bar{P}_c(\tau) = 1$ (since $\bar{P}(\tau)$ is a probability distribution) in the second term. The first term is the negative entropy $\sum_{c=1}^{C} \bar{P}_c(\tau) \log \bar{P}_c(\tau) = -H\left(\bar{P}(\tau)\right)$ of $\bar{P}(\tau)$. Thus, we obtain:

$$J(\tau) = \log C - H\left(\bar{P}(\tau)\right). \tag{16}$$

Lower Bound The entropy $H\left(\bar{P}(\tau)\right)$ is maximized when $\bar{P}(\tau)$ is uniform, achieving $H\left(\bar{P}(\tau)\right) = \log C$. Therefore:

$$J(\tau) = \log C - H\left(\bar{P}(\tau)\right) \geq \log C - \log C = 0. \tag{17}$$

Equality holds if and only if $\bar{P}(\tau) = U$. Thus, the lower bound of $J(\tau)$ is 0.

Upper Bound The entropy $H\left(\bar{P}(\tau)\right) \geq 0$ for any probability distribution $\bar{P}(\tau)$. Therefore:

$$J(\tau) = \log C - H\left(\bar{P}(\tau)\right) \leq \log C - 0 = \log C. \tag{18}$$

Equality holds when $\bar{P}(\tau)$ is a degenerate distribution (i.e., one class has probability 1 and others 0).

Combining these results, we conclude:

$$0 \leq J(\tau) \leq \log C. \tag{19}$$

□

Proof of Theorem 2. We prove the existence of the global minimum point from the perspectives of continuity, lower boundedness and upper asymptotic convergence. Given that we have proven the continuity of $J(\tau)$ in Theorem1, we focus on proving the lower boundedness and upper asymptotic convergence of $J(\tau)$.

Lower Boundedness By Lemma 1, $J(\tau) = \log C - H(\bar{P}(\tau)) \geq 0$ for all $\tau \in \mathbb{R}$.

Upper Asymptotic Convergence As $\tau \to +\infty$, let $c^* = \arg\min_y \log \pi(y)$. Then, $-\tau \cdot \log \pi(c^*)$ dominates, making:

$$\lim_{\tau \to +\infty} P(y = c \mid x; \tau) = \lim_{\tau \to +\infty} \frac{e^{\theta_c(x) - \tau \cdot \log \pi(c)}}{\sum_{k=1}^{C} e^{\theta_k(x) - \tau \cdot \log \pi(k)}} = \begin{cases} 1, & \text{if} c = c^*, \\ 0, & \text{otherwise.} \end{cases} \tag{20}$$

Consequently, $\bar{P}(\tau) \to \delta_{c^*}$, a Dirac-delta distribution as follows:

$$\begin{aligned} \lim_{\tau \to +\infty} \bar{P}_c(\tau) &= \lim_{\tau \to +\infty} \mathbb{E}_{x \sim \mathcal{D}_{bc}} P(y = c \mid x; \tau) \\ &= \mathbb{E}_{x \sim \mathcal{D}_{bc}} \lim_{\tau \to +\infty} P(y = c \mid x; \tau) \\ &= \begin{cases} 1, & \text{if} c = c^*, \\ 0, & \text{otherwise.} \end{cases} \end{aligned} \tag{21}$$

Substituting $\bar{P}(\tau)$ into $H(\bar{P}(\tau))$, we obtain:

$$\begin{aligned} \lim_{\bar{P}(\tau) \to \delta_{c^*}} H(\bar{P}(\tau)) &= - \lim_{\bar{P}(\tau) \to \delta_{c^*}} \sum_{c=1}^{C} \bar{P}_c(\tau) \log \bar{P}_c(\tau) \\ &= - \sum_{c=1}^{C} \lim_{\bar{P}_c(\tau) \to \delta_{c^*}} \bar{P}_c(\tau) \log \bar{P}_c(\tau). \end{aligned} \tag{22}$$

