Towards Calibrated Model for Long-Tailed Visual Recognition from Prior Perspective

Zhengzhuo Xu[1*], **Zenghao Chai**[1*], **Chun Yuan**[1,2†]
[1]Shenzhen International Graduate School, Tsinghua University
[2]Peng Cheng Laboratory
xzz20@mails.tsinghua.edu.cn, zenghaochai@gmail.com,
yuanc@sz.tsinghua.edu.cn

Abstract

Real-world data universally confronts a severe class-imbalance problem and exhibits a *long-tailed* distribution, i.e., most labels are associated with limited instances. The naïve models supervised by such datasets would prefer dominant labels, encounter a serious generalization challenge and become poorly calibrated. We propose two novel methods from the *prior* perspective to alleviate this dilemma. First, we deduce a balance-oriented data augmentation named Uniform Mixup (UniMix) to promote *mixup* in long-tailed scenarios, which adopts advanced mixing factor and sampler in favor of the minority. Second, motivated by the Bayesian theory, we figure out the Bayes Bias (Bayias), an inherent bias caused by the inconsistency of *prior*, and compensate it as a modification on standard cross-entropy loss. We further prove that both the proposed methods ensure the classification *calibration* theoretically and empirically. Extensive experiments verify that our strategies contribute to a better-calibrated model and their combination achieves state-of-the-art performance on CIFAR-LT, ImageNet-LT, and iNaturalist 2018.

1 Introduction

Balanced and large-scaled datasets [53, 42] have promoted deep neural networks to achieve remarkable success in many visual tasks [25, 52, 23]. However, real-world data typically exhibits a *long-tailed* (LT) distribution [37, 45, 28, 19], and collecting a minority category (**tail**) sample always leads to more occurrences of common classes (**head**) [57, 29], resulting in most labels associated with limited instances. The paucity of samples may cause insufficient feature learning on the tail classes [69, 14, 36, 45], and such data imbalance will bias the model towards dominant labels [56, 57, 40]. Hence, the generalization of minority categories is an enormous challenge.

The intuitive approaches such as directly over-sampling the tail [11, 6, 48, 54, 7] or under-sampling the head [20, 6, 22] will cause serious robustness problems. *mixup* [64] and its extensions [59, 63, 12] are effective feature improvement methods and contribute to a well-calibrated model in balanced datasets [58, 65], i.e., *the predicted confidence indicates actual accuracy likelihood* [18, 58]. However, *mixup* is inadequately calibrated in an imbalanced LT scenario (Fig.1). In this paper, we raise a conception called ξ-*Aug* to analyze *mixup* and figure out that it tends to generate more head-head pairs, resulting in unsatisfactory generalization of the tail. Therefore, we propose Uniform Mixup (UniMix), which adopts a tail-favored *mixing factor* related to label *prior* and a *inverse sampling strategy* to encourage more head-tail pairs occurrence for better generalization and *calibration*.

[*]Equal contribution.
[†]Corresponding author.

35th Conference on Neural Information Processing Systems (NeurIPS 2021).

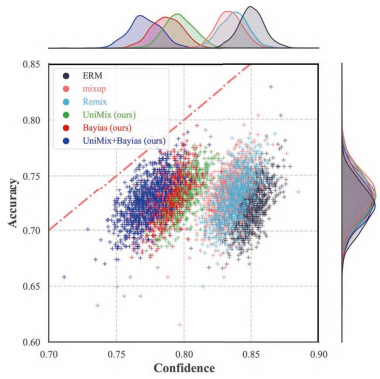


Figure 1: Joint density plots of accuracy vs. confidence to measure the *calibration* of classifiers on CIFAR-100-LT-100 during training. A well-calibrated classifier's density will lay around the red dot line $y = x$, indicating prediction score reflects the actual likelihood of accuracy. *mixup* manages to regularize classifier on balanced datasets. However, both *mixup* and its extensions tend to be overconfident in LT scenarios. Our UniMix reconstructs a more balanced dataset and Bayias-compensated CE erases *prior* bias to ensure better *calibration*. Without loss of accuracy, either of proposed methods trains the same classifier more calibrated and their combination achieves the best. How to measure *calibration* and more visualization results are available in Appendix D.2.

Previous works adjust the logits *weight* [34, 14, 56, 62] or *margin* [8, 47] on standard *Softmax* cross-entropy (CE) loss to tackle the bias towards dominant labels. We analyze the inconstancy of label *prior*, which varies in LT train set and balanced test set, and pinpoint an inherent bias named Bayes Bias (Bayias). Based on the Bayesian theory, the *posterior* is proportional to *prior* times *likelihood*. Hence, it's necessary to adjust the *posterior* on train set by compensating different *prior* for each class, which can serve as an additional *margin* on CE. We further demonstrate that the Bayias-compensated CE ensures classification *calibration* and propose a unified learning manner to combine Bayias with UniMix towards a better-calibrated model (see in Fig.1). Furthermore, we suggest that bad calibrated approaches are counterproductive with each other, which provides a heuristic way to analyze the combined results of different feature improvement and loss modification methods (see in Tab.4).

In summary, our contributions are: 1) We raise the concept of ξ-*Aug* to theoretically explain the reason of *mixup*'s miscalibration in LT scenarios and propose Unimix (Sec.3.1) composed of novel mixing and sampling strategies to construct a more class-balanced virtual dataset. 2) We propose the Bayias (Sec.3.2) to compensate the bias incurred by different label *prior*, which can be unified with UniMix by a training manner for better classification *calibration*. 3) We conduct sufficient experiments to demonstrate that our method trains a well-calibrated model and achieves state-of-the-art results on CIFAR-10-LT, CIFAR-100-LT, ImageNet-LT, and iNaturalist 2018.

2 Analysis of *mixup*

The core of supervised image classification is to find a θ parameterized mapping $\mathcal{F}_\theta : X \in \mathbb{R}^{c\times h\times w} \mapsto Y \in \mathbb{R}^{C\times 1}$ to estimate the empirical Dirac delta distribution $\mathbb{P}_\delta(x,y) = \frac{1}{N}\sum_{i=1}^{N}\delta(x_i,y_i)$ of N instances $x \in \mathcal{X}$ and labels $y \in \mathcal{Y}$. The learning progress by minimizing Eq.1 is known as Empirical Risk Minimization (ERM), where $\mathcal{L}(Y = y_i, \mathcal{F}_\theta(X = x_i))$ is x_i's conditional risk.

$$R_\delta(\mathcal{F}_\theta) = \int_{x\in\mathcal{X}} \mathcal{L}\left(Y = y, \mathcal{F}_\theta(X = x)\right) d\mathbb{P}_\delta(x,y) = \frac{1}{N}\sum\nolimits_{i=1}^{N} \mathcal{L}\left(Y = y_i, \mathcal{F}_\theta(X = x_i)\right) \quad (1)$$

To overcome the over-fitting caused by insufficient training of N samples, *mixup* utilizes Eq.2 to extend the feature space to its vicinity based on Vicinal Risk Minimization (VRM) [10].

$$\widetilde{x} = \xi \cdot x_i + (1-\xi)\cdot x_j \qquad \widetilde{y} = \xi \cdot y_i + (1-\xi)\cdot y_j \quad (2)$$

where $\xi \sim Beta(\alpha,\alpha), \alpha \in [0,1]$, the sample pair $(x_i,y_i),(x_j,y_j)$ is drawn from training dataset $\mathcal{D}_{train}$ randomly. Hence, Eq.2 converts $\mathbb{P}_\delta(X,Y)$ into empirical *vicinal distribution* $\mathbb{P}_\nu(\widetilde{x},\widetilde{y}) = \frac{1}{N}\sum_{i=1}^{N}\nu(\widetilde{x},\widetilde{y}|x_i,y_i)$, where $\nu(\cdot)$ describes the manner of finding virtual pairs $(\widetilde{x},\widetilde{y})$ in the vicinity of arbitrary sample (x_i,y_i). Then, we construct a new dataset $\mathcal{D}_\nu := \{(\widetilde{x}_k,\widetilde{y}_k)\}_{k=1}^{M}$ via Eq.2 and minimize the empirical vicinal risk by Vicinal Risk Minimization (VRM):

$$R_\nu(\mathcal{F}_\theta) = \int_{\widetilde{x}\in\widetilde{\mathcal{X}}} \mathcal{L}\left(\widetilde{Y} = \widetilde{y}, \mathcal{F}_\theta(\widetilde{X} = \widetilde{x})\right) d\mathbb{P}_\nu(\widetilde{x},\widetilde{y}) = \frac{1}{M}\sum\nolimits_{i=1}^{M} \mathcal{L}\left(\widetilde{Y} = \widetilde{y}_i, \mathcal{F}_\theta(\widetilde{X} = \widetilde{x}_i)\right) \quad (3)$$

mixup is proven to be effective on balanced dataset due to its improvement of *calibration* [58, 18], but it is unsatisfactory in LT scenarios (see in Tab.4). In Fig.1, *mixup* fails to train a calibrated model,

which surpasses baseline (ERM) a little in accuracy and seldom contributes to *calibration* (far from $y = x$). To analyze the insufficiency of *mixup*, the definition of ξ-*Aug* is raised.

Definition 1 *ξ-Aug. The virtual sample $(\widetilde{x}_{i,j}, \widetilde{y}_{i,j})$ generated by Eq.2 with mixing factor ξ is defined as a ξ-Aug sample, which is a robust sample of class y_i (class y_j) iff $\xi \geq 0.5 (\xi < 0.5)$ that contributes to class y_i (class y_j) in model's feature learning.*

In LT scenarios, we reasonably assume the instance number n of each class is *exponential* with parameter λ [14] if indices are descending sorted by n_{y_i}, where $y_i \in [1, C]$ and C is the total class number. Generally, the imbalance factor is defined as $\rho = n_{y_1}/n_{y_C}$ to measure how skewed the LT dataset is. It is easy to draw $\lambda = \ln \rho/(C-1)$. Hence, we can describe the LT dataset as Eq.4:

$$\mathbb{P}(Y = y_i) = \frac{\iint_{x_i \in \mathcal{X}, y_j \in \mathcal{Y}} \mathbb{1}(X = x_i, Y = y_i) dx_i dy_j}{\iint_{x_i \in \mathcal{X}, y_j \in \mathcal{Y}} \mathbb{1}(X = x_i, Y = y_j) dx_i dy_j} = \frac{\lambda}{e^{-\lambda} - e^{-\lambda C}} e^{-\lambda y_i}, y_i \in [1, C] \quad (4)$$

Then, we derive the following corollary to illustrate the limitation of naïve *mixup* strategy.

Corollary 1 *When $\boldsymbol{\xi \sim Beta(\alpha, \alpha)}, \alpha \in [0, 1]$, the newly mixed dataset $\mathcal{D}_\nu$ composed of ξ-Aug samples $(\widetilde{x}_{i,j}, \widetilde{y}_{i,j})$ follows the same long-tailed distribution as the origin dataset $\mathcal{D}_{train}$, where (x_i, y_i) and (x_j, y_j) are **randomly** sampled from $\mathcal{D}_{train}$. (See detail derivation in Appendix A.2.)*

$$\begin{aligned} \mathbb{P}_{mixup}(Y^* = y_i) &= \mathbb{P}^2(Y = y_i) + \mathbb{P}(Y = y_i) \iint_{y_i \neq y_j} Beta(\alpha, \alpha) \mathbb{P}(Y = y_j) d\xi dy_j \\ &= \frac{\lambda}{e^{-\lambda} - e^{-\lambda C}} e^{-\lambda y_i}, y_i \in [1, C] \end{aligned} \quad (5)$$

In *mixup*, the probability of any $(\widetilde{x}_{i,j}, \widetilde{y}_{i,j})$ belongs to class y_i or class y_j is strictly determined by ξ and $\mathbb{E}(\xi) \equiv 0.5$. Furthermore, both (x_i, y_i) and (x_j, y_j) are randomly sampled and concentrated on the head instead of tail, resulting in that the head classes get more ξ-*Aug* samples than the tail ones.

3 Methodology

3.1 UniMix: balance-oriented feature improvement

mixup and its extensions tend to generate head-majority pseudo data, which leads to the deficiency on the tail feature learning and results in a bad-calibrated model. To obtain a more balanced dataset $\mathcal{D}_\nu$, we propose the UniMix Factor $\xi^*_{i,j}$ related to the *prior* probability of each category and a novel UniMix Sampler to obtain sample pairs. Our motivation is to generate comparable ξ-*Aug* samples of each class for better generalization and *calibration*.

UniMix Factor. Specifically, the *prior* in imbalanced train set and balanced test set of class y_i is defined as $\mathbb{P}_{train}(Y = y_i) \triangleq \pi_{y_i}$, and $\mathbb{P}_{test}(Y = y_i) \equiv 1/C$, respectively. We design the UniMix Factor $\xi^*_{i,j}$ for each virtual sample $\widetilde{x}_{i,j}$ instead of a fixed ξ in *mixup*. Consider adjusting ξ with the class *prior* probability π_{y_i}, π_{y_j}. It is intuitive that a proper factor $\xi_{i,j} = \pi_{y_j}/(\pi_{y_i} + \pi_{y_j})$ ensures $\widetilde{x}_{i,j}$ to be a ξ-*Aug* sample of class y_j if $\pi_{y_i} \geq \pi_{y_j}$, i.e., class y_i occupies more instances than class y_j.

However, $\xi_{i,j}$ is uniquely determined by π_{y_i}, π_{y_j}. To improve the robustness and generalization, original $Beta(\alpha, \alpha)$ is adjusted to obtain UniMix Factor $\xi^*_{i,j}$. Notice that ξ is close to 0 or 1 and symmetric at 0.5, we transform it to maximize the probability of $\xi_{i,j} = \pi_{y_j}/(\pi_{y_i} + \pi_{y_j})$ and its vicinity. Specifically, if note $\xi \sim Beta(\alpha, \alpha)$ as $f(\xi; \alpha, \alpha)$, we define $\xi^*_{i,j} \sim \mathscr{U}(\pi_{y_i}, \pi_{y_j}, \alpha, \alpha)$ as:

$$\xi^*_{i,j} \sim \mathscr{U}(\pi_{y_i}, \pi_{y_j}, \alpha, \alpha) = \begin{cases} f(\xi^*_{i,j} - \dfrac{\pi_{y_j}}{\pi_{y_i} + \pi_{y_j}} + 1; \alpha, \alpha), & \xi^*_{i,j} \in [0, \dfrac{\pi_{y_j}}{\pi_{y_i} + \pi_{y_j}}); \\ f(\xi^*_{i,j} - \dfrac{\pi_{y_j}}{\pi_{y_i} + \pi_{y_j}}; \alpha, \alpha), & \xi^*_{i,j} \in [\dfrac{\pi_{y_j}}{\pi_{y_i} + \pi_{y_j}}, 1] \end{cases} \quad (6)$$

Rethink Eq.2 with $\xi^*_{i,j}$ described as Eq.6:

$$\widetilde{x}_{i,j} = \xi^*_{i,j} \cdot x_i + (1 - \xi^*_{i,j}) \cdot x_j \qquad \widetilde{y}_{i,j} = \xi^*_{i,j} \cdot y_i + (1 - \xi^*_{i,j}) \cdot y_j \quad (7)$$

We have the following corollary to show how $\xi^*_{i,j}$ ameliorates the imbalance of $\mathcal{D}_{train}$:

Corollary 2 *When* $\boldsymbol{\xi^*_{i,j} \sim \mathscr{U}(\pi_{y_i}, \pi_{y_j}, \alpha, \alpha)}, \alpha \in [0,1]$*, the newly mixed dataset* $\mathcal{D}_\nu$ *composed of* ξ*-Aug samples* $(\widetilde{x}_{i,j}, \widetilde{y}_{i,j})$ *follows a middle-majority distribution (see Fig.2), where* (x_i, y_i) *and* (x_j, y_j) *are both* ***randomly*** *sampled from* $\mathcal{D}_{train}$*. (See detail derivation in Appendix A.3.)*

$$\begin{aligned}\mathbb{P}^*_{mixup}(Y^* = y_i) &= \mathbb{P}(Y = y_i) \int_{y_j < y_i} \mathbb{1}\left(\int \xi^*_{i,j} \mathscr{U}(\pi_i, \pi_j, \alpha, \alpha) d\xi^*_{i,j} \geq 0.5\right) \mathbb{P}(Y = y_j) dy_j \\ &= \frac{\lambda}{\left(e^{-\lambda} - e^{-\lambda C}\right)^2} \left(e^{-\lambda(y_i+1)} - e^{-2\lambda y_i}\right), y_i \in [1, C]\end{aligned} \tag{8}$$

UniMix Sampler. UniMix Factor facilitates ξ*-Aug* samples more balance-distributed over all classes. However, most samples are still ξ*-Aug* for the head or middle (see Fig.2(green)). Actually, the constraint that pair x_i, x_j drawn from the head and tail respectively is preferred, which dominantly generates ξ*-Aug* samples for tail classes with $\xi^*_{i,j}$. To this end, we consider sample x_j from $\mathcal{D}_{train}$ with probability inverse to the label *prior*:

$$\mathbb{P}_{inv}(Y = y_i) = \frac{\mathbb{P}^\tau(Y = y_i)}{\int_{y_j \in \mathcal{Y}} \mathbb{P}^\tau(Y = y_j) dy_j} \tag{9}$$

When $\tau = 1$, UniMix Sampler is equivalent to a random sampler. $\tau < 1$ indicates that x_j has higher probability drawn from tail class. Note that x_i is still randomly sampled from $\mathcal{D}_{train}$, i.e., it's most likely drawn from the majority class. The virtual sample $\widetilde{x}_{i,j}$ obtained in this manner is mainly a ξ*-Aug* sample of the tail composite with x_i from the head. Hence Corollary3 is derived:

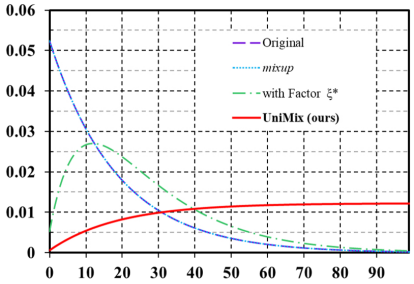


Figure 2: Visualization of ξ*-Aug* samples distribution ($C = 100, \rho = 200$) in Corollary1,2,3. x-axis: class indices. y-axis: probability of each class. *mixup* (blue) exhibits the same LT distribution as origin (purple). ξ^* (green) alleviates such situation and the full pipeline ($\tau = -1$) (red) constructs a more uniform distributed dataset. See more results in Appendix A.5.

