CUE: Concept-Aware Multi-Label Expansion to Mitigate Concept Confusion in Long-Tailed Learning

Supplementary Material

A. Overall Training Procedure

We summarize the complete optimization process of our framework in Algorithm 1. It outlines how the baseline logit-adjusted loss and the two binary logit-adjusted terms are jointly optimized within each minibatch.

Algorithm 1: CUE:Concept-Aware Multi-Label Expansion

Input: Training set $\mathcal{D} = \{(x_i, y_i)\}$, class prior π_c, CLIP model θ^{zs}, LLM prior $\mathcal{N}^{\text{llm}}$

Output: Optimized model parameters Θ

for *each minibatch* $(\mathbf{x}_i, y_i)$ **do**

- Compute logits by model: $\theta(\mathbf{x}_i) = f(\mathbf{x}_i; \Theta)$
- Compute $\mathcal{L}^{\text{LA}}$ by Eq. (5).
- **VLM prior:** $\tilde{\mathbf{t}}_i^{\text{zs}} = \mathbf{1}_{\{c \in \{y_i\} \cup \text{Top-}k(\theta^{\text{zs}}(\mathbf{x}_i))\}}$
- **LLM prior:** $\tilde{\mathbf{t}}_i^{\text{llm}} = \mathbf{1}_{\{c \in \{y_i\} \cup \mathcal{N}^{\text{llm}}(y_i)\}}$
- Compute $\mathcal{L}_{\text{VLM}}^{\text{BLA}}$ and $\mathcal{L}_{\text{LLM}}^{\text{BLA}}$ by Eq. (6).
- **Joint optimization:** Combine all terms:

$$\mathcal{L} = \mathcal{L}^{\text{LA}} + \lambda_{\text{zs}} \mathcal{L}_{\text{VLM}}^{\text{BLA}} + \lambda_{\text{llm}} \mathcal{L}_{\text{LLM}}^{\text{BLA}}$$

- Update $\Theta \leftarrow \Theta - \eta \nabla_\Theta \mathcal{L}$

end

B. Proof of Logit Adjustment Formulation

B.1. Proof of LA

Consider a long-tailed training distribution

$$P_{\text{train}}(x, y), \qquad P_{\text{train}}(y = c) = \pi_c,$$

where π_c is the empirical class prior. Under the training prior, Bayes' rule gives

$$P_{\text{train}}(y = c \mid x) \propto P(x \mid y = c)\, \pi_c,$$

while under the desired balanced prior

$$P_{\text{bal}}(y = c) = \frac{1}{C}, \qquad P_{\text{bal}}(y = c \mid x) \propto P(x \mid y = c)\, \frac{1}{C}.$$

Dividing the two posteriors yields

$$P_{\text{bal}}(y = c \mid x) \propto \frac{P_{\text{train}}(y = c \mid x)}{\pi_c}.$$

Assume the model outputs logits $\theta_c(x)$ such that

$$P_{\text{train}}(y = c \mid x) = \frac{\exp(\theta_c(x))}{\sum_j \exp(\theta_j(x))} = \text{softmax}_c(\theta(x)),$$

we obtain the balanced-posterior form

$$\begin{aligned} P_{\text{bal}}(y = c \mid x) &= \frac{\exp(\theta_c(x) - \log \pi_c)}{\sum_k \exp(\theta_k(x) - \log \pi_k)} \\ &= \text{softmax}_c\big(\theta_c(x) - \log \pi_c\big). \end{aligned}$$

Thus we define logit-adjusted logits

$$\theta'_c(x) = \theta_c(x) + \tau \log \pi_c,$$

where τ is a temperature coefficient: $\tau < 0$ is used during training, while $\tau > 0$ is used as a post-hoc adjustment during testing. Applying cross-entropy on $\theta'(x)$ yields the LA loss in Eq. (5).