For the components in 22, when $c = c^*$, $\lim_{\bar{P}_{c^*}(\tau) \to 1} \bar{P}_{c^*}(\tau) \log \bar{P}_{c^*}(\tau) = 0$. When $c \neq c^*$, $\lim_{\tau \to +\infty} \bar{P}_c(\tau) = 0$, $\lim_{\bar{P}_c(\tau) \to 0} \bar{P}_c(\tau) \log \bar{P}_c(\tau) = 0$. Therefore, the addition result of the components is $\lim_{\bar{P}(\tau) \to \delta_{c^*}} H(\bar{P}(\tau)) = 0$. Substituting $H(\bar{P}(\tau)) \to 0$ into $J(\tau)$:

$$\lim_{\tau \to +\infty} J(\tau) = \log C - H(\bar{P}(\tau)) = \log C - 0 = \log C. \tag{23}$$

As $\tau \to -\infty$, define $c^* = \arg\max_y \log \pi(y)$. Similarly, we have:

$$\lim_{\tau \to -\infty} J(\tau) = \log C. \tag{24}$$

□

Proof of Proposition 1. The KL divergence $\mathrm{KL}(\bar{P}(\tau) \parallel U)$ is minimized when $\bar{P}(\tau) = U$, as the KL divergence is non-negative and zero at exact matching. This requires solving:

$$\mathbb{E}_{x \sim \mathcal{D}_{\text{bc}}} P(x, \theta; \tau) = \mathbb{E}_{x \sim \mathcal{D}_{\text{bc}}} \text{softmax}\left(\theta(x) - \tau \cdot \log \pi(y)\right) = U, \quad U_c = \frac{1}{C} \; (\forall c \in \{1, \ldots, C\}). \tag{25}$$

Leveraging the normalized weighted geometric mean approximation [32], 25 can be approximated as:

$$\begin{aligned} \mathbb{E}_{x \sim \mathcal{D}_{\text{bc}}} \text{softmax}\left(\theta(x) - \tau \cdot \log \pi(y)\right) &\approx \text{softmax}(\mathbb{E}_{x \sim \mathcal{D}_{\text{bc}}}\left[\theta(x) - \tau \cdot \log \pi(y)\right]) \\ &= \text{softmax}(\mathbb{E}_{x \sim \mathcal{D}_{\text{bc}}}[\theta(x)] - \tau \cdot \log \pi(y)) \qquad \approx U. \end{aligned} \tag{26}$$

Considering the i-th component in 26, we have:

$$\text{softmax}(\mathbb{E}_{x \sim \mathcal{D}_{\text{bc}}}[\theta_c(x)] - \tau \cdot \log \pi(c)) \approx \frac{1}{C}. \tag{27}$$

Taking the log of both sides, we obtain:

$$\mathbb{E}_{x \sim \mathcal{D}_{\text{bc}}}[\theta_c(x)] - \tau \cdot \log \pi(c) = \log K, \quad \forall c, \tag{28}$$

where $K = \frac{\sum_{j=1}^{C}(\exp\left(\mathbb{E}_{x\sim\mathcal{D}_{\text{bc}}}[\theta_j(x)]-\tau\log\pi_j(y)\right))}{C}$ denotes a constant. Choosing class $c = 1$ as a reference, we subtract the equation for $c = 1$ from that of class c:

$$\mathbb{E}_{x\sim\mathcal{D}_{\text{bc}}}[\theta_c(x)] - \mathbb{E}_{x\sim\mathcal{D}_{\text{bc}}}[\theta_1(x)] = \tau\cdot(\log\pi(c) - \log\pi(1)). \tag{29}$$

Define the residual 29 $r_c = \mathbb{E}_{x\sim\mathcal{D}_{\text{bc}}}[\theta_c(x)] - \mathbb{E}_{x\sim\mathcal{D}_{\text{bc}}}[\theta_1(x)] - \tau\cdot(\log\pi(c) - \log\pi(1))$, the least-squares objective is:

$$\min_{\tau}\sum_{c=1}^{C} r_c^2 = \min_{\tau}\sum_{c=1}^{C}\left(\mathbb{E}_{x\sim\mathcal{D}_{\text{bc}}}[\theta_c(x)] - \mathbb{E}_{x\sim\mathcal{D}_{\text{bc}}}[\theta_1(x)] - \tau\left(\log\pi(c) - \log\pi(1)\right)\right)^2. \tag{30}$$