Corollary 3 *When* $\boldsymbol{\xi^*_{i,j} \sim \mathscr{U}(\pi_{y_i}, \pi_{y_j}, \alpha, \alpha)}, \alpha \in [0,1]$*, the newly mixed dataset* $\mathcal{D}_\nu$ *composed of* ξ*-Aug samples* $(\widetilde{x}_{i,j}, \widetilde{y}_{i,j})$ *follows a tail-majority distribution (see Fig.2), where* (x_i, y_i) *is* ***randomly*** *and* (x_j, y_j) *is* ***inversely*** *sampled from* $\mathcal{D}_{train}$*, respectively. (See detail derivation in Appendix A.4.)*

$$\begin{aligned}\mathbb{P}_{UniMix}(Y^* = y_i) &= \mathbb{P}(Y = y_i) \int_{y_j < y_i} \mathbb{1}\left(\int \xi^*_{i,j} \mathscr{U}(\pi_i, \pi_j, \alpha, \alpha) d\xi^*_{i,j} \geq 0.5\right) \mathbb{P}_{inv}(Y = y_j) dy_j \\ &= \frac{\lambda}{\left(e^{-\lambda} - e^{-\lambda C}\right)\left(e^{-\lambda\tau C} - e^{-\lambda\tau}\right)} \left(e^{-\lambda y_i(\tau+1)} - e^{-\lambda(\tau+y_i)}\right), y_i \in [1, C]\end{aligned} \tag{10}$$

With the proposed UniMix Factor and UniMix Sampler, we get the complete UniMix manner, which constructs a uniform ξ*-Aug* samples distribution for VRM and greatly facilitates model's *calibration* (See Fig.2 (red) & 1). We construct $\mathcal{D}_\nu := \{(\widetilde{x}_k, \widetilde{y}_k)\}_{k=1}^M$ where $\{\widetilde{x}_k, \widetilde{y}_k\}$ is $(\widetilde{x}_{i,j}, \widetilde{y}_{i,j})$ generated by (x_i, y_i) and (x_j, y_j). We conduct training via Eq.3 and the loss via VRM is available as:

$$\mathcal{L}(\widetilde{y}_k, \mathcal{F}_\theta(\widetilde{x}_k)) = \xi^*_{i,j} \mathcal{L}(y_i, \mathcal{F}_\theta(\widetilde{x}_{i,j})) + (1 - \xi^*_{i,j}) \mathcal{L}(y_j, \mathcal{F}_\theta(\widetilde{x}_{i,j})) \tag{11}$$

3.2 Bayias: an inherent bias in LT

The bias between LT set and balanced set is ineluctable and numerous studies [14, 57, 61] have demonstrated its existence. To eliminate the systematic bias that classifier tends to predict the head, we reconsider the parameters training process. Generally, a classifier can be modeled as:

$$\hat{y} = \arg\max_{y_i \in \mathcal{Y}} \frac{e^{\sum_{d_i \in D}[(W^T)^{(d_i)}_{y_i} \mathcal{F}(x;\theta)^{(d_i)}] + b_{y_i}}}{\sum_{y_j \in \mathcal{Y}} e^{\sum_{d_i \in D}[(W^T)^{(d_i)}_{y_j} \mathcal{F}(x;\theta)^{(d_i)}] + b_{y_j}}} \triangleq \arg\max_{y_i \in \mathcal{Y}} \frac{e^{\psi(x;\theta,W,b)_{y_i}}}{\sum_{y_j \in \mathcal{Y}} e^{\psi(x;\theta,W,b)_{y_j}}} \tag{12}$$

where $\hat{y}$ indicts the predicted label, and $\mathcal{F}(x;\theta) \in \mathbb{R}^{D \times 1}$ is the D-dimension feature extracted by the backbone with parameter θ. $W \in \mathbb{R}^{D \times C}$ represents the parameter matrix of the classifier.

Previous works [14, 57] have demonstrated that it is not suitable for imbalance learning if one ignores such bias. In LT scenarios, the instances number in each class of the train set varies greatly, which means the corresponding *prior* probability $\mathbb{P}_{train}(Y=y)$ is highly skewed whereas the distribution on the test set $\mathbb{P}_{test}(Y=y)$ is uniform.

According to Bayesian theory, *posterior* is proportional to *prior* times *likelihood*. The supervised training process of $\psi(x;\theta,W,b)$ in Eq.12 can regard as the estimation of *likelihood*, which is equivalent to get *posterior* for inference in balanced dataset. Considering the difference of *prior* during training and testing, we have the following theorem (See detail derivation in Appendix B.1):

Theorem 3.1 *For classification, let $\psi(x;\theta,W,b)$ be a hypothesis class of neural networks of input $X=x$, the classification with Softmax should contain the influence of prior, i.e., the predicted label during training should be:*

$$\hat{y} = \arg\max_{y_i \in \mathcal{Y}} \frac{e^{\psi(x;\theta,W,b)_{y_i} + \mathbf{\log(\pi_{y_i})} + \mathbf{\log(C)}}}{\sum_{y_j \in \mathcal{Y}} e^{\psi(x;\theta,W,b)_{y_j} + \mathbf{\log(\pi_{y_j})} + \mathbf{\log(C)}}} \tag{13}$$

In balanced datasets, all classes share the same *prior*. Hence, the supervised model $\psi(x;\theta,W,b)$ could use the estimated *likelihood* $\mathbb{P}(X=x|Y=y)$ of train set to correctly obtain *posterior* $\mathbb{P}(Y=y|X=x)$ in test set. However, in LT datasets where $\mathbb{P}_{train}(Y=y_i)=\pi_{y_i}$ and $\mathbb{P}_{test}(Y=y_i)\equiv 1/C$, *prior* cannot be regard as a constant over all classes any more. Due to the difference on *prior*, the learned parameters $\theta,W,b \triangleq \Theta$ will yield class-level bias, i.e., the optimization direction is no longer as described in Eq.12. Thus, the bias incurred by *prior* should compensate at first. To correctness the bias for inferring, the offset term that model in LT dataset to compensate is:

$$\mathscr{B}_y = \log(\pi_y) + \log(C) \tag{14}$$

Furthermore, the proposed Bayias $\mathscr{B}_y$ enables predicted probability reflecting the actual correctness likelihood, expressed as Theorem 3.2. (See detail derivation in Appendix B.2.)

Theorem 3.2 *$\mathscr{B}_y$-compensated cross-entropy loss in Eq.15 ensures classification calibration.*

$$\mathcal{L}_{\mathscr{B}}(y_i,\psi(x;\Theta)) = \log\left[1+\sum\nolimits_{y_k\neq y_i} e^{(\mathscr{B}_{y_k}-\mathscr{B}_{y_i})}\cdot e^{\psi(x;\Theta)_{y_k}-\psi(x;\Theta)_{y_i}}\right] \tag{15}$$

Here, the optimization direction during training will convert to $\psi(X;\theta,W,b)+\mathscr{B}_y$. In particular, if the train set is balanced, $\mathbb{P}_{train}(Y=y)\triangleq\pi_y\equiv 1/C$, then $\mathscr{B}_y=\log(1/C)+\log(C)\equiv 0$, which means the Eq.12 is a balanced case of Eq.13. We further raise that $\mathscr{B}_y$ is critical to the classification *calibration* in Theorem 3.2. The pairwise loss in Eq.15 will guide model to avoid over-fitting the tail or under-fitting the head with better generalization, which contributes to a better calibrated model.

Compared with logit adjustment [47], which is also a *margin* modification, it is necessary to make a clear statement about the concrete difference from three points. 1) Logit adjustment is motivated by Balanced Error Rate (BER), while the Bayias compensated CE loss is inspired by the Bayesian theorem. We focus more on the model performance on the real-world data distribution. 2) As motioned above, our loss is consistent with standard CE loss when the train set label prior is the same as real test label distribution. 3) Our loss can tackle the imbalanced test set situation as well by simply setting the margin as $\mathscr{B}_y=\log(\pi_y)+\log(\pi'_y)$, where the π'_y represents the test label distribution. The experiment evidence can be found in Appendix Tab.D2.

3.3 Towards calibrated model with UniMix and Bayias

It's intuitive to integrate feature improvement methods with loss modification ones for better performance. However, we find such combinations fail in most cases and are counterproductive with each other, i.e., the combined methods reach unsatisfactory performance gains. We suspect that these methods take contradictory trade-offs and thus result in overconfidence and bad *calibration*. Fortunately, the proposed UniMix and Bayias are both proven to ensure *calibration*. To achieve a better-calibrated model for superior performance gains, we introduce Alg.1 to tackle the previous dilemma and integrate our two proposed approaches to deal with poor generalization of tail classes. Specially, inspired by previous work [26], we overcome the coverage difficulty in *mixup* [64] by removing UniMix in the last several epochs and thus maintain the same epoch as baselines. Note that Bayias-compensated CE is only adopted in the training process as discussed in Sec.3.2.

Algorithm 1 Integrated training manner towards calibrated model.

Input: $\mathcal{D}_{train}$, Batch Size $\mathcal{N}$, Stop Steps T_1, T_2, Random Sampler $\mathcal{R}$, UniMix Sampler $\mathcal{R}^*$
Output: Optimized Θ^*, i.e., feature extractor parameters θ^*, classifier parameters W^*, b^*

Initialize the parameters $\Theta^{(0)}$ randomly and calculate $\mathscr{B}_y$ via Eq.14
for $t = 0$ to T_1 **do**
  Sample a mini-batch $\mathcal{B} = \{x_i, y_i\}_{i=1}^{\mathcal{N}} \leftarrow \mathcal{R}(\mathcal{D}_{train}, \mathcal{N})$
  Sample a mini-batch $\mathcal{B}^* = \{x_j^*, y_j^*\}_{j=1}^{\mathcal{N}} \leftarrow \mathcal{R}^*(\mathcal{D}_{train}, \mathcal{N})$
  Calculate UniMix factor ξ^* via Eq.6
  Construct VRM dataset $\mathcal{B}_\nu = \{\widetilde{x}_k, \widetilde{y}_k\}_{k=1}^{\mathcal{N}}$ via Eq.7
  Calculate $\mathcal{L}_{\mathcal{B}_\nu} = \mathbb{E}[\xi_{i,j}^* \mathcal{L}_{\mathscr{B}}(y_i, \psi(\widetilde{x}; \Theta^{(t)})) + (1 - \xi_{i,j}^*)\mathcal{L}_{\mathscr{B}}(y_j^*, \psi(\widetilde{x}; \Theta^{(t)}))]$ via Eq.11,15
  Update $\Theta^{(t+1)} \leftarrow \Theta^{(t)} - \alpha \nabla_{\Theta^{(t)}} \mathcal{L}_{\mathcal{B}_\nu}$
end for
for $t = T_1$ to T_2 **do**
  Sample a mini-batch $\mathcal{B} = \{x_i, y_i\}_{i=1}^{\mathcal{N}} \leftarrow \mathcal{R}(\mathcal{D}_{train}, \mathcal{N})$
  Calculate $\mathcal{L}_{\mathcal{B}} = \mathbb{E}[\mathcal{L}_{\mathscr{B}}(y_i, \psi(x_i; \Theta))]$ via Eq.15
  Update $\Theta^{(t+1)} \leftarrow \Theta^{(t)} - \alpha \nabla_{\Theta^{(t)}} \mathcal{L}_{\mathcal{B}}$
end for

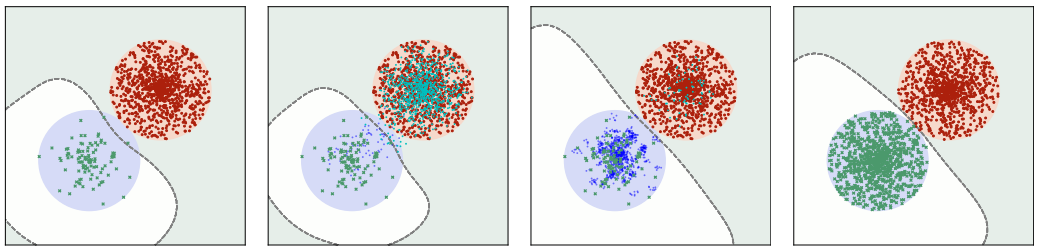
4 Experiment

4.1 Results on synthetic dataset

We make an ideal binary classification using Support Vector Machine (SVM) [16] to show the distinguish effectiveness of UniMix. Suppose there are samples from two disjoint circles respectively:

$$\begin{aligned} z^+ &= \{(x, y) | (x - x_0)^2 + (y - y_0)^2 \leq r^2\} \\ z^- &= \{(x, y) | (x + x_0)^2 + (y + y_0)^2 \leq r^2\} \end{aligned} \tag{16}$$

To this end, we randomly sample m discrete point pairs from z^+ to compose positive samples $z_p^+ = \{(x_1^+, y_1^+), \cdots, (x_m^+, y_m^+)\}$, and m negative samples $z_n^- = \{(x_1^-, y_1^+), \cdots, (x_m^-, y_m^-)\}$ from z^- correspondingly, thus to generate a balanced dataset $\mathcal{D}_{bal} = \{z_p^+, z_n^-\}$ with $\mathbb{P}\left((x, y) \in z_p^+\right) = \mathbb{P}\left((x, y) \in z_n^-\right) = 0.5$. For imbalance data, we sample $n (n \ll m)$ negative data from z^- to generate $z_{n'}^- = \{(x_1'^-, y_1'^-), \cdots, (x_n'^-, y_n'^-)\}$, so as to compose the imbalance dataset $\mathcal{D}_{imbal} = \{z_p^+, z_{n'}^-\}$, with $\mathbb{P}\left((x, y) \in z_p^+\right) \gg \mathbb{P}\left((x, y) \in z_{n'}^-\right)$. We train the SVM model on the two synthetic datasets, and visualize the classification boundary of each dataset in Fig.3.



(a) imbalanced scenario (b) with *mixup* (c) with UniMix (d) balanced scenario

Figure 3: SVM decision boundary on the synthetic balanced dataset (Fig.3(d)) and imbalanced dataset (Fig.3(a),3(b),3(c)). The theoretical classification boundary of the synthetic dataset is y=$-x$. "$\circ$" represents generated pseudo data, where blue and green represent belong to z^- and z^+, respectively.

The SVM reaches an approximate ideal boundary on balanced datasets (Fig.3(d)) but severely deviates from the $y = -x$ in the imbalanced dataset (Fig.3(a)). As proven in Sec.2, *mixup* (Fig.3(b)) is incapable of shifting imbalance distribution, resulting in no better result than the original one (Fig.3(a)). After adopting the proposed UniMix, the classification boundary in Fig.3(c) shows much better results than the original imbalanced dataset, which gets closed to the ideal boundary.

4.2 Results on CIFAR-LT

The imbalanced datasets CIFAR-10-LT and CIFAR-100-LT are constructed via suitably discarding training samples following previous works [69, 14, 47, 8]. The instance numbers exponentially decay per class in train dataset and keep balanced during inference. We extensively adopt $\rho \in \{10, 50, 100, 200\}$ for comprehensive comparisons. See implementation details in Appendix C.1.

Comparison methods. We evaluate the proposed method against various representative and effective approaches extensively, summarized into the following groups: **a) Baseline.** We conduct plain training with CE loss called ERM as baseline. **b) Feature improvement methods** modify the input feature to cope with LT datasets. *mixup* [64] convexly combines images and labels to build virtual data for VRM. Manifold mixup [59] performs the linear combination in latent states. Remix [12] conducts the same combination on images and adopts tail-favored rules on labels. M2m [36] converts majority images to minority ones by adding noise perturbation, which need an additional pre-trained classifier. BBN [69] utilizes features from two branches in a cumulative learning manner. **c) Loss modification methods** either adjust the logits *weight* or *margin* before the *Softmax* operation. Specifically, focal loss [41], CB [14] and CDT [62] re-weight the logits with elaborate strategies, while LDAM [8] and Logit Adjustment [47] add the logits *margin* to shift decision boundary away from tail classes. **d) Other methods.** We also compare the proposed method with other two-stage approaches (e.g. DRW [8]) for comprehensive comparisons.

Table 1: Top-1 validation accuracy(%) of ResNet-32 on CIFAR-10/100-LT. E2E: end to end training. Underscore: the best performance in each group. †: our reproduced results. ‡: reported results in [69]. ⋆: reported results in [62]. Our *calibration* ensured method achieves the best performance.