B.2. Proof of BLA

For the multi-label setting, each class c is associated with a binary variable $z_c \in \{0, 1\}$, with

$$P_{\text{train}}(z_c = 1) = P_{\text{train}}(y = c) = \pi_c,$$

and the balanced prior

$$P_{\text{bal}}(z_c = 1) = \frac{1}{C}.$$

Analogous to the multiclass case, Bayes' rule gives the posterior relation

$$P_{\text{bal}}(z_c = 1 \mid x).$$

Assuming the model outputs logits $\theta_c(x)$ such that

$$P_{\text{train}}(z_c = 1 \mid x) = \sigma(\theta_c(x)),$$

we obtain the *exact* balanced posterior logit:

$$\begin{aligned} \theta_c^{\text{bal}}(x) &:= \log \frac{P_{\text{bal}}(z_c = 1 \mid x)}{P_{\text{bal}}(z_c = 0 \mid x)} \\ &= \theta_c(x) + \log \frac{\frac{1}{C}}{\frac{C-1}{C}} - \log \frac{\pi_c}{1 - \pi_c} \\ &= \theta_c(x) + \log \frac{1}{C - 1} - \log \frac{\pi_c}{1 - \pi_c}. \end{aligned}$$

In long-tailed learning settings, C is typically fixed. Under this regime, the balanced logit reduces to the following long-tailed approximation:

$$\theta_c^{\text{bal}}(x) \approx \theta_c(x) - \log \pi_c + \log(1 - \pi_c) + \text{const},$$

where the constant term is class-independent and thus does not affect class-wise relative preference. At this point, the balanced logit for the binary case can already be used directly for training. However, noting that the term $-\log \pi_c$ is the dominant class-dependent factor under long-tailed priors, and in order to maintain consistency with the multiclass LA formulation (thus simplifying implementation), we adopt the same prior-adjustment form as LA:

$$\tilde{\theta}_c(x) = \theta_c(x) + \tau_b \log \pi_c,$$

where τ_b is a temperature coefficient (typically $\tau_b < 0$ during training). Applying binary cross-entropy to $\tilde{\theta}_c(x)$ with multi-label targets $\tilde{t}_{i,c}$ yields the BLA loss in Eq. (6).

C. Additional Results on Different Top-k Settings

In addition to the main results reported in the paper, we further evaluate the effect of different Top-k selections used in constructing VLM-based instance cues. Experiments are conducted on both ImageNet-LT and Places-LT, with $k \in \{10, 20, 50\}$. The results in Table 10 show that CUE remains consistently stable across different k values on both benchmarks.

Table 10. Results under different Top-k settings. We report All / Many / Medium / Few performance.

Top-k	All	Many	Med.	Few
ImageNet-LT				
Top-10	**77.5**	**80.3**	76.4	73.4
Top-20	**77.5**	80.2	76.3	**73.6**
Top-50	**77.5**	80.2	**76.5**	73.2
Places-LT				
Top-10	**51.7**	50.6	**52.2**	52.4
Top-20	51.6	50.6	52.1	**52.5**
Top-50	51.6	**50.9**	**52.2**	51.4

D. Extension to Additional Benchmarks

To further assess the generality of CUE beyond the benchmarks reported in the main paper, we extend our evaluation to Places-LT and ImageNet-LT. We conduct experiments using both PEFT-based adaptation methods and from-scratch training baselines, As shown in the following sections.

D.1 Results on Places-LT

As shown in Table 11, CUE consistently brings improvements across different adaptation paradigms, especially on the Few-shot categories. For the from-scratch setting, following prior work[8, 22], we adopt a ResNet-152 backbone initialized from an ImageNet-pretrained checkpoint and train the model for 30 epochs using a cosine learning rate schedule from 0.1 to 0.

Table 11. Results on Places-LT using PEFT methods and from-scratch baselines. We report All / Many / Med. / Few performance.

