Why Not Hyperparameter-Friendly Optimisation? A Monotonic Adaptive Norm Rescaling Approach For Long-Tailed Recognition

Supplementary Material

The supplementary material provides additional derivations, discussions, and results to support the main paper. It is organized as follows: Sec. A presents a detailed derivation for Eq. (10) and Eq. (11) from Sec. 3.2; Sec. B introduces the specific datasets and evaluation metrics used in this paper; Sec. C describes the detailed experimental setup corresponding to Sec. 4.2.

A. Derivations of Class-Conditional Distributions

According to Eq. (7), the relationship between the model's logits and the class-conditional distribution $\mathcal{D}_{\mathrm{x}}^{k}(\mathbf{x}_i)$ is defined as:

$$e^{w_k^\top \mathbf{x}_i + b_k} = Z_k \mathcal{D}_{\mathrm{x}}^{k}(\mathbf{x}_i), \tag{16}$$

where Z_k is the partition function (normalization constant). Taking the natural logarithm of both sides of Eq. (16) gives:

$$w_k^\top \mathbf{x}_i + b_k = \log \mathcal{D}_{\mathrm{x}}^{k}(\mathbf{x}_i) + \log Z_k. \tag{17}$$

Next, we multiply by a factor α where $0 < \alpha < 1$:

$$\alpha w_k^\top \mathbf{x}_i + \alpha b_k = \alpha \log \mathcal{D}_{\mathrm{x}}^{k}(\mathbf{x}_i) + \alpha \log Z_k. \tag{18}$$

We then add $(1-\alpha)b_k$ to both sides of Eq. (18). Rearranging the terms and simplifying the left-hand side yields:

$$\begin{aligned} \alpha w_k^\top \mathbf{x}_i + b_k &= (\alpha w_k^\top \mathbf{x}_i + \alpha b_k) + (1-\alpha) b_k \\ &= \alpha \log \mathcal{D}_{\mathrm{x}}^{k}(\mathbf{x}_i) + \alpha \log Z_k \\ &\quad + (1-\alpha) b_k. \end{aligned} \tag{19}$$

Now, we exponentiate both sides of Eq. (19):

$$\begin{aligned} e^{\alpha w_k^\top \mathbf{x}_i + b_k} &= \exp(\alpha \log \mathcal{D}_{\mathrm{x}}^{k}(\mathbf{x}_i) + \alpha \log Z_k \\ &\quad + (1-\alpha) b_k) \\ &= \exp(\alpha \log \mathcal{D}_{\mathrm{x}}^{k}(\mathbf{x}_i)) \cdot \exp(\alpha \log Z_k) \\ &\quad \cdot \exp((1-\alpha) b_k) \\ &= [\mathcal{D}_{\mathrm{x}}^{k}(\mathbf{x}_i)]^{\alpha} (Z_k)^{\alpha} e^{(1-\alpha) b_k}. \end{aligned} \tag{20}$$

We define our new, unnormalized distribution, $\mathcal{D}'^{k}_{\mathrm{x}}(\mathbf{x}_i)$, as being proportional to the new exponentiated logits $e^{\alpha w_k^\top \mathbf{x}_i + b_k}$. From Eq. (20), we have:

$$\begin{aligned} \mathcal{D}'^{k}_{\mathrm{x}}(\mathbf{x}_i) &\propto e^{\alpha w_k^\top \mathbf{x}_i + b_k} \\ &\propto [\mathcal{D}_{\mathrm{x}}^{k}(\mathbf{x}_i)]^{\alpha} (Z_k)^{\alpha} e^{(1-\alpha) b_k}. \end{aligned} \tag{21}$$

This unnormalized relationship corresponds to Eq. (10) in the main paper.

Since both $(Z_k)^{\alpha}$ and $e^{(1-\alpha)b_k}$ are independent of $\mathbf{x}_i$, they can be omitted as constant factors, yielding the proportional relationship:

$$\mathcal{D}'^{k}_{\mathrm{x}}(\mathbf{x}_i) \propto [\mathcal{D}_{\mathrm{x}}^{k}(\mathbf{x}_i)]^{\alpha}. \tag{22}$$

Finally, we normalize the distribution. We introduce a normalization constant C' such that the distribution integrates to one:

$$\mathcal{D}'^{k}_{\mathrm{x}}(\mathbf{x}_i) = C' \cdot [\mathcal{D}_{\mathrm{x}}^{k}(\mathbf{x}_i)]^{\alpha}. \tag{23}$$