Dataset	E2E	CIFAR-10-LT				CIFAR-100-LT			
ρ (easy $\rightarrow$ hard)	-	10	50	100	200	10	50	100	200
ERM†	✓	86.39	74.94	70.36	66.21	55.70	44.02	38.32	34.56
mixup‡ [64]	✓	87.10	77.82	73.06	67.73	58.02	44.99	39.54	34.97
Manifold Mixup‡ [59]	✓	87.03	77.95	72.96	-	56.55	43.09	38.25	-
Remix [12]	✓	88.15	79.20	75.36	67.08	59.36	46.21	41.94	36.99
M2m [36]	✗	87.90	-	78.30	-	58.20	-	42.90	-
BBN‡ [69]	✗	88.32	82.18	79.82	-	59.12	47.02	42.56	-
Focal⋆ [41]	✓	86.55	76.71†	70.43	65.85	55.78	44.32†	38.41	35.62
Urtasun et al [51]	✓	82.12	76.45	72.23	66.25	52.12	43.17	38.90	33.00
CB-Focal [14]	✓	87.10	79.22	74.57	68.15	57.99	45.21	39.60	36.23
τ-norm⋆ [33]	✓	87.80	82.78†	75.10	70.30	59.10	48.23†	43.60	39.30
LDAM† [8]	✓	86.96	79.84	74.47	69.50	56.91	46.16	41.76	37.73
LDAM+DRW† [8]	✗	88.16	81.27	77.03	74.74	58.71	47.97	42.04	38.45
CDT⋆ [62]	✓	89.40	81.97†	79.40	74.70	58.90	45.15†	44.30	40.50
Logit Adjustment [47]	✓	89.26†	83.38†	79.91	75.13†	59.87†	49.76†	43.89	40.87†
Ours	✓	**89.66**	**84.32**	**82.75**	**78.48**	**61.25**	**51.11**	**45.45**	**42.07**

Results. We present results of CIFAR-10-LT and CIFAR-100-LT in Tab.1. Our proposed method achieves state-of-the-art results against others on each ρ, with performance gains improved as ρ gets increased (See Appendix D.1). Specifically, our method overcomes the ignorance in tail classes effectively with better *calibration*, which integrates advantages of two group approaches and thus surpass most two-stage methods (i.e., BBN, M2m, LDAM+DRW). However, not all combinations can get ideal performance gains as expected. More details will be discussed in Sec.4.4.

To quantitatively describe the contribution to model *calibration*, we make quantitative comparisons on CIFAR-10-LT and CIFAR-100-LT. According to the definition ECE and MCE (see Appendix Eq.D.2,D.3), a well-calibrated model should minimize the ECE and MCE for better generalization and robustness. In this experiment, we adopt the most representative $\rho \in \{10, 100\}$ with previous mainstream state-of-the-art methods.

The results in Tab.2 show that either of the proposed methods generally outperforms previous methods, and their combination enables better classification *calibration* with smaller ECE and MCE. Specifically, *mixup* and Remix have negligible contributions to model *calibration*. As analyzed before, such methods tend to generate head-head pairs in favor of the feature learning of majority

Table 2: Quantitative *calibration* metric of ResNet-32 on CIFAR-10/100-LT test set. Smaller ECE and MCE indicate better *calibration* results. Either of the proposed methods achieves a well-calibrated model compared with others. The combination of UniMix and Bayias achieves the best performance.

Dataset	CIFAR-10-LT				CIFAR-100-LT			
ρ	10		100		10		100	
Calibration Metric (%)	ECE	MCE	ECE	MCE	ECE	MCE	ECE	MCE
ERM	6.60	74.96	20.53	73.91	22.85	34.50	38.23	87.22
mixup [64]	6.55	24.54	19.20	37.84	19.69	38.53	32.72	50.46
Remix [12]	6.81	22.44	15.38	27.99	20.17	32.99	33.56	50.96
LDAM+DRW [8]	11.22	45.92	19.89	49.07	30.54	55.57	42.18	64.78
UniMix (ours)	6.00	25.99	12.87	28.30	19.38	33.40	27.12	41.46
Bayias (ours)	5.52	20.14	11.05	**23.72**	17.42	28.26	24.31	39.66
UniMix+Bayias (ours)	**4.74**	**13.67**	**10.19**	25.47	**15.24**	**23.67**	**23.04**	**37.36**

classes. However, more head-tail pairs are required for better feature representation of the tail classes. In contrast, both the proposed UniMix and Bayias pay more attention to the tail and achieve satisfactory results. It is worth mentioning that improving *calibration* in post-hoc manners [18, 68] is also effective, and we will discuss it in Appendix D.2.3. Note that LDAM is even worse calibrated compared with baseline. We suggest that LDAM adopts an additional margin only for the ground-truth label from the angular perspective, which shifts the decision boundary away from the tail class and makes the tail predicting score tend to be larger. Additionally, LDAM requires the normalization of input features and classifier weight matrix. Although a scale factor is proposed to enlarge the logits for better *Softmax* operation [60], it is still harmful to *calibration*. It also accounts for its contradiction with other methods. Miscalibration methods combined will make models become even more overconfident and damage the generalization and robustness severely.

4.3 Results on large-scale datasets

We further verify the proposed method's effectiveness quantitatively on large-scale imbalanced datasets, i.e. ImageNet-LT and iNaturalist 2018. ImageNet-LT is the LT version of ImageNet [53] by sampling a subset following *Pareto* distribution, which contains about $115K$ images from $1,000$ classes. The number of images per class varies from 5 to $1,280$ exponentially, i.e., $\rho = 256$. In our experiment, we utilize the balanced validation set constructed by Cui *et al.* [14] for fair comparisons. The iNaturalist species classification dataset [28] is a large-scale real-world dataset which suffers from extremely label LT distribution and fine-grained problems [28, 69]. It is composed of $435,713$ images over $8,142$ classes with $\rho = 500$. The official splits of train and validation images [8, 69, 33] are adopted for fair comparisons. See implementation details in Appendix C.2.

Table 3: Top-1 validation accuracy(%) of ResNet-10/50 on ImageNet-LT and ResNet-50 on iNaturalist 2018. E2E: end to end training. †: our reproduced results. ‡: results reported in origin paper.

Dataset		ImageNet-LT				iNaturalist 2018	
Method	E2E	ResNet-10	Δ	ResNet-50	Δ	ResNet-50	Δ
CE†	✓	35.88	-	38.88	-	60.88	-
CB-CE† [14]	✓	37.06	+1.18	40.85	+1.97	63.50	+2.62
LDAM [8]	✓	36.05†	+0.17	41.86†	+2.98	64.58‡	+3.70
OLTR‡ [45]	✗	35.60	-0.28	40.36	+1.48	63.90	+3.02
LDAM+DRW [8]	✗	38.22†	+2.34	45.75†	+6.87	68.00‡	+7.12
BBN‡ [69]	✗	-		-		66.29	+5.41
c-RT [33]	✗	41.80‡	+5.92	47.54†	+8.66	67.60†	+6.72
Ours	✓	**42.90**	**+7.02**	**48.41**	**+9.53**	**69.15**	**+8.27**

Results. Tab.3 illustrates the results on large-scale datasets. Ours is consistently effective and outperforms existing mainstream methods, achieving distinguish improvement compared with previous SOTA c-RT [33] in the compared backbones. Especially, our method outperforms the baseline on ImageNet-LT and iNaturalist 2018 by **9.53%** and **8.27%** with ResNet-50, respectively. As can be noticed in Tab.3, the proposed method also surpasses the well-known two-stage methods [33, 8, 69], achieving superior accuracy with less computation load in a concise training manner.

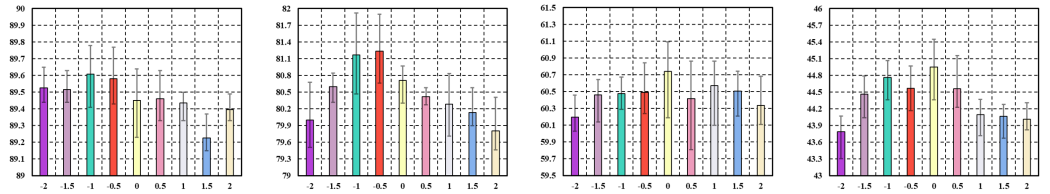
4.4 Further Analysis

Effectiveness of UniMix and Bayias. We conduct extensive ablation studies in Tab.4 to demonstrate the effectiveness of the proposed UnixMix and Bayias, with detailed analysis in various combinations of feature-wise and loss-wise methods on CIFAR-10-LT and CIFAR-100-LT. Indeed, both UniMix and Bayias turn out to be effective in LT scenarios. Further observation shows that with *calibration* ensured, the proposed method gets significant performance gains and achieve state-of-the-art results. Noteworthy, LDAM [8] makes classifiers miscalibrated, which leads to unsatisfactory improvement when combined with *mixup* manners.

Table 4: Ablation study between feature-wise and loss-wise methods. LDAM is counterproductive to *mixup* and its extensions. Bayias-compensated CE ensures *calibration* and shows excellent performance gains especially combined with UniMix.

Dataset		CIFAR-10-LT				CIFAR-100-LT			
ρ (easy $\rightarrow$ hard)		100		200		100		200	
Mix	Loss	Top1 Acc	Δ	Top1 Acc	Δ	Top1 Acc	Δ	Top1 Acc	Δ
None	CE	70.36	-	66.21	-	38.32	-	34.56	-
mixup	CE	73.06	+2.70	67.73	+1.52	39.54	+1.22	34.97	+0.41
Remix	CE	75.36	+5.00	67.08	+0.87	41.94	+3.62	36.99	+2.43
UniMix	CE	76.47	+6.11	68.42	+2.21	41.46	+3.14	37.63	+3.07
None	LDAM	74.47	-	69.50	-	41.76	-	37.73	-
mixup	LDAM	73.96	-0.15	67.89	-1.61	40.22	-1.54	37.52	-0.21
Remix	LDAM	74.33	-0.14	69.66	+0.16	40.59	-1.17	37.66	-0.07
UniMix	LDAM	75.35	+0.88	70.77	+1.27	41.67	-0.09	37.83	+0.01
None	Bayias	78.70	-	74.21	-	43.52	-	38.83	-
mixup	Bayias	81.75	+3.05	76.69	+2.48	44.56	+1.04	41.19	+2.36
Remix	Bayias	81.55	+2.85	75.81	+1.60	45.01	+1.49	41.44	+2.61
UniMix	Bayias	**82.75**	**+4.05**	**78.48**	**+4.27**	**45.45**	**+1.93**	**42.07**	**+3.24**

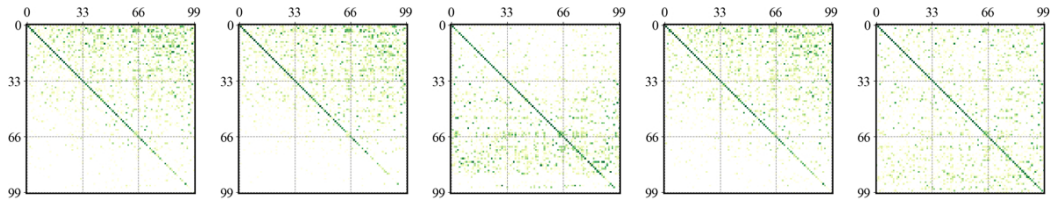
Evaluating different UniMix Sampler. Corollary 1,2,3 demonstrate distinguish influence of UniMix. However, the ξ-*sample* can not be completely equivalent with orginal ones. Hence, an appropriate τ in Eq.9 is also worth further searching. Fig.4 illustrates the accuracy with different τ on CIFAR-10-LT and CIFAR-100-LT setting $\rho = 10$ and 100. For CIFAR-10-LT (Fig.4(a),4(b)), $\tau = -1$ is possibly ideal, which forces more head-tail instead of head-head pairs get generated to compensate tail classes. In the more challenging CIFAR-100-LT, $\tau = 0$ achieves the best result. We suspect that unlike simple datasets (e.g., CIFAR-10-LT), where overconfidence occurs in head classes, all classes need to get enhanced in complicated LT scenarios. Hence, the augmentation is effective and necessary both on head and tail. $\tau = 0$ allows both head and tail get improved simultaneously.



(a) CIFAR-10-LT-10. (b) CIFAR-10-LT-100. (c) CIFAR-100-LT-10. (d) CIFAR-100-LT-100.

Figure 4: Comparison of top-1 validation accuracy(%) of ResNet-32 on CIFAR-LT when varying τ in Eq.9 for UniMix. The histogram indicates average results in repeated experiments.

Do minorities really get improved? To observe the amelioration on tail classes, Fig.5 visualizes log-confusion matrices on CIFAR-100-LT-100. In Fig.5(e), our method exhibits satisfactory generalization on the tail. Vanilla ERM model (Fig.5(a)) is a trivial predictor which simplifies tail instances as head labels to minimize the error rate. Feature improvement [12] and loss modification [8, 62] methods do alleviate LT problem to some extent. The misclassification cases (i.e., non-diagonal elements) in Fig.5(b),5(c),5(d) become smaller and more balanced distributed compared with ERM. However, the error cases are still mainly in the upper or lower triangular, indicating the existence of inherent bias between the head and tail. Our method (Fig.5(e)) significantly alleviates such dilemma. The non-diagonal elements are more uniformly distributed throughout the matrix rather than in



(a) ERM (b) LDAM+DRW [8] (c) CDT [62] (d) Remix [12] (e) Ours

Figure 5: The log-confusion matrix on CIFAR-100-LT-100 validation dataset. The x-axis and y-axis indicate the ground truth and predicted labels, respectively. Deeper color indicates larger values.

the corners, showing superiority to erase the bias in LT scenarios. Our method enables effective feature improvement for data-scarce classes and alleviates the *prior* bias, suggesting our success in regularizing tail remarkably.

5 Related work and discussion

Why need *calibration*? To quantify the predictive uncertainty, *calibration* [3] is put forward to describe the relevance between predictive score and actual correctness likelihood. A well-calibrated model is more reliable with better interpretability, which probabilities indicate optimal expected costs in Bayesian decision scenarios [46]. Guo *et al.* [18] firstly provide metric to measure the *calibration* of CNN and figure out well-performed models are always in lack of *calibration*, indicating that CNN is sensitive to be overconfidence and lacks robustness. Thulasidasan *et al.* [58] point out that the effectiveness of *mixup* in balanced datasets originates from superior *calibration* modification. Menon *et al.* [47] further show how to ensure optimal classification *calibration* for a pair-wise loss.

Feature-wise methods. Intuitively, under-sampling the head [20, 6, 22] or over-sampling the tail [11, 6, 48, 54, 7] can improve the inconsistent performance of imbalanced datasets but tend to either weaken the head or over-fitting the tail. Hence, many effective works generate additional samples [13, 67, 36] to compensate the tail classes. BBN [69] uses two branches to extract features from head and tail simultaneously, while c-RT [33] trains feature representation learning and classification stage separately. *mixup* [64] and its variants [59, 63, 12] are effective and easy-implement feature-wise methods that convexly combine input and label pairs to generate virtual samples. However, naïve *mixup* manners are deficient in LT scenarios as we discussed in Sec.2. In contrast, our UniMix tackles such a dilemma by constructing class balance-oriented virtual data as describe in Sec.3.1 and shows satisfactory *calibration* as Fig.1 exhibits.

Loss modification. Numerous experimental and theoretical studies [50, 14, 57, 61] have demonstrated the existence of inherent *bias* between LT train set and balanced test set in supervised learning. Previous works [30, 31, 34, 14, 41] make networks prefer learning tail samples by additional class-related weight on CE loss. Some works further correct CE according to the gradient generated by different samples [56, 39] or from the perspective of Gaussian distribution and Bayesian estimation [21, 35]. Meta-learning approaches [17, 1, 55, 51, 32] optimize the weights of each class in CE as learnable parameters and achieve remarkable success. Cao *et al.* [8] theoretically provides the ideal optimal *margin* for CE from the perspective of *VC* generalization bound. Compared with Logit Adjustment [47] motivated by balance error rate, our Bayias-compensated CE eliminates *bias* incured by *prior* and is consistent with balanced datasets, which ensures classification *calibration* as well.

6 Conclusion

We systematically analyze the limitations of mainstream feature improvement methods, i.e., *mixup* and its extensions in the label-imbalanced situation, and propose the UniMix to construct a more class-balanced virtual dataset that significantly improves classification *calibration*. We further pinpoint an inherent bias induced by the inconstancy of label distribution *prior* between long-tailed train set and balanced test set. We prove that the standard cross-entropy loss with the proposed Bayias compensated can ensure classification *calibration*. The combination of UniMix and Bayias achieves state-of-the-art performance and contributes to a better-calibrated model (Fig.1). Further study in Tab.4 shows that the bad *calibration* methods are counterproductive with each other. However, more in-depth analysis and theoretical guarantees are still required, which we leave for our future work.

Acknowledgments and Disclosure of Funding

This work was supported by NSFC project Grant No. U1833101, SZSTI Grant No. JCYJ20190809 172201639 and WDZC20200820200655001, the Joint Research Center of Tencent and Tsinghua.

References

[1] Marcin Andrychowicz, Misha Denil, Sergio Gomez Colmenarejo, Matthew W. Hoffman, David Pfau, Tom Schaul, and Nando de Freitas. Learning to learn by gradient descent by gradient descent. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016*, pages 3981–3989, 2016.