Method	All	Many	Med.	Few
PEFT				
LoRA [16]	51.0	**50.5**	**51.9**	50.0
+ CUE	**51.1**	50.4	51.7	**50.8**
VPT-shallow [19]	49.2	**49.5**	**51.0**	44.6
+ CUE	**49.2**	48.5	50.8	**46.9**
VPT-deep [19]	50.9	**50.8**	**51.8**	49.0
+ CUE	**50.9**	50.7	51.3	**50.3**
Adapter [15]	51.3	**51.1**	**52.0**	50.0
+ CUE	**51.3**	50.5	51.8	**51.6**
From Scratch				
LA [30]	37.0	42.0	36.9	28.7
+ CUE	**37.3**	**42.0**	**37.0**	**30.0**
ResLT [8]	35.6	**34.6**	39.3	30.0
+ CUE	**36.6**	32.4	**40.8**	**35.0**

D.2 Results on ImageNet-LT

We further evaluate CUE on ImageNet-LT to examine its effectiveness across different settings. As shown in Table 12, CUE consistently improves performance for both PEFT-based methods and from-scratch baselines, with clear gains on medium- and few-shot categories. For the from-scratch experiments, we follow the standard ImageNet-LT protocol and train a ResNet-50 backbone for 90 epochs using SGD. The learning rate is initialized at 0.1 and annealed to 0 using a cosine decay schedule.

Table 12. Results on ImageNet-LT using PEFT methods and from-scratch baselines. We report All / Many / Med. / Few performance.

Method	All	Many	Med.	Few
PEFT				
LoRA [16]	75.8	78.7	75.2	70.0
+ CUE	**76.3**	**78.9**	**75.7**	**70.9**
VPT-shallow [19]	74.2	**77.9**	73.9	65.3
+ CUE	**74.7**	77.7	**74.4**	**67.1**
VPT-deep [19]	76.1	78.6	75.4	71.2
+ CUE	**76.3**	**78.8**	**75.7**	**71.8**
Adapter [15]	76.9	**80.2**	75.9	71.2
+ CUE	**77.2**	80.0	**76.2**	**72.7**
From Scratch				
LA [30]	49.1	**59.8**	46.6	28.9
+ CUE	**49.4**	59.8	**46.8**	**30.1**
ResLT [8]	46.9	**52.2**	47.3	31.4
+ CUE	**47.6**	52.2	**48.1**	**33.7**

E. Multi-Label Mappings

E.1. LLM-based Class-level Mappings

We provide examples of the expanded multi-label targets generated by the LLM prior. All semantic neighbors are obtained using the **Qwen-plus** model, queried with the prompt described in Section F. The mappings below illustrate how concept-level similarity leads to meaningful auxiliary labels that enrich the supervision signal.

Table 13. Examples of LLM-based cues produced by **Qwen-plus**.

Class	Expanded Cues
apple	[orange, pear, mushroom, sweet_pepper]
aquarium_fish	[dolphin, shark, trout, ray, seal]
willow_tree	[maple_tree, oak_tree, pine_tree, palm_tree, forest]
woman	[man, girl, boy, baby]
worm	[snail, snake, spider, beetle, caterpillar]

E.2. VLM-based Instance-level Mappings.

We also visualize the instance-level expansions generated from the VLM prior. Each figure shows one input image along with the top-5 VLM cues used as additional labels.

Figure 6. Example VLM-based cues for the query images.

F. Prompts Used for Querying the LLM

We use the following prompt template to obtain cues from the LLM. The prompt is shown below. The red-highlighted placeholders will be dynamically replaced with real values.

```
You are given a complete list of all_classes:
{all_classes}

Task: Find similar classes for {batch},
using items from the complete list
above.

Requirements:

- Return a valid JSON object only.

- Use the following format: { "class1":
["similar1", "similar2"], "class2":
["similar1", "similar2"] }

- Only choose from the provided complete
list.

- Output JSON only, no additional
explanation.
```

G. Token Usage Statistics

We report the total token usage incurred during the generation of LLM-based cues. For each dataset, class names are grouped into batches and queried using the prompt described in Section F. Table 14 summarizes the approximate number of tokens used for each dataset.

Table 14. Approximate token usage for LLM-based neighbor generation.

Dataset	#Classes	Total Tokens
CIFAR100-LT	100	∼8.4K
Places-LT	365	∼85.6K
ImageNet-LT	1000	∼539.6K
iNaturalist	8142	∼57.3M

All token costs remain inexpensive under modern LLM API pricing, and the process needs to be performed only once per dataset.