Distilling Balanced Knowledge from a Biased Teacher

Supplementary Material

A. Detailed decomposition of KL divergence

In a long-tailed dataset, the KD loss based on KL divergence is formulated as:

$$\begin{aligned}\mathrm{KD} &= \mathrm{KL}(\mathbf{p}^T \| \mathbf{p}^S) \\ &= \sum_{i=1}^{C} p_i^T \log\left(\frac{p_i^T}{p_i^S}\right) \\ &= \sum_{\mathcal{G}} \sum_{i \in \mathcal{G}} p_i^T \log\left(\frac{p_i^T}{p_i^S}\right).\end{aligned} \tag{16}$$

Using the relation $p_i = p_{\mathcal{G}} \cdot \tilde{p}_{\mathcal{G}_i}$, we can rewrite Eq. (16) as:

$$\begin{aligned}\mathrm{KD} &= \sum_{\mathcal{G}} \sum_{i \in \mathcal{G}} p_i^T \log\left(\frac{p_{\mathcal{G}}^T \tilde{p}_{\mathcal{G}_i}^T}{p_{\mathcal{G}}^S \tilde{p}_{\mathcal{G}_i}^S}\right) \\ &= \sum_{\mathcal{G}} \sum_{i \in \mathcal{G}} p_i^T \left(\log\frac{p_{\mathcal{G}}^T}{p_{\mathcal{G}}^S} + \log\frac{\tilde{p}_{\mathcal{G}_i}^T}{\tilde{p}_{\mathcal{G}_i}^S}\right) \\ &= \sum_{\mathcal{G}} \sum_{i \in \mathcal{G}} p_i^T \log\left(\frac{p_{\mathcal{G}}^T}{p_{\mathcal{G}}^S}\right) + \sum_{\mathcal{G}} \sum_{i \in \mathcal{G}} p_i^T \log\left(\frac{\tilde{p}_{\mathcal{G}_i}^T}{\tilde{p}_{\mathcal{G}_i}^S}\right).\end{aligned} \tag{17}$$

Since $p_{\mathcal{G}}^T$ and $p_{\mathcal{G}}^S$ do not depend on the class index i and $\sum_{i \in \mathcal{G}} p_i^T = p_{\mathcal{G}}^T$, the first term in Eq. (17) simplifies to:

$$\begin{aligned}\sum_{\mathcal{G}} \sum_{i \in \mathcal{G}} p_i^T \log\left(\frac{p_{\mathcal{G}}^T}{p_{\mathcal{G}}^S}\right) &= \sum_{\mathcal{G}} p_{\mathcal{G}}^T \log\left(\frac{p_{\mathcal{G}}^T}{p_{\mathcal{G}}^S}\right) \\ &= \mathrm{KL}\left(\mathbf{p}_{\mathcal{G}}^T || \mathbf{p}_{\mathcal{G}}^S\right).\end{aligned} \tag{18}$$

Likewise, the second term can be written as:

$$\begin{aligned}\sum_{\mathcal{G}} \sum_{i \in \mathcal{G}} p_i^T \log\left(\frac{\tilde{p}_{\mathcal{G}_i}^T}{\tilde{p}_{\mathcal{G}_i}^S}\right) &= \sum_{\mathcal{G}} p_{\mathcal{G}}^T \sum_{i \in \mathcal{G}} \tilde{p}_{\mathcal{G}_i}^T \log\left(\frac{\tilde{p}_{\mathcal{G}_i}^T}{\tilde{p}_{\mathcal{G}_i}^S}\right) \\ &= \sum_{\mathcal{G}} p_{\mathcal{G}}^T \cdot \mathrm{KL}\left(\tilde{\mathbf{p}}_{\mathcal{G}}^T || \tilde{\mathbf{p}}_{\mathcal{G}}^S\right).\end{aligned} \tag{19}$$

Finally, the KD loss in Eq. (16) can be decomposed into cross-group and within-group KL terms:

$$\begin{aligned}\mathrm{KD} &= \mathrm{KL}\left(\mathbf{p}^T \| \mathbf{p}^S\right) \\ &= \mathrm{KL}\left(\mathbf{p}_{\mathcal{G}}^T \| \mathbf{p}_{\mathcal{G}}^S\right) + \sum_{\mathcal{G}} p_{\mathcal{G}}^T \cdot \mathrm{KL}\left(\tilde{\mathbf{p}}_{\mathcal{G}}^T \| \tilde{\mathbf{p}}_{\mathcal{G}}^S\right) \\ &= \mathrm{KL}\left(\mathbf{p}_{\mathcal{G}}^T \| \mathbf{p}_{\mathcal{G}}^S\right) + p_{\mathcal{H}}^T \cdot \mathrm{KL}\left(\tilde{\mathbf{p}}_{\mathcal{H}}^T \| \tilde{\mathbf{p}}_{\mathcal{H}}^S\right) \\ &\quad + p_{\mathcal{M}}^T \cdot \mathrm{KL}\left(\tilde{\mathbf{p}}_{\mathcal{M}}^T \| \tilde{\mathbf{p}}_{\mathcal{M}}^S\right) + p_{\mathcal{T}}^T \cdot \mathrm{KL}\left(\tilde{\mathbf{p}}_{\mathcal{T}}^T \| \tilde{\mathbf{p}}_{\mathcal{T}}^S\right).\end{aligned} \tag{20}$$

Algorithm 1 Process of knowledge distillation in long-tailed dataset.