[2] Arsenii Ashukha, Alexander Lyzhov, Dmitry Molchanov, and Dmitry P. Vetrov. Pitfalls of in-domain uncertainty estimation and ensembling in deep learning. In *8th International Conference on Learning Representations, ICLR 2020*. OpenReview.net, 2020.

[3] Peter L Bartlett, Michael I Jordan, and Jon D McAuliffe. Convexity, classification, and risk bounds. *Journal of the American Statistical Association*, 101(473):138–156, 2006.

[4] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D. Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, Xin Zhang, Jake Zhao, and Karol Zieba. End to end learning for self-driving cars. *CoRR*, abs/1604.07316, 2016.

[5] Glenn W Brier et al. Verification of forecasts expressed in terms of probability. *Monthly weather review*, 78(1):1–3.

[6] Mateusz Buda, Atsuto Maki, and Maciej A. Mazurowski. A systematic study of the class imbalance problem in convolutional neural networks. *Neural Networks*, 106:249–259, 2018.

[7] Jonathon Byrd and Zachary Chase Lipton. What is the effect of importance weighting in deep learning? In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019*, volume 97 of *Proceedings of Machine Learning Research*, pages 872–881. PMLR, 2019.

[8] Kaidi Cao, Colin Wei, Adrien Gaidon, Nikos Aréchiga, and Tengyu Ma. Learning imbalanced datasets with label-distribution-aware margin loss. In *Advances in Neural Information Processing Systems NeurIPS 2019*, pages 1565–1576, 2019.

[9] Rich Caruana, Yin Lou, Johannes Gehrke, Paul Koch, Marc Sturm, and Noemie Elhadad. Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1721–1730. ACM, 2015.

[10] Olivier Chapelle, Jason Weston, Léon Bottou, and Vladimir Vapnik. Vicinal risk minimization. In *Advances in Neural Information Processing Systems 13, Papers from Neural Information Processing Systems (NIPS) 2000, Denver, CO, USA*, pages 416–422. MIT Press, 2000.

[11] Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. SMOTE: synthetic minority over-sampling technique. *J. Artif. Intell. Res.*, 16:321–357, 2002.

[12] Hsin-Ping Chou, Shih-Chieh Chang, Jia-Yu Pan, Wei Wei, and Da-Cheng Juan. Remix: Rebalanced mixup. In *Computer Vision - ECCV 2020 Workshops*, volume 12540, pages 95–110. Springer, 2020.

[13] Peng Chu, Xiao Bian, Shaopeng Liu, and Haibin Ling. Feature space augmentation for long-tailed data. In *Computer Vision - ECCV 2020 - 16th European Conference*, volume 12374 of *Lecture Notes in Computer Science*, pages 694–710. Springer, 2020.

[14] Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge J. Belongie. Class-balanced loss based on effective number of samples. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019*, pages 9268–9277. Computer Vision Foundation / IEEE, 2019.

[15] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019*, pages 4690–4699. Computer Vision Foundation / IEEE, 2019.

[16] Theodoros Evgeniou and Massimiliano Pontil. Support vector machines: Theory and applications. In *Machine Learning and Its Applications, Advanced Lectures*, volume 2049 of *Lecture Notes in Computer Science*, pages 249–257. Springer, 2001.

[17] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 1126–1135. PMLR, 2017.

[18] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 1321–1330. PMLR, 2017.

[19] Agrim Gupta, Piotr Dollár, and Ross B. Girshick. LVIS: A dataset for large vocabulary instance segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019*, pages 5356–5364. Computer Vision Foundation / IEEE, 2019.

[20] Hui Han, Wenyuan Wang, and Binghuan Mao. Borderline-smote: A new over-sampling method in imbalanced data sets learning. In *Advances in Intelligent Computing, International Conference on Intelligent Computing, ICIC 2005*, volume 3644 of *Lecture Notes in Computer Science*, pages 878–887. Springer, 2005.

[21] Munawar Hayat, Salman H. Khan, Syed Waqas Zamir, Jianbing Shen, and Ling Shao. Gaussian affinity for max-margin class imbalanced learning. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019*, pages 6468–6478. IEEE, 2019.

[22] Haibo He and Edwardo A. Garcia. Learning from imbalanced data. *IEEE Trans. Knowl. Data Eng.*, 21(9):1263–1284, 2009.

[23] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask R-CNN. In *IEEE International Conference on Computer Vision, ICCV 2017*, pages 2980–2988. IEEE Computer Society, 2017.

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

[25] Tong He, Zhi Zhang, Hang Zhang, Zhongyue Zhang, Junyuan Xie, and Mu Li. Bag of tricks for image classification with convolutional neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019*, pages 558–567. Computer Vision Foundation / IEEE, 2019.

[26] Zhuoxun He, Lingxi Xie, Xin Chen, Ya Zhang, Yanfeng Wang, and Qi Tian. Data augmentation revisited: Rethinking the distribution gap between clean and augmented data. *CoRR*, abs/1909.09148, 2019.

[27] 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.

[28] Grant Van Horn, Oisin Mac Aodha, Yang Song, Yin Cui, Chen Sun, Alexander Shepard, Hartwig Adam, Pietro Perona, and Serge J. Belongie. The inaturalist species classification and detection dataset. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018*, pages 8769–8778. IEEE Computer Society, 2018.

[29] Grant Van Horn and Pietro Perona. The devil is in the tails: Fine-grained classification in the wild. *CoRR*, abs/1709.01450, 2017.

[30] Chen Huang, Yining Li, Chen Change Loy, and Xiaoou Tang. Learning deep representation for imbalanced classification. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016*, pages 5375–5384. IEEE Computer Society, 2016.

[31] Chen Huang, Yining Li, Chen Change Loy, and Xiaoou Tang. Deep imbalanced learning for face recognition and attribute prediction. *IEEE Trans. Pattern Anal. Mach. Intell.*, 42(11):2781–2794, 2020.

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

[33] Bingyi Kang, Saining Xie, Marcus Rohrbach, Zhicheng Yan, Albert Gordo, Jiashi Feng, and Yannis Kalantidis. Decoupling representation and classifier for long-tailed recognition. In *8th International Conference on Learning Representations, ICLR 2020*. OpenReview.net, 2020.

[34] Salman H. Khan, Munawar Hayat, Mohammed Bennamoun, Ferdous Ahmed Sohel, and Roberto Togneri. Cost-sensitive learning of deep feature representations from imbalanced data. *IEEE Trans. Neural Networks Learn. Syst.*, 29(8):3573–3587, 2018.

[35] Salman H. Khan, Munawar Hayat, Syed Waqas Zamir, Jianbing Shen, and Ling Shao. Striking the right balance with uncertainty. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019*, pages 103–112. Computer Vision Foundation / IEEE, 2019.

[36] Jaehyung Kim, Jongheon Jeong, and Jinwoo Shin. M2m: Imbalanced classification via major-to-minor translation. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020*, pages 13893–13902. IEEE, 2020.

[37] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A. Shamma, Michael S. Bernstein, and Li Fei-Fei. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *Int. J. Comput. Vis.*, 123(1):32–73, 2017.

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

[39] Buyu Li, Yu Liu, and Xiaogang Wang. Gradient harmonized single-stage detector. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019*, pages 8577–8584. AAAI Press, 2019.

[40] Yu Li, Tao Wang, Bingyi Kang, Sheng Tang, Chunfeng Wang, Jintao Li, and Jiashi Feng. Overcoming classifier imbalance for long-tail object detection with balanced group softmax. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10991–11000, 2020.

[41] Tsung-Yi Lin, Priya Goyal, Ross B. Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *IEEE International Conference on Computer Vision, ICCV 2017*, pages 2999–3007. IEEE Computer Society, 2017.

[42] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. In *Computer Vision - ECCV 2014 - 13th European Conference*, volume 8693, pages 740–755. Springer, 2014.

[43] Hao Liu, Xiangyu Zhu, Zhen Lei, and Stan Z. Li. Adaptiveface: Adaptive margin and sampling for face recognition. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019*, pages 11947–11956. Computer Vision Foundation / IEEE, 2019.

[44] Weiyang Liu, Yandong Wen, Zhiding Yu, Ming Li, Bhiksha Raj, and Le Song. Sphereface: Deep hypersphere embedding for face recognition. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, pages 6738–6746. IEEE Computer Society, 2017.

[45] Ziwei Liu, Zhongqi Miao, Xiaohang Zhan, Jiayun Wang, Boqing Gong, and Stella X. Yu. Large-scale long-tailed recognition in an open world. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 2537–2546. Computer Vision Foundation / IEEE, 2019.

[46] Juan Maroñas, Roberto Paredes, and Daniel Ramos. Calibration of deep probabilistic models with decoupled bayesian neural networks. *Neurocomputing*, 407:194–205, 2020.

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

[48] Sankha Subhra Mullick, Shounak Datta, and Swagatam Das. Generative adversarial minority oversampling. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019*, pages 1695–1704. IEEE, 2019.

[49] Jeremy Nixon, Michael W. Dusenberry, Linchuan Zhang, Ghassen Jerfel, and Dustin Tran. Measuring calibration in deep learning. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2019*, pages 38–41.

[50] Jiawei Ren, Cunjun Yu, Shunan Sheng, Xiao Ma, Haiyu Zhao, Shuai Yi, and Hongsheng Li. Balanced meta-softmax for long-tailed visual recognition. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020*, 2020.

[51] Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. Learning to reweight examples for robust deep learning. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 4331–4340. PMLR, 2018.

[52] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster R-CNN: towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015*, pages 91–99, 2015.

[53] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Fei-Fei Li. Imagenet large scale visual recognition challenge. *Int. J. Comput. Vis.*, 115(3):211–252, 2015.

[54] Li Shen, Zhouchen Lin, and Qingming Huang. Relay backpropagation for effective learning of deep convolutional neural networks. In *Computer Vision - ECCV 2016 - 14th European Conference*, volume 9911, pages 467–482. Springer, 2016.

[55] Jun Shu, Qi Xie, Lixuan Yi, Qian Zhao, Sanping Zhou, Zongben Xu, and Deyu Meng. Meta-weight-net: Learning an explicit mapping for sample weighting. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019*, pages 1917–1928, 2019.

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

[57] Kaihua Tang, Jianqiang Huang, and Hanwang Zhang. Long-tailed classification by keeping the good and removing the bad momentum causal effect. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020*, 2020.

[58] Sunil Thulasidasan, Gopinath Chennupati, Jeff A. Bilmes, Tanmoy Bhattacharya, and Sarah Michalak. On mixup training: Improved calibration and predictive uncertainty for deep neural networks. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019*, pages 13888–13899, 2019.

[59] Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, David Lopez-Paz, and Yoshua Bengio. Manifold mixup: Better representations by interpolating hidden states. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019*, volume 97, pages 6438–6447. PMLR, 2019.

[60] Feng Wang, Xiang Xiang, Jian Cheng, and Alan Loddon Yuille. Normface: L_2 hypersphere embedding for face verification. In *Proceedings of the 2017 ACM on Multimedia Conference, MM 2017*, pages 1041–1049. ACM, 2017.

[61] Yuzhe Yang and Zhi Xu. Rethinking the value of labels for improving class-imbalanced learning. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020*, 2020.

[62] Han-Jia Ye, Hong-You Chen, De-Chuan Zhan, and Wei-Lun Chao. Identifying and compensating for feature deviation in imbalanced deep learning. *CoRR*, abs/2001.01385, 2020.

[63] Sangdoo Yun, Dongyoon Han, Sanghyuk Chun, Seong Joon Oh, Youngjoon Yoo, and Junsuk Choe. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019*, pages 6022–6031. IEEE, 2019.

[64] Hongyi Zhang, Moustapha Cissé, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *6th International Conference on Learning Representations, ICLR 2018*. OpenReview.net, 2018.

[65] Linjun Zhang, Zhun Deng, Kenji Kawaguchi, Amirata Ghorbani, and James Zou. How does mixup help with robustness and generalization? In *International Conference on Learning Representations*, 2021.

[66] Yifan Zhang, Bryan Hooi, Lanqing Hong, and Jiashi Feng. Test-agnostic long-tailed recognition by test-time aggregating diverse experts with self-supervision. *arXiv preprint arXiv:2107.09249*, 2021.

[67] Yongshun Zhang, Xiu-Shen Wei, Boyan Zhou, and Jianxin Wu. Bag of tricks for long-tailed visual recognition with deep convolutional neural networks. In *AAAI*, 2021.

[68] Zhisheng Zhong, Jiequan Cui, Shu Liu, and Jiaya Jia. Improving calibration for long-tailed recognition. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021*, pages 16489–16498. Computer Vision Foundation / IEEE, 2021.

[69] Boyan Zhou, Quan Cui, Xiu-Shen Wei, and Zhao-Min Chen. BBN: bilateral-branch network with cumulative learning for long-tailed visual recognition. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020*, pages 9716–9725. IEEE, 2020.

A Missing proofs and derivations of UniMix

A.1 Basic setting

Setting 1 *Without loss of generality, we suppose that the long-tailed distribution satisfies some kind exponential distribution with parameter λ [14]. The imbalance factor is defined as $\rho = n_{\max}/n_{\min}$, where $n_{\max}$ and $n_{\min}$ represent the number of the most and least samples in the dataset with C classes, respectively. Then the probability of class Y belongs to y_i is:*

$$\mathbb{P}(Y = y_i) = \begin{cases} \alpha e^{-\lambda y_i} & y_i \in [1, C] \\ 0 & others \end{cases} \quad \sim \quad long-tailed \ \ distribution \tag{A.1}$$

According to the definition of ρ, we can directly deduce the relationship between ρ and λ:

$$\rho = \frac{n_{\max}}{n_{\min}} = \frac{\mathbb{P}(Y = y_i)_{\max}}{\mathbb{P}(Y = y_i)_{\min}} \Rightarrow \frac{\mathbb{P}(Y = y_1)}{\mathbb{P}(Y = y_C)} = \frac{\alpha e^{-\lambda \cdot 1}}{\alpha e^{-\lambda \cdot C}} \quad \Rightarrow \quad \lambda = \frac{\ln \rho}{C - 1} \tag{A.2}$$

Considering the normalization of probability density, we have:

$$\begin{aligned} \int_{y_i \in \mathcal{Y}} \mathbb{P}(Y = y_i) dy_i = \int_1^C \alpha e^{-\lambda y_i} dy_i \equiv 1 \\ \Rightarrow -\frac{\alpha}{\lambda} e^{-\lambda y_i} \Big|_1^C \equiv 1 \Rightarrow \frac{\alpha}{\lambda} \left(e^{-\lambda} - e^{-\lambda C} \right) \equiv 1 \\ \Rightarrow \alpha = \frac{\lambda}{e^{-\lambda} - e^{-\lambda C}} \end{aligned} \tag{A.3}$$

Hence, we can express the distribution of LT dataset $\mathcal{D}_{train}$, i.e., $\mathbb{P}(Y = y_i)$ represents the probability of class y_i in $\mathcal{D}_{train}$, which can be calculated as follows:

$$\mathbb{P}(Y = y_i) = \frac{\iint_{x_i \in \mathcal{X}, y_j \in \mathcal{Y}} \mathbb{1}(X = x_i, Y = y_i) dx_i dy_j}{\iint_{x_i \in \mathcal{X}, y_j \in \mathcal{Y}} \mathbb{1}(X = x_i, Y = y_j) dx_i dy_j} = \frac{\lambda}{e^{-\lambda} - e^{-\lambda C}} e^{-\lambda y_i}, y_i \in [1, C] \tag{A.4}$$

Notice that Eq.A.4 is determined by total class number C and the imbalance factor ρ.

