

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 rates η_τ , EMA decay β
Output: Trained model θ , Optimized τ
Initialize: Model Parameters θ_0 , Learnable Parameter $\tau_0 \leftarrow 1.0$, Class Prior $\pi(y) \leftarrow \text{EstimatePrior}(\mathcal{D})$, 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**
 $\mathcal{D}_k \leftarrow \text{GetBatch}(\mathcal{D}, k)$
 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}
 $\pi(y) \leftarrow \text{EstimatePrior}(\mathcal{D}_{clean}, \mathcal{D}_{noisy})$ ▷ Estimate class prior of \mathcal{D}
 for $k = 1$ to K **do**
 $\mathcal{D}_k \leftarrow \text{GetBatch}(\mathcal{D}, k)$
 $\theta'_T \leftarrow \begin{cases} \theta_T + \alpha_T, & \text{if } \mathcal{D}_k \subset \mathcal{D}_{clean} \\ \theta_T, & \text{otherwise} \end{cases}$ ▷ 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**
 $\mathcal{D}_k \leftarrow \text{GetBatch}(\mathcal{D}_{bc}, k)$
 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
 $\mathcal{L}_\tau \leftarrow KL(\mathbb{E}_{x \sim \mathcal{D}_k} \text{softmax}(\theta'_{T+1}(\mathcal{D}_k)) \parallel U)$ ▷ 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 $J(\tau) = KL(\bar{P}(\tau) \parallel U)$, $U_c = \frac{1}{C}$, we get:

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

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:

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

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:

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

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:

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

Given the continuous differentiability of P_c and 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:

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

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

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:

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

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 $\sum_{c=1}^C \bar{P}_c(\tau) \log \bar{P}_c(\tau) = -H(\bar{P}(\tau))$ of $\bar{P}(\tau)$. Thus, we obtain:

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

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

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

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

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

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

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:

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

\square

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 Theorem 1, 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 $\tau \rightarrow +\infty$, let $c^* = \arg \min_y \log \pi(y)$. Then, $-\tau \cdot \log \pi(c^*)$ dominates, making:

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

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

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

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

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

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

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

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

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

□

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

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

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

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

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

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

Taking the log of both sides, we obtain:

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

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

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

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

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

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 $\mu_c = \mathbb{E}_{x \sim \mathcal{D}_{bc}}[\theta_c(x)]$, $\forall c$. Using the chain rule, we differentiate $f(\tau)$ with respect to τ :

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

Setting the Derivative to Zero, we obtain:

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

Rearranging terms to solve for τ :

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

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 \leftrightarrow 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: $T_{ij}(x) = P[\tilde{Y} = j | Y = i, x] = 1 - \eta$ when $i = j$. Conversely, $T_{ij}(x) = P[\tilde{Y} = j | Y = i, x] = \frac{n_j}{n - n_i} \eta$. Here, Y and \tilde{Y} represent the random variables for clean labels and noisy labels, respectively.

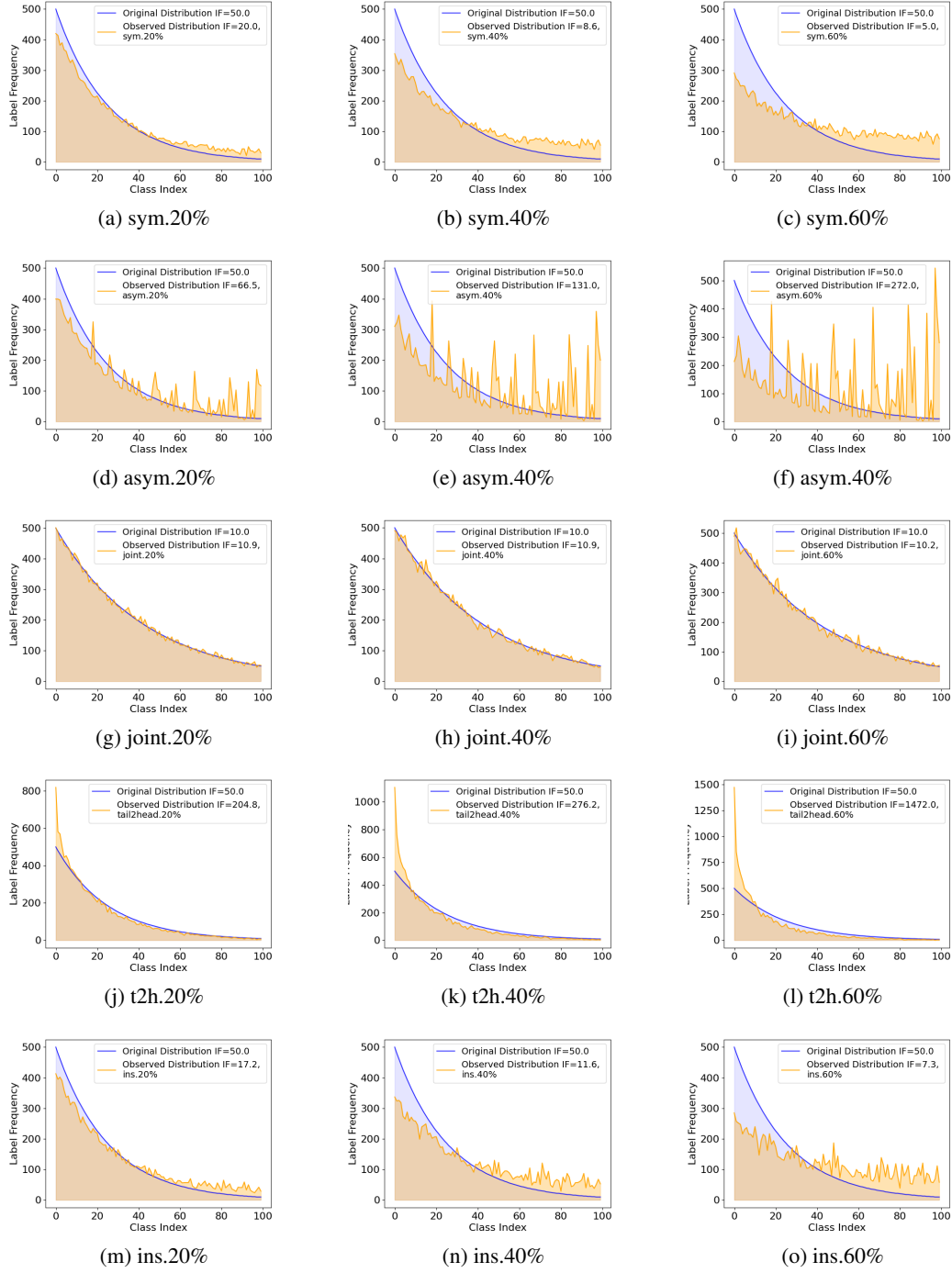


Figure 6: Changes in the imbalance ratio (IR) of observed distributions (orange line) under different noise types with varying noise ratios, in comparison to the fixed IR of the true distribution (blue line). Symmetric (sym) and instance-dependent (ins) noise alleviate IR, while asymmetric noise (asym) 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:

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

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. Based 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 ($\eta=40\%$, 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	$\uparrow 0.18$	$\uparrow 5.30$	$\uparrow 11.06$	$\uparrow 14.96$	$\uparrow 13.08$	$\uparrow 19.87$	$\uparrow 0.63$	$\uparrow 19.95$	$\uparrow 14.38$	$\uparrow 12.54$

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 transition 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 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.