To derive the closed-form solution for τ, we define the objective function $f(\tau) = \sum_{c=1}^{C}(\mu_c - \mu_1 - \tau(\log\pi(c) - \log\pi(1)))^2$ with $\mu_c = \mathbb{E}_{x\sim\mathcal{D}_{\text{bc}}}[\theta_c(x)], \forall c$. Using the chain rule, we differentiate $f(\tau)$ with respect to τ:

$$\begin{aligned}
\frac{\partial f(\tau)}{\partial\tau} &= \sum_{c=1}^{C} 2(\mu_c - \mu_1 - \tau(\log\pi(c) - \log\pi(1)))\cdot\frac{\partial}{\partial\tau}[\mu_c - \mu_1 - \tau(\log\pi(c) - \log\pi(1))] \\
&= \sum_{c=1}^{C} 2(\mu_c - \mu_1 - \tau(\log\pi(c) - \log\pi(1)))\cdot(-(\log\pi(c) - \log\pi(1))) \\
&= -2\sum_{c=1}^{C}(\mu_c - \mu_1 - \tau(\log\pi(c) - \log\pi(1)))(\log\pi(c) - \log\pi(1)).
\end{aligned} \tag{31}$$

Setting the Derivative to Zero, we obtain:

$$\begin{aligned}
-2\sum_{c=1}^{C}(\mu_c - \mu_1 - \tau(\log\pi(c) - \log\pi(1)))(\log\pi(c) - \log\pi(1)) = 0 \\
\sum_{c=1}^{C}(\mu_c - \mu_1)(\log\pi(c) - \log\pi(1)) - \tau\sum_{c=1}^{C}(\log\pi(c) - \log\pi(1))^2 = 0.
\end{aligned} \tag{32}$$

Rearranging terms to solve for τ:

$$\begin{aligned}
\tau\sum_{c=1}^{C}(\log\pi(c) - \log\pi(1))^2 &= \sum_{c=1}^{C}(\mu_c - \mu_1)(\log\pi(c) - \log\pi(1)) \\
\tau^{\text{LS}} &= \frac{\sum_{c=1}^{C}(\mu_c - \mu_1)(\log\pi(c) - \log\pi(1))}{\sum_{c=1}^{C}(\log\pi(c) - \log\pi(1))^2},
\end{aligned} \tag{33}$$