A.2 Proof of Corollary 1

Corollary 1 *When $\boldsymbol{\xi \sim Beta(\alpha, \alpha)}, \alpha \in [0, 1]$, the newly mixed dataset $\mathcal{D}_\nu$ composed of ξ-Aug samples $(\widetilde{x}_{i,j}, \widetilde{y}_{i,j})$ follows the same long-tailed distribution as the origin dataset $\mathcal{D}_{train}$, where (x_i, y_i) and (x_j, y_j) are **randomly** sampled from $\mathcal{D}_{train}$.*

$$\begin{aligned} \mathbb{P}_{mixup}(Y^* = y_i) &= \mathbb{P}^2(Y = y_i) + \mathbb{P}(Y = y_i) \iint_{y_i \neq y_j} Beta(\alpha, \alpha) \mathbb{P}(Y = y_j) d\xi dy_j \\ &= \frac{\lambda}{e^{-\lambda} - e^{-\lambda C}} e^{-\lambda y_i}, y_i \in [1, C] \end{aligned} \tag{A.5}$$

Proof A.1 *Follow the Basic Setting 1, when mixing factor $\xi \sim Beta(\alpha, \alpha)$, consider a ξ-Aug sample generated by $\widetilde{x}_{i,j} = \xi \cdot x_i + (1 - \xi) \cdot x_j$ and $\widetilde{y}_{i,j} = \xi \cdot y_i + (1 - \xi) \cdot y_j$. The ξ-Aug sample $\widetilde{x}_{i,j}$ contributes to class y_i when both pair samples (x_i, y_i) and (x_j, y_j) are in class y_i or one of them are in other labels while the mixing factor ξ is in favor of class y_i.*

$$\begin{aligned} \mathbb{P}(\tilde{y}_{i,j} = y_i) = \mathbb{P}(\tilde{y}_{i,j} = y_i) \cdot \mathbb{P}(\tilde{y}_{i,j} = y_i) \mathbb{P}_\xi(0 \leq \xi \leq 1) \\ + \mathbb{P}(\tilde{y}_{i,j} = y_i) \cdot \mathbb{P}(\tilde{y}_{i,j} \neq y_i) \mathbb{P}_\xi(\xi \geq 0.5) \\ + \mathbb{P}(\tilde{y}_{i,j} \neq y_i) \cdot \mathbb{P}(\tilde{y}_{i,j} = y_i) \mathbb{P}_\xi(\xi < 0.5) \end{aligned} \tag{A.6}$$

Hence, the distribution of new dataset D_ν is:

$$\begin{aligned}
\mathbb{P}_{mixup}(Y=y_i) &= \mathbb{P}(Y=y_i)\cdot\mathbb{P}(Y=y_i)\cdot\int_{0\le\xi\le 1} Beta(\alpha,\alpha)d\xi \\
&\quad + \mathbb{P}(Y=y_i)\cdot\int_{y_j\neq y_i}\int_{\xi\ge 0.5} Beta(\alpha,\alpha)\cdot\mathbb{P}(Y=y_j)d\xi dy_j \\
&\quad + \int_{y_j\neq y_i}\int_{\xi<0.5} Beta(\alpha,\alpha)\cdot\mathbb{P}(Y=y_j)d\xi dy_j\cdot\mathbb{P}(Y=y_i) \\
&= \mathbb{P}^2(Y=y_i) + 0.5\cdot\mathbb{P}(Y=y_i)\cdot(1-\mathbb{P}(Y=y_i)) \\
&\quad + 0.5\cdot(1-\mathbb{P}(Y=y_i))\cdot\mathbb{P}(Y=y_i) \\
&= \mathbb{P}^2(Y=y_i) + 0.5\cdot\mathbb{P}(Y=y_i) - 0.5\cdot\mathbb{P}^2(Y=y_i) \\
&\quad + 0.5\cdot\mathbb{P}(Y=y_i) - 0.5\cdot\mathbb{P}^2(Y=y_i) \\
&= \mathbb{P}(Y=y_i) = \frac{\lambda}{e^{-\lambda}-e^{-\lambda C}}e^{-\lambda y_i}
\end{aligned} \tag{A.7}$$

According to Eq.A.4 and Eq.A.7, the ξ-Aug samples in mixed dataset $\mathcal{D}_\nu$ generated by mixup follow the same distribution of the original long-tailed one. Therefore, the head gets more regulation than the tail. One the one hand, the classification performance will be promoted. On the other hand, however, the performance gap between the head and tail still exists.

A.3 Proof of Corollary 2

Corollary 2 *When $\boldsymbol{\xi^*_{i,j}\sim\mathscr{U}(\pi_{y_i},\pi_{y_j},\alpha,\alpha)}, \alpha\in[0,1]$, the newly mixed dataset $\mathcal{D}_\nu$ composed of ξ-Aug samples $(\widetilde{x}_{i,j},\widetilde{y}_{i,j})$ follows a middle-majority distribution, where (x_i,y_i) and (x_j,y_j) are both **randomly** sampled from $\mathcal{D}_{train}$.*

$$\begin{aligned}
\mathbb{P}^*_{mixup}(Y^*=y_i) &= \mathbb{P}(Y=y_i)\int_{y_j<y_i}\mathbb{1}\left(\int\xi^*_{i,j}\mathscr{U}(\pi_i,\pi_j,\alpha,\alpha)d\xi^*_{i,j}\ge 0.5\right)\mathbb{P}(Y=y_j)dy_j \\
&= \frac{\lambda}{\left(e^{-\lambda}-e^{-C\lambda}\right)^2}\left(e^{-\lambda(y_i+1)}-e^{-2\lambda y_i}\right), y_i\in[1,C]
\end{aligned} \tag{A.8}$$

Proof A.2 *Follow the settings of Proof A.2 and Eq.6, we can get the following relationship of the label y_i and y_j with UniMix Factor $\xi^*_{i,j}$. It easy to know $\pi_{y_i}\le\pi_{y_j}$ if the index $i\ge j$ for the reason that class y_j occupies more instances than class y_i in long-tailed distribution described as Eq.A.4. Under this circumstances, if consider UniMix Factor $\xi^*_{i,j}$ as a constant intuitively, we can deduce that $\xi_{i,j}=\pi_{y_j}/(\pi_{y_i}+\pi_{y_j})\ge 0.5$, i.e.,*

$$i\ge j \quad\Rightarrow\quad \pi_{y_i}\le\pi_{y_j} \quad\Rightarrow\quad \xi_{i,j}=\frac{\pi_{y_i}}{\pi_{y_i}+\pi_{y_j}}\ge 0.5 \tag{A.9}$$

To improve the robustness and generalization, we consider $\xi^*_{i,j}\sim\mathscr{U}(\pi_{y_i},\pi_{y_j},\alpha,\alpha)$, which extends $\xi^*_{i,j}$ to $\pi_{y_j}/(\pi_{y_i}+\pi_{y_j})$ and its vicinity. Hence we can generalize Eq.A.9 to:

$$y_i\ge y_j \quad\Rightarrow\quad \pi_{y_i}\le\pi_{y_j} \quad\Rightarrow\quad \xi_{i,j}\ge 0.5\Rightarrow\int\xi^*_{i,j}\mathscr{U}(\pi_i,\pi_j,\alpha,\alpha)d\xi^*_{i,j}\ge 0.5 \tag{A.10}$$

Given any $i,j\in[1,C]$, the $\widetilde{x}_{i,j}$ will tend to be a ξ-Aug sample of the class y_i if $i\ge j$ for the tail-favored mixing factor. In other words, $\widetilde{x}_{i,j}$ will be a ξ-Aug sample for class y_i with the mean of

$\xi^*_{i,j}$ to be $\pi_{y_j}/(\pi_{y_i}+\pi_{y_j})$. Hence, the probability that $\widetilde{x}_{i,j}$ belongs to class y_i is:

$$\begin{aligned}
\mathbb{P}^*_{mixup}(Y=y_i) &= \mathbb{P}(Y=y_i)\cdot \mathbb{1}\left(\int \xi^*_{i,j}\mathscr{U}(\pi_i,\pi_j,\alpha,\alpha)d\xi^*_{i,j}\geq 0.5\right)\cdot \mathbb{P}(Y<y_i) \\
&= \mathbb{P}(Y=y_i)\int_{y_j<y_i}\mathbb{1}\left(\int \xi^*_{i,j}\mathscr{U}(\pi_i,\pi_j,\alpha,\alpha)d\xi^*_{i,j}\geq 0.5\right)\mathbb{P}(Y=y_j)dy_j \\
&= \mathbb{P}(Y=y_i)\cdot\int_1^{y_i}\mathbb{P}(Y=y_j)dy_j \\
&= \frac{\lambda}{e^{-\lambda}-e^{-\lambda C}}e^{-\lambda y_i}\cdot\int_1^{y_i}\frac{\lambda}{e^{-\lambda}-e^{-\lambda C}}e^{-\lambda y_j}dy_j \\
&= \frac{\lambda}{e^{-\lambda}-e^{-\lambda C}}e^{-\lambda y_i}\cdot -\frac{\frac{\lambda}{e^{-\lambda}-e^{-\lambda C}}}{\lambda}e^{-\lambda y_j}\Big|_1^{y_i} \\
&= \frac{\lambda}{\left(e^{-\lambda}-e^{-C\lambda}\right)^2}\left(e^{-\lambda(y_i+1)}-e^{-2\lambda y_i}\right)
\end{aligned} \tag{A.11}$$

According to Eq.A.11, it's easy to find a derivative zero point in range [1,C]. Hence, the newly distributed dataset $\mathcal{D}_\nu$ generated by mixup with UniMix Factor $\xi^*_{i,j}$ follows a middle-majority distribution that most data concentrates on middle classes.

A.4 Proof of Corollary 3

Corollary 3 *When $\boldsymbol{\xi^*_{i,j}\sim\mathscr{U}(\pi_{y_i},\pi_{y_j},\alpha,\alpha)}, \alpha\in[0,1]$, the newly mixed dataset $\mathcal{D}_\nu$ composed of ξ-Aug samples $(\widetilde{x}_{i,j},\widetilde{y}_{i,j})$ follows a tail-majority distribution, where (x_i,y_i) is **randomly** and (x_j,y_j) is **inversely** sampled from $\mathcal{D}_{train}$, respectively.*

$$\begin{aligned}
\mathbb{P}_{UniMix}(Y^*=y_i) &= \mathbb{P}(Y=y_i)\int_{y_j<y_i}\mathbb{1}\left(\int \xi^*_{i,j}\mathscr{U}(\pi_i,\pi_j,\alpha,\alpha)d\xi^*_{i,j}\geq 0.5\right)\mathbb{P}_{inv}(Y=y_j)dy_j \\
&= \frac{\lambda}{\left(e^{-\lambda}-e^{-C\lambda}\right)\left(e^{-C\tau\lambda}-e^{-\tau\lambda}\right)}\left(e^{-\lambda y_i(\tau+1)}-e^{-\lambda(\tau+y_i)}\right), y_i\in[1,C]
\end{aligned} \tag{A.12}$$

Proof A.3 *Follow the Basic Setting of Proof.1, suppose (x_i,y_i) is randomly sampled from $\mathcal{D}_{train}$, while (x_j,y_j) is inversely sampled from $\mathcal{D}_{train}$ related to τ. i.e., the probability of $\mathbb{P}(Y=y_i)$ is:*

$$\mathbb{P}(Y=y_i)=\frac{\lambda}{e^{-\lambda}-e^{-\lambda C}}e^{-\lambda y_i} \tag{A.13}$$

The probability of UniMix Sampler that the sampled class Y belongs to y_i is $\mathbb{P}_{inv}(Y=y_j)$:

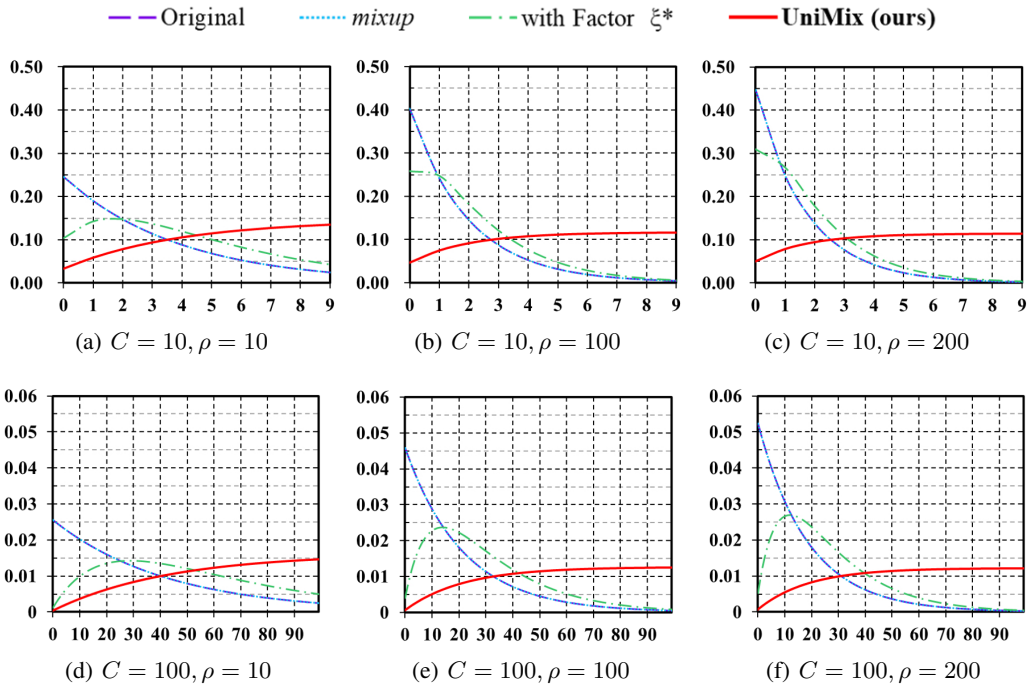
$$\begin{aligned}
\mathbb{P}_{inv}(Y=y_j) &= \frac{\mathbb{P}^\tau(Y=y_j)}{\int_{y_k\in\mathcal{Y}}\mathbb{P}^\tau(Y=y_k)dy_k} \\
&= \frac{\left(\frac{\lambda}{e^{-\lambda}-e^{-\lambda C}}e^{-\lambda y_j}\right)^\tau}{\int_{y_k\in\mathcal{Y}}\left(\frac{\lambda}{e^{-\lambda}-e^{-\lambda C}}e^{-\lambda y_k}\right)^\tau dy_k} \\
&= \frac{e^{-\lambda\tau y_j}}{\int_1^C e^{-\lambda\tau y_k}dy_k} = \frac{\lambda\tau e^{-\lambda\tau y_j}}{e^{-\lambda\tau}-e^{-\lambda\tau C}}
\end{aligned} \tag{A.14}$$

Follow the same proof as Proof.A.3, the probability that $\widetilde{x}_{i,j}$ belongs to class y_i is:

$$\begin{aligned}
\mathbb{P}_{\textit{UniMix}}(Y=y_i) &= \mathbb{P}(Y=y_i)\cdot \mathbb{1}\left(\int \xi^*_{i,j}\mathscr{U}(\pi_i,\pi_j,\alpha,\alpha)d\xi^*_{i,j}\geq 0.5\right)\cdot \mathbb{P}_{inv}(Y<y_i)\\
&= \mathbb{P}(Y=y_i)\int_{y_j<y_i}\mathbb{1}\left(\int \xi^*_{i,j}\mathscr{U}(\pi_i,\pi_j,\alpha,\alpha)d\xi^*_{i,j}\geq 0.5\right)\mathbb{P}_{inv}(Y=y_j)dy_j\\
&= \mathbb{P}(Y=y_i)\cdot\int_1^{y_i}\mathbb{P}_{inv}(Y=y_j)dy_j\\
&= \frac{\lambda}{e^{-\lambda}-e^{-\lambda C}}e^{-\lambda y_i}\cdot\int_1^{y_i}\frac{\lambda\tau e^{-\lambda\tau y_j}}{e^{-\lambda\tau}-e^{-\lambda\tau C}}dy_j\\
&= \frac{\lambda}{e^{-\lambda}-e^{-\lambda C}}e^{-\lambda y_i}\cdot\frac{-e^{-\lambda\tau y_j}}{e^{-\lambda\tau}-e^{-\lambda\tau C}}\Bigg|_1^{y_i}\\
&= \frac{\lambda}{(e^{-\lambda}-e^{-C\lambda})(e^{-C\tau\lambda}-e^{-\tau\lambda})}\left(e^{-\lambda y_i(\tau+1)}-e^{-\lambda(\tau+y_i)}\right)
\end{aligned} \tag{A.15}$$

According to Eq.A.15, $\mathbb{P}_{\textit{UniMix}}(Y=y_i)$ is a gently increasing function in range [1,C]. Hence, the newly mixed dataset $\mathcal{D}_\nu$ generated by UniMix follows a tail-majority distribution that adequate data concentrates on tail classes.

A.5 More visualization of Corollary 1,2,3



(a) $C=10,\rho=10$ (b) $C=10,\rho=100$ (c) $C=10,\rho=200$

(d) $C=100,\rho=10$ (e) $C=100,\rho=100$ (f) $C=100,\rho=200$

Figure A1: Additional visualized comparisons of ξ-*Aug* samples distribution in Corollary 1,2,3. x-axis: class indices. y-axis: probability of each class. *mixup* (blue) exhibits the same LT distribution as origin (purple). UniMix Factor (green) alleviates such situation and the full pipeline (τ=-1) constructs a more uniform distribution of ξ-*Aug* (red), which contributes to a well-calibrated model.

We present additional visualized distribution of ξ-*Aug* samples with different sample strategies for comprehensive comparisons, including $\rho\in\{10,100,200\}$ and $C\in\{10,100\}$. As illustrated in Fig.A1, when the class number C and imbalance factor ρ get larger, the limitations of *mixup* in LT scenarios gradually appear. The VRM dataset $\mathcal{D}_\nu$ generated by *mixup* with naïve mixing factor and random sampler will make $\mathcal{D}_\nu$ following the same LT distribution. The newly mixed pseudo data by

mixup follows the head-majority distribution. It has limited contribution for the tail class' feature learning and regulation, which is the reason for its poor *calibration*.

In contrast, the proposed UniMix Factor $\xi^*_{i,j}$ significantly promotes such situation and improves tail class's feature learning by the preference on the relatively fewer classes of the two samples in a pair. Because of the random sampler, the samples are still mainly from the head, and the newly mixed dataset will follows a middle-majority distribution. Most data concentrates on the middle of classes as green line shows. As C and ρ get larger, the middle distribution will get close to the head. Thanks to the UniMix Sampler that inversely draws data from $\mathcal{D}_{train}$, the D_ν is mainly composed of the head-tail pairs that contribute to the feature learning for the tail when integrated with UniMix Factor. As a result, D_ν follows the tail-majority distribution that improves the generalization on the tail. Specifically, when C and ρ get larger, the distribution of D_ν generated by UniMix still maintains satisfactory tail-majority distribution.