We solve for C' by integrating over the entire domain (using $\mathbf{x}$ as the integration variable):

$$\begin{aligned} \int \mathcal{D}'^{k}_{\mathrm{x}}(\mathbf{x})\, d\mathbf{x} &= 1 \\ \int C' \cdot [\mathcal{D}_{\mathrm{x}}^{k}(\mathbf{x})]^{\alpha}\, d\mathbf{x} &= 1 \\ C' \int [\mathcal{D}_{\mathrm{x}}^{k}(\mathbf{x})]^{\alpha}\, d\mathbf{x} &= 1. \end{aligned} \tag{24}$$

This gives us the constant:

$$C' = \frac{1}{\int [\mathcal{D}_{\mathrm{x}}^{k}(\mathbf{x})]^{\alpha}\, d\mathbf{x}} \tag{25}$$

Substituting C' back into our expression for $\mathcal{D}'^{k}_{\mathrm{x}}(\mathbf{x}_i)$ yields the final, normalized form of Eq. (11) in the main paper:

$$\mathcal{D}'^{k}_{\mathrm{x}}(\mathbf{x}_i) = \frac{[\mathcal{D}_{\mathrm{x}}^{k}(\mathbf{x}_i)]^{\alpha}}{\int [\mathcal{D}_{\mathrm{x}}^{k}(\mathbf{x})]^{\alpha}\, d\mathbf{x}} \tag{26}$$

As this derivation shows, all constant factors from Eq. (20), namely $(Z_k)^{\alpha}$ and $e^{(1-\alpha)b_k}$, are constants with respect to $\mathbf{x}_i$ and are absorbed into the normalization constant C', ultimately canceling out during normalization. The integration variable $\mathbf{x}$ in the denominator is a bound variable, distinct from the specific instance $\mathbf{x}_i$.

B. Datasets and Evaluation Metrics

In this paper, we use four datasets: CIFAR10-LT [21], CIFAR100-LT [21], ImageNet-LT [29], iNaturalist2018 [38] to perform the experiments. CIFAR10-LT and CIFAR100-LT are the long-tailed versions of CIFAR10 and CIFAR100. They are generated by downsampling the per-class training samples using the exponential decay function. The degree of imbalance is controlled by the imbalance factor (IF) mentioned in Sec. 3.1. We set IF $\in \{10, 50, 100\}$ in this paper, while their validation sets are still balanced. ImageNet-LT

was introduced in [29] by artificially truncating the original ImageNet dataset [11]. It has 1,000 classes, and the number of per-class training data ranges from 5 to 1280, with an IF of 256. iNaturalist2018 is another real-world dataset for the identification of species of animals and plants, containing 8,142 classes with an extremely large IF of 500. Regarding the evaluation metric, we use the overall accuracy in all datasets. In addition, we also classify the classes into three subsets of "Many" ($n_k > 100$), "Medium" ($20 \leq n_k \leq 100$), and "Few" ($n_k < 20$) and report their accuracy on ImageNet-LT and iNaturalist2018.

C. Detailed Setup in Sec. 4.2

We evaluate SAMN on the benchmark datasets CIFAR10-LT and CIFAR100-LT, as well as the large-scale datasets ImageNet-LT and iNaturalist2018. All experiments utilize a stochastic gradient descent (SGD) optimizer with 0.9 momentum and a cosine learning rate scheduler [30]. For the CIFAR10-LT and CIFAR100-LT datasets, we adopt the ResNet32 [16] backbone and apply SAMN to models pretrained with CE, GLMC [12], and SLAS [49]. The first stage training for CE and GLMC runs for 200 epochs with a batch size of 64, a weight decay of 5e-3, and an initial learning rate of 0.01. The hyperparameters of GLMC and SLAS follow their original implementations. In the second stage (applying SAMN), we retrain only the classifiers for 20 epochs (batch size = 64), using an initial learning rate of 5e-4 for CE, 5e-5 for GLMC, and 1e-5 (CIFAR10-LT) or 5e-4 (CIFAR100-LT) for SLAS. For the large-scale datasets, we use a ResNeXt50 [42] backbone, employing GLMC for first-stage training and SAMN for second-stage enhancement. Following [12], on ImageNet-LT, the first stage is trained for 135 epochs (batch size = 128, weight decay = 2e-4, learning rate = 0.1), and the second stage is finetuned for 20 epochs (batch size = 512, learning rate = 1e-5). On iNaturalist2018, the first stage is trained for 120 epochs (batch size = 128, weight decay = 5e-3, learning rate = 0.1), and the second stage is finetuned for 20 epochs (batch size = 512, learning rate = 5e-5). Notably, weight decay was not applied during the second stage in any experiment.