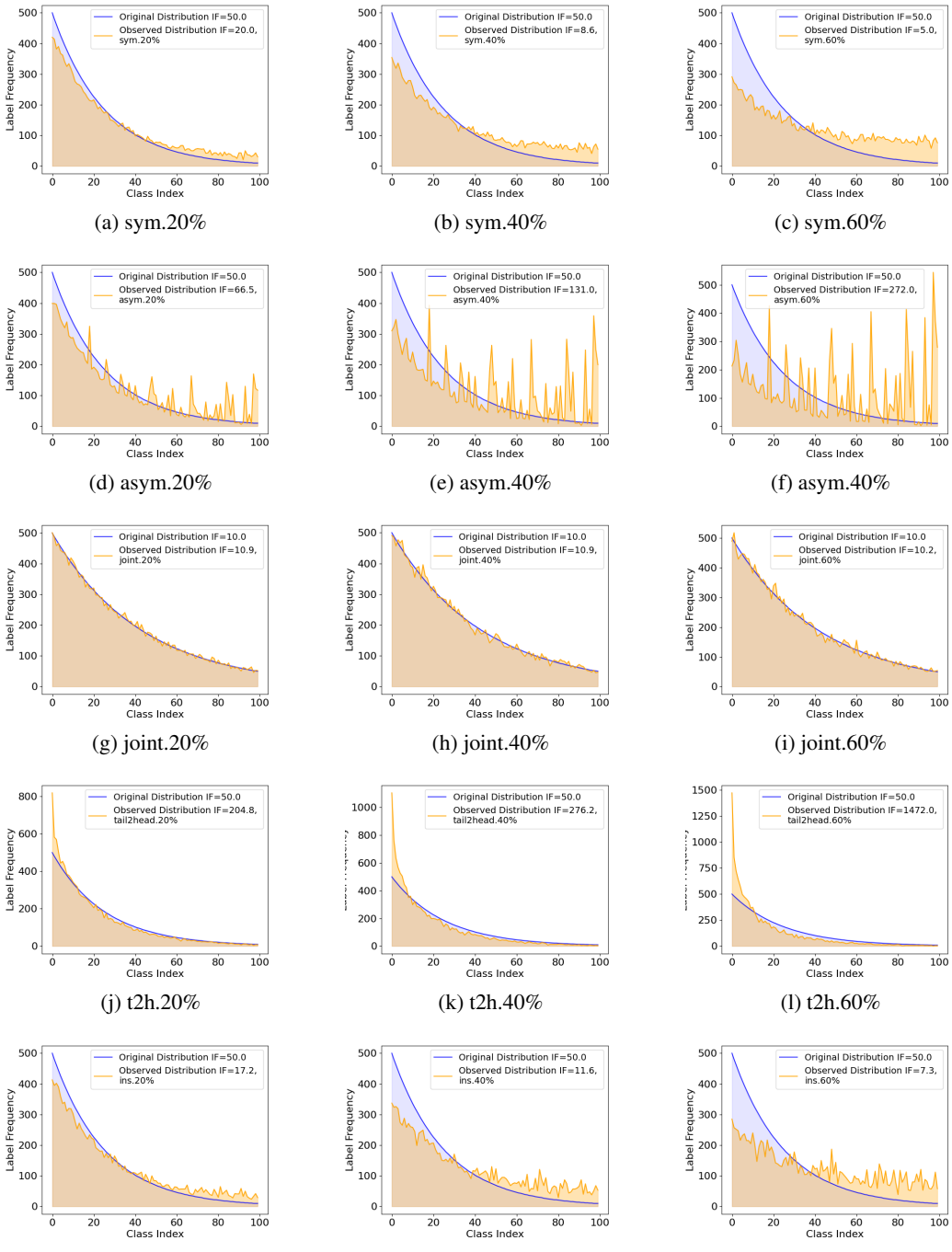
where τ^{LS} is the final closed-form solution for τ.

□

A.3 Methods of Label Noise Addition

Sym. Symmetric noise (Sym) means that for each sample label, we randomly replace it with one of the other classes with a fixed probability η. For a C-class classification task, given a noise rate η, the original label y is uniformly changed to other classes except y with the probability η. Specific noise transition matrix: elements on the diagonal are $1-\eta$, elements on the off-diagonal are $\eta/(C-1)$.

Asym. Asymmetric noise (Asym) simulates the real-world label noise structure. It selects "easily confused" class pairs (such as dog ↔ wolf) and specifies the transition probability, while the rest remain unchanged. Labels are only replaced between similar classes, and are not randomly mislabeled as other classes. The process of label flipping is related to the quantity of each class. With the noise rate denoted as η, we establish the following definitions:$T_{ij}(x) = P[\widetilde{Y} = j|Y = i, x] = 1-\eta$ when $i = j$. Conversely, $T_{ij}(x) = P[\widetilde{Y} = j|Y = i, x] = \frac{n_j}{n-n_i}\eta$. Here, Y and $\widetilde{Y}$ represent the random variables for clean labels and noisy labels, respectively.



(a) sym.20% (b) sym.40% (c) sym.60%

(d) asym.20% (e) asym.40% (f) asym.40%

(g) joint.20% (h) joint.40% (i) joint.60%

(j) t2h.20% (k) t2h.40% (l) t2h.60%

(m) ins.20% (n) ins.40% (o) ins.60%

Figure 6: Changes in the imbalance ratio (IR) of observed distributions (orange line) under different noise types with varying noise ratios, in comparison to the fixed IR of the true distritbuion (blue line). Symmetric (sym) and instance-dependent (ins) noise alleviate IR, while asymmetric noise (asym) may reverse the long-tail pattern. Joint noise preserves IR at low imbalance. Tail-to-head noise (t2h) exacerbates IR compared to the original distribution.