B Missing Proofs and derivations of Bayias

B.1 Proof of Bayias

Theorem B.1 *For classification, let $\psi(x;\theta,W,b)$ be a hypothesis class of neural networks of input $X = x$, the classification with Softmax should contain the influence of prior, i.e., the predicted label during training should be:*

$$\hat{y} = \arg\max_{y_i \in \mathcal{Y}} \frac{e^{\psi(x;\theta,W,b)_{y_i} + \mathbf{\log(\pi_{y_i})} + \mathbf{\log(C)}}}{\sum_{y_j \in \mathcal{Y}} e^{\psi(x;\theta,W,b)_{y_j} + \mathbf{\log(\pi_{y_j})} + \mathbf{\log(C)}}} \tag{B.1}$$

Proof B.1 *Generally, a classifier can be modeled as:*

$$\hat{y} = \arg\max_{y_i \in \mathcal{Y}} \frac{e^{\sum_{d_i \in D}[(W^T)^{(d_i)}_{y_i} \mathcal{F}(x;\theta)^{(d_i)}] + b_{y_i}}}{\sum_{y_j \in \mathcal{Y}} e^{\sum_{d_i \in D}[(W^T)^{(d_i)}_{y_j} \mathcal{F}(x;\theta)^{(d_i)}] + b_{y_j}}} \triangleq \arg\max_{y_i \in \mathcal{Y}} \frac{e^{\psi(x;\theta,W,b)_{y_i}}}{\sum_{y_j \in \mathcal{Y}} e^{\psi(x;\theta,W,b)_{y_j}}} \tag{B.2}$$

where $\hat{y}$ indicates the predicted label and $\mathcal{F}(x;\theta) \in \mathbb{R}^{D\times 1}$ is the D-dimension feature extracted by the backbone with parameter θ. $W \in \mathbb{R}^{D\times C}$ represents the parameter matrix of the classifier. The model attempts to get the maximum $\hat{y}$ given x, i.e., to maximize the posterior, satisfying the following relationship between the prior and likelihood according to the Bayesian theorem:

$$\begin{aligned}
\hat{y} &= \arg\max_{y_i \in \mathcal{Y}} \mathbb{P}(Y = y_i | X = x) \\
&= \arg\max_{y_i \in \mathcal{Y}} \frac{\mathbb{P}(X = x | Y = y_i) \cdot \mathbb{P}(Y = y_i)}{\sum_k \mathbb{P}(Y = y_k) \prod_j \mathbb{P}(X^{(j)} = x^{(j)} | Y = y_k)} \\
&\propto \arg\max_{y_i \in \mathcal{Y}} \mathbb{P}(X = x | Y = y_i) \cdot \mathbb{P}(Y = y_i)
\end{aligned} \tag{B.3}$$

where $\sum_k \mathbb{P}(Y = y_k) \prod_j \mathbb{P}(X^{(j)} = x^{(j)} | Y = y_k)$ is the normalized evidence factor. $\mathbb{P}(Y = y_i)$ is the prior probability estimated by the instance proportion of each category in the dataset. To maximize posterior, we need to get the $\mathbb{P}(X|Y)$ in an ERM supervised training manner. However, in LT scenarios, the likelihood is consistent in the train and test set, but the prior is different. Hence, we derive the posterior on train and test set separately:

$$\begin{cases}
\hat{y} \propto \arg\max_{y_i \in \mathcal{Y}} \mathbb{P}(X = x | Y = y_i) \cdot \mathbb{P}_{train}(Y = y_i) \\
\hat{y}' \propto \arg\max_{y_i \in \mathcal{Y}} \mathbb{P}(X = x' | Y' = y_i) \cdot \mathbb{P}'_{test}(Y' = y_i)
\end{cases} \tag{B.4}$$

where $\hat{y}, \hat{y}'$ represent the prediction results for the train and test set, respectively. The model $\psi(x;\theta,W,b)$ is just the likelihood estimation $\mathbb{P}(X = x|Y = y_i)$ of the train set. To obtain the posterior probability $y' = \arg\max_{y_i \in \mathcal{Y}} \mathbb{P}'(Y = y_i|X = x')$ for inference, one should consider unifying the optimization direction on the train set and test set:

$$\mathbb{P}_{test}(Y = y_i | X = x') \propto \frac{\mathbb{P}_{train}(Y = y_i | X = x)}{\mathbb{P}_{train}(Y = y_i)} \cdot \mathbb{P}_{test}(Y' = y_i) = \frac{\mathbb{P}_{train}(Y = y_i | X = x)}{C \cdot \pi_{y_i}} \tag{B.5}$$

For the difference of label prior, the learned parameters of the model will also yield class-level bias. Hence, the actual optimization direction is not described as Eq.B.2 because the bias incurred by prior should be compensated at first.

$$
\begin{aligned}
\hat{\theta}, \hat{W}, \hat{b} \triangleq \Theta &= \arg\min_{\Theta} \sum\nolimits_{x_i \in \mathcal{X}' \wedge y_i \in \mathcal{Y}} \mathbb{1}\left(y_i \neq \arg\max_{y_j \in \mathcal{Y}} (\mathbb{P}_{test}(Y = y_j | X = x_i)\right) \\
&\Leftrightarrow \arg\min_{\Theta} \sum\nolimits_{x_i \in \mathcal{X} \wedge y_i \in \mathcal{Y}} \mathbb{1}\left(y_i \neq \arg\max_{y_j \in \mathcal{Y}} \frac{\mathbb{P}_{train}(Y = y_j | X = x_i)}{C \cdot \pi_{y_j}}\right) \\
&= \arg\min_{\Theta} \sum\nolimits_{x_i \in \mathcal{X} \wedge y_i \in \mathcal{Y}} \mathbb{1}\left(y_i \neq \arg\max_{y_j \in \mathcal{Y}} \frac{e^{\psi(x_i;\theta,W,b)_{y_j} - \mathbf{\log C - \log(\pi_{y_j})}}}{\sum_{y_k \in \mathcal{Y}} e^{\psi(x_i;\theta,W,b)_{y_k}}}\right) \\
&\Leftrightarrow \arg\min_{\Theta} \sum\nolimits_{x_i \in \mathcal{X} \wedge y_i \in \mathcal{Y}} \mathbb{1}\left(y_i \neq \arg\max_{y_j \in \mathcal{Y}} \frac{e^{\psi(x_i;\theta,W,b)_{y_j} - \mathbf{\log(\pi_{y_j})} - \log C}}{\sum_{y_k \in \mathcal{Y}} e^{\psi(x_i;\theta,W,b)_{y_k} - \mathbf{\log(\pi_{y_k})} - \log C}}\right)
\end{aligned}
\tag{B.6}
$$

To correct the bias for inferring, the offset term that the model in LT datasets needs to compensate is:

$$
\mathscr{B}_y = log(\pi_y) + log(C) \tag{B.7}
$$

B.2 Proof of classification *calibration*

Theorem B.2 *$\mathscr{B}_y$-compensated cross-entropy loss in Eq.B.8 ensures classification calibration.*

$$
\mathcal{L}_{\mathscr{B}}\left(y_i, \psi(x;\Theta)\right) = -\log \frac{e^{\psi(x;\Theta)_{y_i} + \log\left(\pi_{y_i}\right) + \log(C)}}{\sum_{y_k \in \mathcal{Y}} e^{\psi(x;\Theta)_{y_j} + \log\left(\pi_{y_k}\right) + \log(C)}} \tag{B.8}
$$

Classification *calibration* [18, 58] represents the predicted winning *Softmax* scores indicate the actual likelihood of a correct prediction. The miscalibrated models tend to be overconfident which results in that the minimiser of the expected loss (equally, the empirical risk in the infinite sample limit) can not lead to a minimal classification error. Previous work [47] has introduced a theory to measure the *calibration* of a pair-wise loss:

Lemma B.1 *Pairwise loss $\mathcal{L}(y_i, \psi(x;\Theta)) = \alpha_{y_i} \cdot \log\left[1 + \sum_{y_k \neq y_i} e^{\Delta_{y_i y_k}} \cdot e^{(\psi(x;\Theta)_{y_i} - \psi(x,\Theta)_{y_k})}\right]$ ensures classification calibration if for any $\delta \in \mathbb{R}_+^C$:*

$$
\alpha_{y_i} = \delta_{y_i} / \pi_{y_i} \quad \Delta y_i y_k = \log(\delta_{y_k} / \delta_{y_i}) \tag{B.9}
$$

Proof B.2 *To begin, we rewritten the Bayias-compensated cross-entropy loss into the following equation:*

$$
\begin{aligned}
\mathcal{L}_{\mathscr{B}}\left(y_i, \psi(x;\Theta)\right) &= -\log \frac{e^{\psi(x;\Theta)_{y_i} + \log\left(\pi_{y_i}\right) + \log(C)}}{\sum_{y_k \in \mathcal{Y}} e^{\psi(x;\Theta)_{y_j} + \log\left(\pi_{y_k}\right) + \log(C)}} \\
&\Leftrightarrow \log \frac{\sum_{y_k \in \mathcal{Y}} e^{\psi(x;\Theta)_{y_k} + \log\left(\pi_{y_k}\right) + \log(C)}}{e^{\psi(x;\Theta)_{y_i} + \log\left(\pi_{y_i}\right) + \log(C)}} \\
&= \log\left[1 + \frac{\sum_{y_k \neq y_i} e^{\psi(x;\Theta)_{y_k} + \log\left(\pi_{y_k}\right) + \log(C)}}{e^{\psi(x;\Theta)_{y_i} + \log\left(\pi_{y_i}\right) + \log(C)}}\right] \\
&= \log\left[1 + \sum_{y_k \neq y_i} \frac{e^{\psi(x;\Theta)_{y_k} + \log\left(\pi_{y_k}\right) + \log(C)}}{e^{\psi(x;\Theta)_{y_i} + \log\left(\pi_{y_i}\right) + \log(C)}}\right] \\
&= \log\left[1 + \sum_{y_k \neq y_i} \frac{e^{\log\left(\pi_{y_k}\right) + \log(C)} \cdot e^{\psi(x;\Theta)_{y_k}}}{e^{\log\left(\pi_{y_i}\right) + \log(C)} \cdot e^{\psi(x;\Theta)_{y_i}}}\right] \\
&= \log\left[1 + \sum_{y_k \neq y_i} e^{\mathscr{B}_{y_k} - \mathscr{B}_{y_i}} \cdot e^{\psi(x;\Theta)_{y_k} - \psi(x;\Theta)_{y_i}}\right]
\end{aligned}
\tag{B.10}
$$

Compare Eq.B.10 with LemmaB.1, observed that when $\delta_y = \pi_y$.

$$\begin{cases} \alpha_{y_i} = \delta_{y_i}/\pi_{y_i} = \pi_{y_i}/\pi_{y_i} = 1 \\ \Delta_{y_i y_k} = \mathscr{B}_{y_k} - \mathscr{B}_{y_i} \\ \qquad\quad = [log(\pi_{y_k}) + log(C)] - [log(\pi_{y_i}) + log(C)] \\ \qquad\quad = log(\pi_{y_k}) - log(\pi_{y_i}) = log(\pi_{y_k}/\pi_{y_i}) \end{cases} \tag{B.11}$$

According to LemmaB.1, we immediately deduce that Bayias-compensated cross-entropy loss ensures classification calibration.

B.3 Comparisons with other losses

Previous work [34, 14, 56, 62, 8, 47] adjusts the logits *weight* or *margin* on standard *Softmax* cross-entropy (CE) loss to tackle the long-tailed datasets. We summarize the loss modification methods reported in Tab.1 and discuss the difference between theirs and ours here.

Weight-wise losses. Focal loss [41] is proposed to balance the positive/negative samples during object detection and extends for classification by assigning low weight loss for easy samples. It re-wights the loss with a factor $(1 - p_{y_i})^\gamma$ on standard cross-entropy loss, where p_{y_i} is the prediction probability of class y_i:

$$\mathcal{L}_{Focal} = -(1 - p_{y_i})^\gamma \log \frac{e^{\psi(x;\Theta)_{y_i}}}{\sum_{y_k \in \mathcal{Y}} e^{\psi(x;\Theta)_{y_k}}} \tag{B.12}$$

CB loss is proposed by Cui *et al.* [14] with the *effective number*. It adopts a coefficient $(1 - \beta)/(1 - \beta^{n_{y_i}})$ for standard cross-entropy loss:

$$\mathcal{L}_{CB} = -\frac{1 - \beta}{1 - \beta^{n_{y_i}}} \log \frac{e^{\psi(x;\Theta)_{y_i}}}{\sum_{y_k \in \mathcal{Y}} e^{\psi(x;\Theta)_{y_k}}} \tag{B.13}$$

CDT loss [62] re-weights each *Softmax* logit with a temperature scale factor $a_{y_i} = (\frac{n_{max}}{n_{y_i}})^\gamma$. It artificially reduces the decision values for head classes:

$$\mathcal{L}_{CDT} = -\log \frac{e^{\psi(x;\Theta)_{y_i}/(\frac{n_{max}}{n_{y_i}})^\gamma}}{\sum_{y_k \in \mathcal{Y}} e^{\psi(x;\Theta)_{y_k}/(\frac{n_{max}}{n_{y_k}})^\gamma}} \tag{B.14}$$

All above re-weight methods are proven effective empirically, more or less. However, such approaches will confront the coverage dilemma when the train data gets highly imbalanced. The weights related to instances number may be large in this situation and result in unstable gradients that deteriorate head classes' performance gain. They are also sensitive to hyper-parameters, which makes it hard to adopt in varied datasets.

Margin-wise losses. Previous work in Deep Metrics Learning [44, 15, 60, 43] attempts to obtain better inter-class and intra-class distance from the margin perspective. Cao *et al.* [8] analyze the optimal margin between two different classes via generalization error bounds in the long tail visual recognition. They propose the LDAM loss to encourage the tail classes to enjoy larger margins. In detail, a label-aware margin $C/n_{y_i}^{1/4}$ is added to the groud truth logit where C is an independent constant:

$$\mathcal{L}_{LDAM} = -\log \frac{e^{\psi(x;\Theta)_{y_i} - C/n_{y_i}^{1/4}}}{e^{\psi(x;\Theta)_{y_i} - C/n_{y_i}^{1/4}} + \sum_{y_k \neq y_i} e^{\psi(x;\Theta)_{y_k}}} \tag{B.15}$$

Logit Adjustment [47] loss is proposed to overcome the long-tailed dataset motivated by the balanced error rate. They suppose that the model shows similar performance on each class in the validation dataset. The authors propose a margin $\tau \log(\pi_{y_i})$ on standard *Softmax* cross-entropy loss. Different from the LDAM loss, such margin is added for all logits with label priors π_y:

$$\mathcal{L}_{LA} = -\log \frac{e^{\psi(x;\Theta)_{y_i} + \tau \log(\pi_{y_i})}}{\sum_{y_k \in \mathcal{Y}} e^{\psi(x;\Theta)_{y_k} + \tau \log(\pi_{y_k})}} \tag{B.16}$$

Our Bayias-compensated CE loss is motivated by erasing the difference of label *prior* in the train set and test set based on the Bayesian theory. The network parameters are only the *likelihood* estimation, while we need a reliable *posterior* for unbiased inference. Hence, we compensate the bias incurred by various label *prior* via adding a margin related to the train set label *prior* and minus a margin related to the test set label *prior*:

$$\mathcal{L}_{ours} = -\log \frac{e^{\psi(x;\Theta)_{y_i}+\log(\pi_{y_i})-\log(1/C)}}{\sum_{y_k\in\mathcal{Y}} e^{\psi(x;\Theta)_{y_k}+\log(\pi_{y_k})-\log(1/C)}} \tag{B.17}$$

One may notice that our loss is similar to LA loss with an extra constant margin $-\log(1/C)$ when the $\tau = 1$ in LA loss. However, the difference lies in two aspects: 1) the margin $-\log(1/C)$ is a positive value and makes all logits get larger, which makes *Softmax* operation more distinguishable. 2) Notice our two margins are related to the label *prior* of the train set and test set. Hence, when the train set is balanced, i.e., $\pi_{y_i} = 1/C$, the margin will be $\log(1/C) - \log(1/C) \equiv 0$, and our loss coverage to the standard cross-entropy loss. Furthermore, our loss can handle the situation that both train set and test set are imbalanced distribution via directly setting the margin as $\log(\pi_{y_i}) - \log(\pi'_{y_i})$, where π_{y_i} is the label *prior* in the train set and π'_{y_i} is the label *prior* in test set correspondingly.

C Implement detail

We conduct experiments on CIFAR-10-LT [38], CIFAR-100-LT [38], ImageNet-LT [53], and iNaturalist 2018 [28]. We adopt long-tailed version CIFAR datasets which are build by suitably discarding training instances following the Exp files given in [14, 8]. The instance number of each class exponentially decays in train dataset and keeps balanced during validation process. Here, an imbalance factor ρ is define as n_{max}/n_{min} to measure how imbalanced the dataset is. ImageNet-LT is the LT version of ImageNet. It samples instances for each class following *Pareto* distribution which is also long-tailed. iNaturalist 2018 is a real-world dataset with $8,142$ classes and $\rho = 500$, which suffers extremely class-imbalanced and fine-grained problems.

C.1 Implement details on CIFAR-LT

ResNet-32 is the backbone on CIFAR-LT. For all CIFAR-10-LT and CIFAR-100-LT, the universal data augmentation strategies [24] are used for training. Specifically, we pad 4 pixels on each side and crop a 32×32 region. The cropped regions are flipped horizontally with 0.5 probability and normalized by the mean and standard deviation for each color channel. We train the model via stochastic gradient decent (SGD) with momentum 0.9 and weight decay of 2×10^{-4} for all experiments. All models are trained for 200 epochs setting mini-batch as 128 with same learning rate [8] for fair comparisons. The learning rate is set as [8]: the initial value is 0.1 and linear warm up at first 5 epochs. The learning rate is decayed at the 160^{th} and 180^{th} epoch by 0.01. For all *mixup* [64] and its extension methods [59, 12], we fine-tune the model after 120^{th} epoch. The α of Beta distribution is 0.5 for UniMix and 1.0 for others.