Joint. The Joint noise [5] label is generated by the noise transfer matrix, which represents the probability of the clean label flipping to the noise label. Let Y represent the clean label variable, $\tilde{Y}$ represent the noise label, X represent the instance feature, and the transfer matrix $T(X = x)$ is

defined as $T_{ij}(X) = \mathbb{P}(\tilde{Y} = j|Y = i, X = x)$. Specifically, given the noise ratio $\eta \in [0, 1]$, it is defined as follows:

$$T_{ij}(X) = \mathbb{P}\left[\tilde{Y} = j \mid Y = i, X = x\right] = \begin{cases} 1 - \eta & i = j \\ \frac{\eta N_j}{N - N_i} & otherwise \end{cases} \tag{34}$$

Here, N denote the total number of training examples and N_j is the frequency of class j. It combines the class prior information in the dataset to set the transition probability, which is more in line with the situation in real-world scenarios where samples are easily mislabeled as frequent classes.

T2H. Tail-to-Head noise (T2H) [33] refers to the phenomenon that in the long-tail data distribution, the tail class samples tend to be mislabeled as the head class samples. The generation of T2H noise mainly includes two steps: separating transferable and non-transferable samples; randomly selecting tail class samples and assigning head class labels to them. We define a transition matrix $T \in \mathbb{R}^{C \times C}$, where each element $T_{t,h} = P(y_i = h|\tilde{y}_i = t)$ represents the probability that an instance with the true class t is mis-labeled as class h. In the context of T2H noise, the samples from the tail class t have a relatively high probability of being mis-labeled as the head class h, i.e., $T_{t,h} > T_{t,t'}$ (where t' denotes a rarer tail class with fewer samples than t). Meanwhile, the probability that a sample from the head class h is mislabeled as the tail class t is very low, $T_{h,t} \approx 0$. Through this transition matrix, the T2H noise is quantitatively defined from a probabilistic perspective.

IDN. Instance-dependent noise (IDN) [13] is closely related to the characteristics of each instance and its class label. It is generated by setting a random noise rate for each instance, which follows a truncated Gaussian distribution, and the noise rate of each class is also randomly set. In this noise model, the probability of label flipping varies for each specific instance. Taking the CIFAR-10/100 datasets as examples, when generating instance-dependent noise, for a given clean sample set and a set noise rate η, a random noise rate is set for each instance one by one according to the truncated gaussian distribution, thereby realizing the generation of instance-dependent noise.

A.4 Analysis of the Effects of Different Noise Additions Methods on Long-tailed distribution

We systematically investigate the impact of existing noise addition methods on the imbalance ratio (IR) of true distribution when applied to clean long-tailed datasets. The results are presented in the Figure 6. **Sym** noise alleviates the long-tail problem because the samples of the head category are evenly distributed to other classes, indirectly balancing the data distribution. Baesd on this finding, we choose symmetric noise to simulate *relieve* scenario. **Asym** noise induces a reverse long-tail which means that some tail classes surge in count because head-class samples are frequently mislabeled as them. However, the reverse long-tail scenario is impractical. Therefore, we opt not to use asymmetric noise in our experimental setup. **Joint** noise usually refers to the existence of some correlation between the noisy label and the long-tailed distribution. In some cases like long-tailed distribution with low IR, joint noise maintains the original long-tailed structure, keeping the imbalance ratio unchanged. However, in other cases, it may either exacerbate or alleviate the IR, which is uncontrollable. Thus, we only select joint noise to construct *consistent* scenarios for long-tailed distribution with low IR. **T2H** noise significantly aggravates the long-tail problem, and the number of tail-class samples is further reduced, causing the model to be biased towards the head classes. We choose t2h noise to construct simulated *aggravate* scenarios. **IDN** noise, like symmetric noise, alleviates IR after adding noise.