C.2 Implement details on large-scale datasets

We adopt ResNet-10 & ResNet-50 for ImageNet-LT and ResNet-50 for iNaturalist 2018. For ImageNet-LT, ResNet-10 and ResNet-50 are trained from scratch for 90 epochs, setting mini-batch as 512 and 64, respectively. For iNaturalist 2018, we adopt the vanilla ResNet-50 with mini-batch as 512 and train for 90 epochs. With proper data augment strategy [69], the SGD optimizer with momentum 0.9 and weight decay 1×10^{-4} optimizes the model with same learning rate for fair comparisons [8, 69, 33]. We set base learning rate as 0.2 and decay it at the 30^{th} and 60^{th} epoch by 0.1, respectively.

D Additional Experiment Results

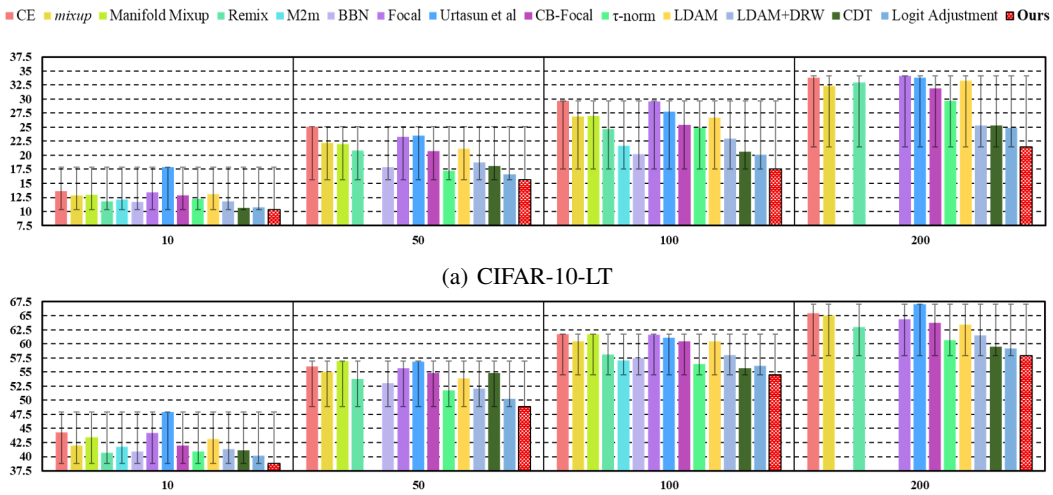
We present additional experiments for comprehensive comparisons with previous methods and make a detailed analysis, which can be summarized as follows:

- Visualized top-1 validation error (%) comparisons of previous methods.

- Additional quantity and visualized comparisons of classification *calibration*.
- Additional visualized comparisons of confusion matrix and log-confusion matrix.
- Additional comparisons on test imbalance scenarios.
- Additional comparisons on state-of-the-art two-stage method.

D.1 Visualized comparisons on CIFAR-10-LT and CIFAR-100-LT

Previous sections have shown the remarkable performance of the proposed UniMix and Bayias. Fig.D1 shows the visualized top-1 validation error rate (%) comparisons on CIFAR-10-LT and CIFAR-100-LT with $\rho \in \{10, 50, 100, 200\}$ for clear and comprehensive comparisons. The histogram indicates the value of each method. The positive error term represents its distance towards the best method, while the negative term indicates the advance towards the worst one.



(a) CIFAR-10-LT

(b) CIFAR-100-LT

Figure D1: Visualized top1 error (%) (y-axis) comparisons of CIFAR-10-LT and CIFAR-100-LT on different ρ (x-axis) in ResNet-32. The proposed method (red) achieves the lowest error rate compared with other methods. The leading advantage of our method increases consistently as imbalance factor gets larger.

Results in Fig.D1 show that the proposed method outperforms others with lower error rate over all imbalance factors settings. As the dataset gets more skewed and imbalanced, the advantage of our method gradually emerges. On the one hand, the proposed UniMix generates a tail-majority pseudo dataset favoring the tail feature learning, which makes the model achieve better *calibration*. It is practical to improve the generalization of all classes and avoid potential over-fitting and under-fitting risks. On the other hand, the proposed Bayias overcomes the bias caused by existing *prior* differences, which improves the model's performance on the balanced validation dataset. Even in extremely imbalanced scenarios (e.g., CIFAR-100-LT-200), the proposed method still achieves satisfactory performance.

D.2 Additional quantitative and qualitative *calibration* comparisons

D.2.1 Definition of *calibration*

Calibration means a model's predicting probability can estimate the representative of the actual correctness likelihood, which is vital in many real-world decision-making applications [52, 9, 4]. Suppose a dataset contains N samples $\mathcal{D} := \{(x_k, y_k)\}_{k=1}^{N}$, where x_i and y_i represent the i^{th} sample and its corresponding label, respectively. Let $\hat{p}_{y_i} = \mathbb{P}(Y = y_i | x = x_i)$ be the confidence of the predicted label, and divide dataset $\mathcal{D}$ into a mini-batch size m with $M = N/m$ in total. The accuracy

(acc) and the confidence (cfd) in a mini-batch $\mathcal{B}_m$ are:

$$\begin{aligned} acc(\mathcal{B}_m) &= \frac{1}{m}\sum\nolimits_{i\in\mathcal{B}_m} \mathbb{1}(\hat{y}_i = y_i) \\ cfd(\mathcal{B}_m) &= \frac{1}{m}\sum\nolimits_{i\in\mathcal{B}_m} \hat{p}_{y_i} \end{aligned} \tag{D.1}$$

According to Eq.D.1, a perfectly calibrated model will strictly have $acc(\mathcal{B}_m) \equiv cfd(\mathcal{B}_m)$ for all $m \in \{1, \cdots, M\}$. Hence, the Expected Calibration Error (ECE) is proposed as a scalar statistic of *calibration* to quantitatively measure classifiers' mean distance to the ideal $acc(\mathcal{B}_m) \equiv cfd(\mathcal{B}_m)$, which is defined as:

$$ECE = \sum\nolimits_{m=1}^{M} \frac{|\mathcal{B}_m|}{n} \left|acc(\mathcal{B}_m) - cfd(\mathcal{B}_m)\right| \tag{D.2}$$

In addition, reliable confidence measures are essential in high-risk applications. The Maximum Calibration Error (MCE) describes the worst-case deviation between confidence and accuracy, which can be defined as:

$$MCE = \max_{m\in\{1,\cdots,M\}} \left|acc(\mathcal{B}_m) - cfd(\mathcal{B}_m)\right| \tag{D.3}$$

According to Eq.D.2,D.3, a well-calibrated model should coverage ECE and MCE to 0, indicating that the prediction score reflects the actual accuracy *likelihood*. Under this circumstances, the calibrated model will show excellent robustness and generalization, especially in LT scenarios. Although the authors in [18] propose the temperature scaling as a post-hoc method to adjust the classifier, our motivation is to train a calibrated model end to end without loss of accuracy.

Previous work [2] points out that ECE scores suffer from several shortcomings. Hence, additional metrics [49, 2] are proposed to show more robustness, e.g., Adaptivity & Adaptive Calibration Error (ACE), Thresholding & Thresholded Adaptive Calibration Error (TACE), Static Calibration Error (SCE), and Brier Score (BS). The definition of the above metrics are as follows:

TACE disregards all predicted probabilities that are less than a certain threshold ϵ. It adaptively chooses the bin locations to ensure each bin has the same instance numbers and estimates the miscalibration of probabilities across all classes in the prediction while the ECE only chooses the top-1 predicted class. TACE is defined as Eq.D.4:

$$TACE = \frac{1}{CR}\sum\nolimits_{c=1}^{C}\sum\nolimits_{r=1}^{R} \left|acc(\mathcal{B}_r, c) - cfd(\mathcal{B}_r, c)\right| \tag{D.4}$$

where $acc(\mathcal{B}_r, c)$ and $cfd(\mathcal{B}_r, c)$ are the accuracy and confidence of adaptive calibration range r for class label c, respectively. Calibration range r defined by the $\lfloor N/R \rfloor_{th}$ index of the sorted and thresholded predictions, exclude the prediction less than ϵ. If set $\epsilon = 0$, TACE converts to ACE.

SCE is a simple extension of ECE to every probability in the multi-class setting. SCE bins predictions separately for each class probability, computes the calibration error within the bin, and averages across bins. SCE is defined as Eq.D.5:

$$SCE = \frac{1}{C}\sum\nolimits_{c=1}^{C}\sum\nolimits_{b=1}^{B} \frac{n_{bc}}{N} |acc(\mathcal{B}_b, c) - cfd(\mathcal{B}_b, c)| \tag{D.5}$$

where $acc(\mathcal{B}_b, c)$ and $cfd(\mathcal{B}_b, c)$ are the accuracy and confidence of bin $\mathcal{B}_b$ for class label c, respectively. n_{bc} is the prediction number in bin $\mathcal{B}_b$ for class c. N is the total number of test set.

BS [5] has also been known as a metric for the verification of predicted probabilities. Similarly to the log-likelihood, BS penalizes low probabilities assigned to correct predictions and high probabilities assigned to wrong ones, which is defined as Eq.D.6:

$$BS = \frac{1}{NC}\sum\nolimits_{i=1}^{N}\sum\nolimits_{c=1}^{C} \left(\mathbb{1}(y_i^* = c) - \mathbb{P}(Y = y_c|X = x_i)\right)^2 \tag{D.6}$$

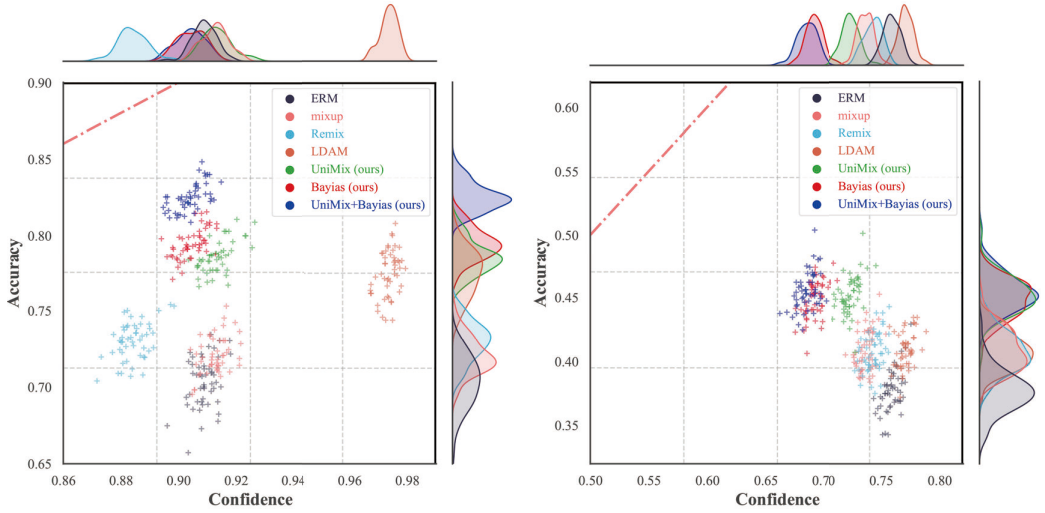
where N represents the total number of test set. y_i^*, $\mathbb{P}(Y = y_c|X = x_i)$ represent the ground truth and predicted label, respectively.

D.2.2 Quantitative results of *calibration* on CIFAR-LT

Besides ECE and MCE results, we additionally provide quantitative metrics results, i.e., ACE, TACE setting threshold $\epsilon = 1e-3$, SCE, and BS. The comparisons are illustrated in Tab.D1.

Table D1: Quantitative calibration metric of ResNet-32 on CIFAR-10/100-LT-100 test set. Smaller ACE, TACE, SCE, and BS indicate better calibration results. Either of the proposed methods achieves a well-calibrated model compared with others. The combination of UniMix and Bayias still achieves the best performance.

Dataset	CIFAR-10-LT-100				CIFAR-100-LT-100			
Calibration Metric ($\times$ 100)	ACE	TACE	SCE	BS	ACE	TACE	SCE	BS
ERM	5.42	4.66	5.46	53.61	0.831	0.709	0.952	97.05
mixup [64]	4.87	4.20	4.92	49.09	0.751	0.683	0.849	89.26
Remix [12]	3.49	3.32	3.77	44.84	0.740	0.675	0.846	89.33
LDAM+DRW [8]	4.14	2.84	4.43	44.76	0.801	0.671	1.093	107.4
UniMix (ours)	3.48	3.27	3.57	38.98	0.505	0.555	0.687	80.70
Bayias (ours)	2.45	2.40	2.69	34.30	0.460	0.518	0.631	78.78
UniMix+Bayias (ours)	**2.31**	**2.17**	**2.43**	**32.84**	**0.450**	**0.517**	**0.623**	**78.24**



(a) CIFAR-10-LT-100 (b) CIFAR-100-LT-100

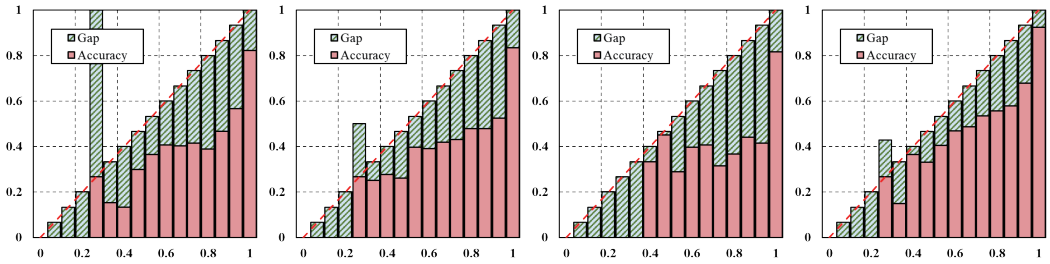
Figure D2: Additional comparisons of joint density plots of accuracy vs. confidence on CIFAR-10-LT and CIFAR-100-LT. A well-calibrated classifier's density will lay around the $y = x$ (red dot line). The combination of UniMix and Bayias achieves remarkable results especially in the severely imbalance scenarios (i.e., $\rho = 100$).

D.2.3 Additional visualized *calibration* comparison on CIFAR-LT

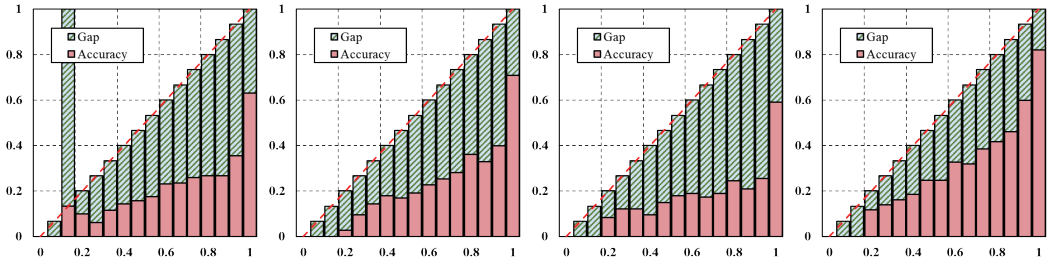
To make intuitive comparisons, we visualize additional confidence-accuracy joint density plots and reliability diagrams in CIFAR-LT test set setting $\rho = 100$. The confidence data is obtained by the average *Softmax* winning score in a test mini-batch [58]. The reliability diagrams data is obtained follow [18], which groups all prediction score into 15 interval bins and calculate the accuracy of each bin. The results are available in Fig.D2.

Fig.D2 clearly shows that the proposed method achieves better *calibration* in the CIFAR-LT test set. Each scatters data is the corresponding result of confidence and accuracy in a mini-batch m. In LT scenarios, the validation accuracy is usually lower than train accuracy, especially for the tail classes. Such overconfidence and miscalibration obstruct the network from having better generalization performance. In Fig.D2, the distribution of scattered points reflects a model's generalization and *calibration*. Our proposed method is the closest towards ideal $y = x$. It means each class gets enough regulation and hence contributes to a well-calibrated model. In contrast, *mixup* and its extensions are similar to ERM, which show limited improvement on classification *calibration* in LT scenarios. LDAM ameliorates the LT situation to some extent but results in more severe overconfidence cases, which may explain why it is counterproductive to other methods.

The reliability diagram is another visual representation of model *calibration*. As Fig.D3 shows, the baseline (ERM) is overconfident in its predictions and the accuracy is generally below ideal $y = x$



(a) ERM (b) *mixup* [64] (c) LDAM+DRW [8] (d) Ours



(e) ERM (f) *mixup* [64] (g) LDAM+DRW [8] (h) Ours

Figure D3: Reliability diagrams of ResNet-32 on CIFAR-10-LT-100 (top) and CIFAR-100-LT-100 (bottom). x-axis and y-axis represent the confidence and accuracy, respectively. A well-calibrated model shows smaller gap in each bin, i.e., the accuracy bar is closed to $y = x$. Our proposed method shows best results compared with others without any further fine-tune.

for each bin. Our method produces much better confidence estimates than others, which means our success in regulating all classes. Although some post-hoc adjustment methods [18, 68] can achieve better classification *calibration*, our approach trains a calibrated model end-to-end, which avoids the potential adverse effects on the original task. Considering the similarity of purposes with MiSLAS [68], we make detailed discussion in Appendix D.5.

D.3 Additional visualized comparisons of confusion matrix on CIFAR-LT

D.3.1 Visualized confusion matrix on CIFAR-10-LT

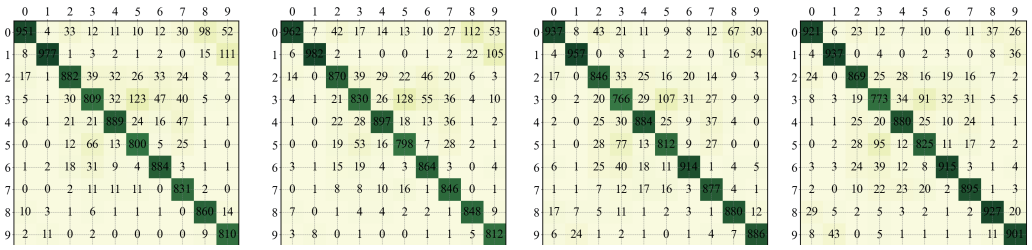
We further give additional visualized results on CIFAR-10-LT setting $\rho = 10$ and $\rho = 100$, which are available in Fig.D4,D5. The results show that all methods perform well compared with ERM in the simple dataset (e.g., CIFAR-10-LT-10, CIFAR-10-LT-100). In general, the proposed method significantly improves the accuracy of the tail for the larger diagonal values. The improvement on the tail is superior than other methods, which leads to state-of-the-art performance.

As for misclassification results, the non-diagonal elements concentrate on the top-right triangle. Hence, most of previous methods have limited improvement on minority classes, which tend to simply predict the instances in tail as majority ones. In contrast, the proposed method improves the performance for the tail and makes the misclassification case more balance distributed.

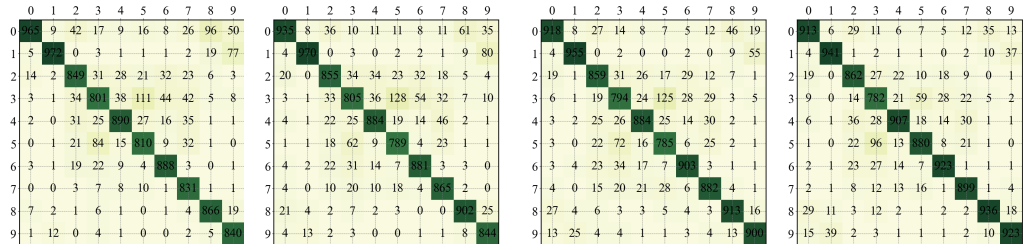
D.3.2 Visualized $\log$-confusion matrix on CIFAR-100-LT

Furthermore, we visualize the distinguishing results of our methods on challenging CIFAR-100-LT for comprehensive comparisons. Specifically, we plot additional comparisons on CIFAR-100-LT-10 (Fig.D6) and CIFAR-100-LT-100 (Fig.D7).

Fig.D6 D7 show that in the challenging LT scenarios, the diagonal values get decreased towards the tail. A proper confusion matrix should exhibit a balanced distribution of misclassification case and large enough diagonal values. The proposed method outperforms others both in the correct cases (the diagonal elements) and the misclassification case distribution. In general, the proposed method shows the best performance, especially in the tail classes (deeper color represents larger value). The

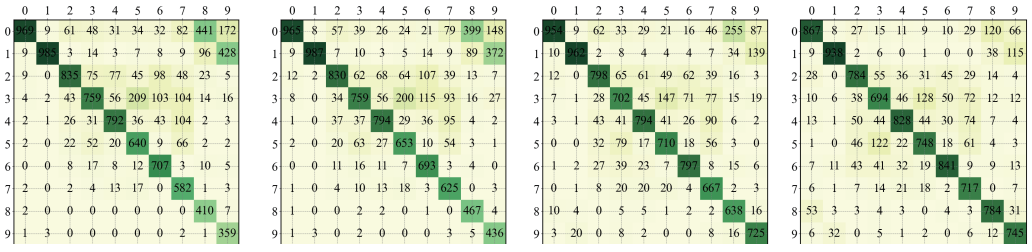


(a) ERM (b) *mixup* [64] (c) LDAM+DRW [8] (d) CDT [62]

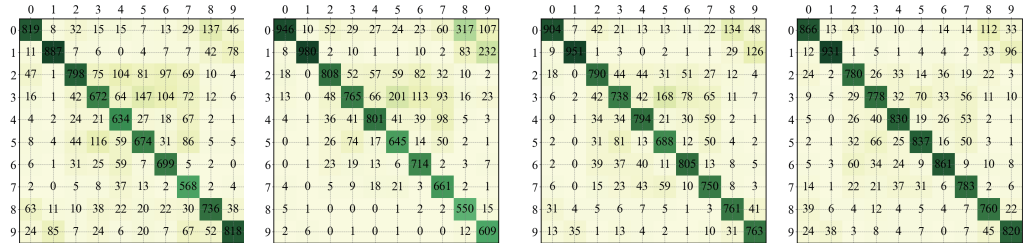


(e) Remix [12] (f) CB+DRW [14] (g) Logit Adjustment [47] (h) Ours

Figure D4: Additional confusion matrix comparisons on CIFAR-10-LT-10. The x-axis and y-axis indicate the ground truth and predicted labels, respectively. Deeper color indicates larger values. All methods achieve satisfactory results while ours further improves the tail feature learning and make misclassification cases more balanced distributed.



(a) ERM (b) *mixup* [64] (c) LDAM+DRW [8] (d) CDT [62]



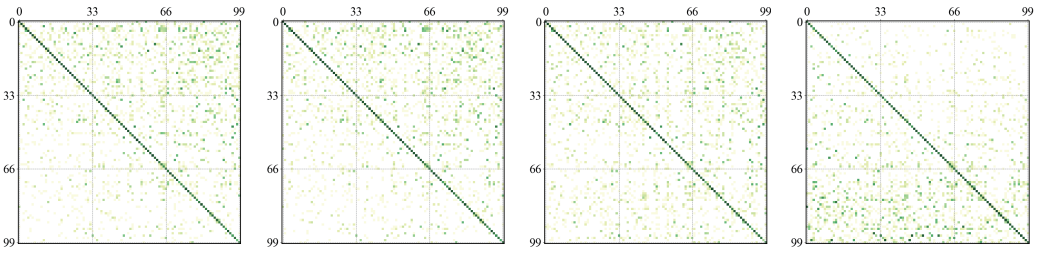
(e) Remix [12] (f) CB+DRW [14] (g) Logit Adjustment [47] (h) Ours

Figure D5: Additional confusion matrix comparisons on CIFAR-10-LT-100. The x-axis and y-axis indicate the ground truth and predicted labels, respectively. Deeper color indicates larger values. The disparity of these methods gradually gets appeared. Our method achieves the best accuracy as well as better non-diagonal distribution. Other methods exhibit an obvious bias towards the head or tail.

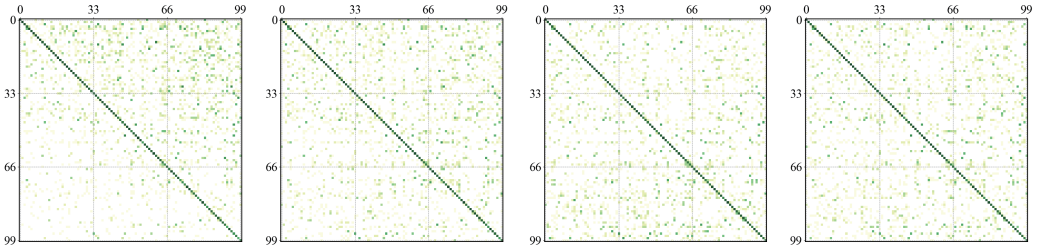
misclassification cases in other methods apparently concentrate on the upper triangular, which means the classifiers simply predict the tail samples as the head and vise versa.

D.4 Results on test imbalance scenarios

Zhang et al. [66] have discussed the existing challenging and severe bias issue in imbalanced distributed test set. When the test set distribution is ideally known or well estimated, Bayias enables to overcome the bias issues in such scenarios by considering the priors. In contrast, CE loss implicitly

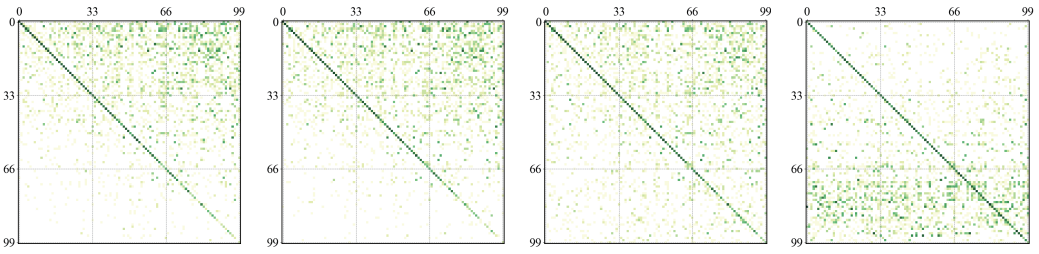


(a) ERM (b) *mixup* [64] (c) LDAM+DRW [8] (d) CDT [62]

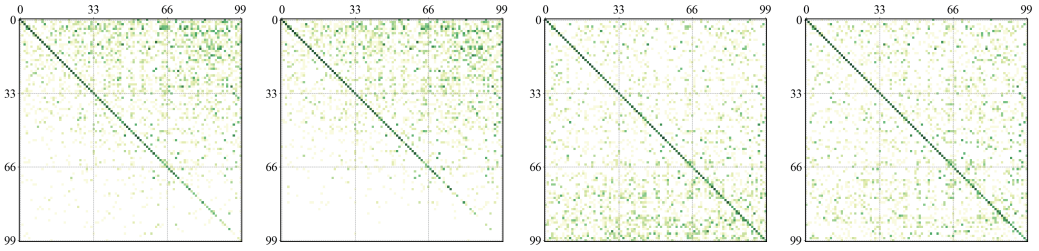


(e) Remix [12] (f) CB+DRW [14] (g) Logit Adjustment [47] (h) Ours

Figure D6: Additional $\log$-confusion matrix comparisons on CIFAR-100-LT-10. The x-axis and y-axis indicate the ground truth and predicted labels, respectively. $\log$ operation is adopted for clearer visualisation. Deeper color indicates larger values. The increased class number makes the misclassification distribution more clearly.



(a) ERM (b) *mixup* [64] (c) LDAM+DRW [8] (d) CDT [62]



(e) Remix [12] (f) CB+DRW [14] (g) Logit Adjustment [47] (h) Ours

Figure D7: Additional $\log$-confusion matrix comparisons on CIFAR-100-LT-100. The x-axis and y-axis indicate the ground truth and predicted labels, respectively. $\log$ operation is adopted for clearer comparisons. Deeper color indicates larger values. Other methods show obviously poor generalization on the tail in the challenging CIFAR-100-LT-100, while the proposed method achieves significantly superior correct classification results and balanced misclassification case distribution.

requires train set and test set same distributed, while LA loss requires test set balanced distributed. LA and CE losses are powerless in other scenarios, while Bayias is flexible and capable with imbalanced distributed test sets, i.e., the constant term $\log C$ can change to be the test set distribution $\log \pi'$ to deal with such scenarios.

We sample test sets under different distribution with test imbalance factor ρ' (i.e., long-tail ($\rho' > 1$), balanced ($\rho' = 1$), and reversed long-tail ($\rho' < 1$)), corresponding to different imbalance factor

$100, 10, 1, 0.1, 0.01$. The following comparisons in Tab.D2 show the necessity of considering test distribution's influence, which reveals our motivation to deal with bias issues. When train and test sets follow the same distribution, Bayias is equal to CE and outperforms LA. When the distribution of train and test set is different severely (e.g., $\rho' = 100$ and 0.01), Bayias outperforms both CE and LA. Similar experiments and conclusions can be obtained from [27] as well.

Table D2: Comparisons of CE, LA and Bayias loss on CIFAR-100-LT class-imbalanced test set. Note that the imbalanced test set is sampled by discarding some data. Therefore, the instance numbers of test samples vary in different imbalance factors, and longitudinal comparisons can reflect the effectiveness of Bayias.

Train set	**CIFAR-100-LT-10**					**CIFAR-100-LT-100**				
ρ'	100	10	1	0.1	0.01	100	10	1	0.1	0.01
CE	69.73	63.67	55.70	48.48	43.13	63.54	52.74	38.32	25.00	14.03
LA	61.94	60.14	59.87	56.12	54.06	59.13	51.06	43.89	31.14	26.42
Bayias	**71.89**	**63.36**	**60.02**	**58.20**	**62.32**	**64.90**	**54.15**	**43.92**	**36.20**	**35.42**

D.5 Comparisons with the two-stage method

We propose UniMix and Bayias to improve model *calibration* by tackling the bias issues caused by imbalanced distributed train and test set. We improve the model calibration and accuracy simultaneously in an end-to-end manner. However, there are some methods to ameliorate calibration in post-hoc (e.g., temperature scaling [18]) or two-stage (e.g., label aware smoothing [68]) way. As we discussed above, such pipelines will be effective and we integrate our proposed methods with one of them for further comparison. The compared results are illustrated in Tab.D3.

Table D3: Comparisons with state-of-the-art two-stage method on CIFAR-100-LT.

Dataset		CIFAR-100-LT-10		CIFAR-100-LT-50		CIFAR-100-LT-100	
Metric (%)		Accuracy	ECE	Accuracy	ECE	Accuracy	ECE
MiSLAS	1st-stage	58.7	3.91	45.6	6.00	40.3	10.77
	2nd-stage	**63.2**	**1.73**	52.3	**2.25**	47.0	4.83
Ours	1st-stage	61.1	7.79	51.8	4.91	47.6	3.24
	2nd-stage	63.0	1.98	**52.6**	3.45	**48.3**	**1.44**

MiSLAS adopts *mixup* for the 1st-stage and label aware smoothing (LAS) for 2nd-stage. We add the LAS in additional 10 epochs following our vanilla pipeline to build a two-stage manner. Our method generally outperforms MiSLAS in the 1st-stage and can achieve on par with accuracy and lower ECE. In specific, MiSLAS mainly improves accuracy and ECE in the 2nd-stage, i.e., classifier learning, by a large margin, which indicates LAS's effectiveness. We adopt the LAS to replace Bayias compensated CE loss in the 2nd-stage and can further improve accuracy and ECE, especially in the more challenging CIFAR-100-LT-100. However, the improvement of the 2nd-stage is not significant compared to MiSLAS. Since the LAS and Bayias compensated CE loss are both modifications on standard CE loss, it is hard to implement both of them simultaneously. We suggest that the LAS plays similar roles as Bayias compensated CE loss does, which explains the limited performance improvement.

E Additional discussion

What is the difference between UniMix and traditional over-sample approaches? We take representative over-sample approach SMOTE [11] for comparisons. SMOTE is a classic over-sample method for imbalanced data learning, which constructs synthetic data via interpolation among a sample's neighbors in the same class. However, the authors in [58] illustrate that such interpolation without labels is negative for classification calibration. In contrast, *mixup* manners take elaborate-designed label mix strategies. The success of label smoothing trick [25] and *mixup*s imply that the fusion of label may play an important role in classification accuracy and calibration. Different from other methods, our UniMix pays more attention to the tail when mixing the label and thus achieves remarkable performance gain.

Why choose $\boldsymbol{\xi^*_{i,j} \sim \mathscr{U}(\pi_{y_i}, \pi_{y_j}, \alpha, \alpha)}$? As we have discussed the motivation of UniMix Factor before, it is intuitive to set $\xi_{i,j} = \pi_{y_j}/(\pi_{y_i} + \pi_{y_j})$. Although satisfactory performance gain is obtained on CIFAR-LT in this manner, we fail on large-scaled dataset like ImageNet-LT. We suspect that the $\xi^*_{i,j}$ will be close to 0 or 1 when the datasets become extremely imbalanced and hence the effect of our mixing manner disappears. To improve the robustness and generalization, we transform the origin $Beta(\alpha, \alpha)$ distribution ($\alpha \leq 1$) to maximize the probability of $\xi_{i,j} = \pi_{y_j}/(\pi_{y_i} + \pi_{y_j})$ and its vicinity. We set $\alpha = 0.5$ on CIFAR-LT and keep the same α as *mixup* and Remix on ImageNet-LT and iNaturalist 2018.

Why does Bayias work? According to Bayesian theory, the estimated *likelihood* is positive correlation to *posterior* both in balance-distributed train and test set, with the same constant coefficient of *prior* and evidence factor. However, though evidence factor is regraded as a constant in LT scenarios, the LT dataset suffers seriously skewed *prior* of each category in the train set, which is extremely different from the balanced test set. Hence, the model based on maximizing *posterior* probability needs the to compensate the different priors firstly. From the perspective of optimization, Deep Neural Network (DNN) is a non-convex model, minuscule bias will eventually cause disturbances in the optimization direction, resulting in parameters converging to another local optimal. In addition, $\log(\pi_i) + \log(C)$ is consistent with traditional balanced datasets (i.e., $\pi_i \equiv 1/C$), which turns to be a special case (i.e., $\log(1/C) + \log(C) \equiv 0$) of Bayias. Furthermore, our Bayias can compensate any discrepancy between train set and test set via directly setting the Bayias as $\log(\pi_{y_i}) - \log(\pi'_{y_i})$, where π_{y_i} is the label prior in train set and π'_{y_i} is the label prior in test set.