A.5 Results on Simulated Scenarios Based on CIFAR-10

We conduct experiments using the NLL method DPC on CIFAR-10 to evaluate the performance of our method under scenarios with varying imbalance ratios (IR=10, 50, 100), noise types (joint, sym, t2h), and noise rates (η=40%, 60%). As shown in Table 5, DPC combined with `Unlocker` (DPC+`Unlocker`) achieves SOTA test accuracies across all the scenarios, outperforming baselines DPC and DPC with direct LA integration. In the consistent scenarios, DPC+`Unlocker` achieves accuracies of 93.71% and 90.93%, yielding improvements of 0.18% and 5.30% over the original DPC (93.53%, 85.63%) respectively In the relieve scenarios where tail-class clean sample selection is particularly challenging, DPC+`Unlocker` reaches significant improvements over DPC, ranging from 11.06% to 19.87%, validating its effectiveness in mitigating long-tail induced model bias and restoring the original NLL performance. Under aggravate scenarios, DPC+`Unlocker` maintains stable gains

Table 5: Test accuracy (%) comparison of methods on the CIFAR-10 dataset under varying imbalance ratios (IR), noise types and noise rates η, involving three scenarios of true distribution shifts. Green numbers indicate improvements over the original NLL method. Boldface represents the best performance in each case.

dataset	CIFAR-10									
types	consistent		relieve				aggravate			
IR	10		50		100		10		50	
η	joint 40%	joint 60%	sym 40%	sym 60%	sym 40%	sym 60%	t2h 40%	t2h 60%	t2h 40%	t2h 60%
DPC [14]	93.53	85.63	78.33	57.90	60.07	45.39	93.13	72.10	74.29	70.81
DPC+LA (post-hoc)	92.74	85.31	78.93	58.05	62.40	40.08	83.82	76.21	75.93	61.12
DPC+LA	88.31	81.53	75.60	45.77	30.94	36.50	91.27	83.11	61.75	48.40
DPC+LA+`Unlocker`	**93.71**	**90.93**	**89.39**	**72.86**	**73.15**	**65.26**	**93.76**	**92.05**	**88.67**	**83.35**
vs. DPC	↑0.18	↑5.30	↑11.06	↑14.96	↑13.08	↑19.87	↑0.63	↑19.95	↑14.38	↑12.54

over of 1.04% to 19.95% DPC. These results highlight efficacy of `Unlocker` in disentangling the NLL-LTL deadlock and enhancing model robustness against long-tailed noisy label data.

A.6 Related Work

Noisy Label Learning (NL). Noisy label learning focuses on tackling the challenge of inaccurate supervised label in datasets. It mainly evolves along two directions: noisy label detection and correction, and robust noise label learning. The former mainstream typically uses a two-step process: selecting noisy labels via metrics such as loss or divergence, and then correcting them through techniques like semi-supervision learning [10, 11, 12, 13, 14]. By directly selecting and filtering out noisy labels, these methods have demonstrated efficiency in both experimental and real-world scenarios. Robust noisy label learning mitigates noise by adjusting loss functions via regularization or noisy transiton matrix [34, 35, 36, 37, 38] to disregard or reduce noise impact.

Long-tailed Learning (LT). Long-tailed learning is aimed to improve the accuracy of tail classes caused by skewed datasets distributions. Re-sampling is a classic method, which directly balances the distribution through reducing samples of head or augmenting samples of tail [39, 40, 41]. Re-weighting enhances the focus of the model on tail classes by adjusting the sample weights in the loss function [42, 23, 43, 44]. Ensembling learning improves model performance by aggregating multiple networks within a multi-expert framework [45, 46, 47, 48, 49]. The two-stage decoupling strategy achieves a rebalancing of decision boundaries through the fine-tuning of classifiers [50, 18, 51]. Logit Adjustment (LA) corrects biased logits by adding an offset term to the model's logit [15, 16, 17, 18, 19]. Extensive empirical studies have substantiated the efficacy of the LA. Moreover, data augmentation is an an effective way to alleviate the scarcity of tail classes by generating tail samples [52, 53, 54]. Besides, recent work such as strategy fusion tailored for multi-objective optimization (MOO) [55] and model parameter space rebalancing [56] also shows promise in balancing model.