Input: Long-tailed dataset $\mathcal{D}$,
Head, Medium, Tail groups $\mathcal{H}, \mathcal{M}, \mathcal{T}$,
Temperature scaling τ,
Softmax function $\sigma(\cdot)$,
Teacher, Student networks f_T, f_S,
Hyperparameters for cross-group loss α,
and within-group loss β.

Output: Trained student network f_S.

for each image-label pair $\{\mathbf{x}, y\} \in \mathcal{D}$ **do**
  $f_T(\mathbf{x}) \rightarrow \mathbf{z}^T, f_S(\mathbf{x}) \rightarrow \mathbf{z}^S$
  $\sigma\left(\mathbf{z}^T/\tau\right) \rightarrow \mathbf{p}^T, \sigma\left(\mathbf{z}^S/\tau\right) \rightarrow \mathbf{p}^S$
  $\mathcal{L}_{\text{CE}} = \text{CrossEntropy}(\mathbf{z}^S, y)$

  // Sec. 3.2: Cross-Group Distillation
  $\left[\sum_{i \in \mathcal{H}} p_i^T, \sum_{i \in \mathcal{M}} p_i^T, \sum_{i \in \mathcal{T}} p_i^T\right] \rightarrow \mathbf{p}_{\mathcal{G}}^T$
  rebalance from Eq. (10) and Eq. (11) $\rightarrow \hat{\mathbf{p}}_{\mathcal{G}}^T$
  $\left[\sum_{i \in \mathcal{H}} p_i^S, \sum_{i \in \mathcal{M}} p_i^S, \sum_{i \in \mathcal{T}} p_i^S\right] \rightarrow \mathbf{p}_{\mathcal{G}}^S$
  $\mathcal{L}_{\text{Cross}} = \mathrm{KL}(\hat{\mathbf{p}}_{\mathcal{G}}^T \| \mathbf{p}_{\mathcal{G}}^S)$

  // Sec. 3.3: Within-Group Distillation
  for each class group $\mathcal{G} \in \{\mathcal{H}, \mathcal{M}, \mathcal{T}\}$ **do**
    $\sigma\left(\mathbf{z}^T[\mathcal{G}]/\tau\right) \rightarrow \tilde{\mathbf{p}}_{\mathcal{G}}^T, \sigma\left(\mathbf{z}^S[\mathcal{G}]/\tau\right) \rightarrow \tilde{\mathbf{p}}_{\mathcal{G}}^S$
    $\mathcal{L}_{\mathcal{G}} = \mathrm{KL}(\tilde{\mathbf{p}}_{\mathcal{G}}^T \| \tilde{\mathbf{p}}_{\mathcal{G}}^S)$
  end for
  $\mathcal{L}_{\text{Within}} = \mathcal{L}_{\mathcal{H}} + \mathcal{L}_{\mathcal{M}} + \mathcal{L}_{\mathcal{T}}$

  // Total Long-Tailed Knowledge Distillation
  $\mathcal{L}_{\text{Total}} = \mathcal{L}_{\text{CE}} + \alpha \mathcal{L}_{\text{Cross}} + \beta \mathcal{L}_{\text{Within}}$
  Update f_S by minimizing $\mathcal{L}_{\text{Total}}$
end for

B. Algorithm

Algorithm 1 summarizes the LTKD training process. The total loss $\mathcal{L}_{\text{Total}}$ comprises three components.

First, $\mathcal{L}_{\text{CE}}$ is the standard cross-entropy loss between the student's logits and the ground-truth label. Second, $\mathcal{L}_{\text{Cross}}$, detailed in Sec. 3.2, is the rebalanced cross-group distillation loss. It forces the student to mimic a rebalanced teacher group distribution ($\hat{\mathbf{p}}_{\mathcal{G}}^T$), which is corrected using Eq. (10) and Eq. (11). Third, $\mathcal{L}_{\text{Within}}$, detailed in Sec. 3.3, is the sum of KL losses computed independently within each class group (head $\mathcal{H}$, medium $\mathcal{M}$, and tail $\mathcal{T}$).

These losses are weighted by hyperparameters α and β and summed. The student network f_S is then updated by minimizing this $\mathcal{L}_{\text{Total}}$.

C. Experimental setting

C.1. Balanced dataset

The balanced CIFAR-100 [22] consists of 100 categories with 50,000 training and 10,000 test images, each of size 32×32 pixels. TinyImageNet [23] is a subset of ImageNet comprising 200 categories, with each class containing 500 training and 50 test images, resized to 64×64 pixels. ImageNet [31] consists of 1.28 million training images and 50,000 test images across 1,000 categories, each resized to 224×224 pixels. As mentioned in Sec. 4.1, we constructed CIFAR-100-LT, TinyImageNet-LT, and ImageNet-LT by introducing class imbalance to these balanced datasets.

C.2. Implementation details

We conduct experiments using widely used convolutional neural network architectures, including ResNet [14], VGG [33], WRN [46], ShuffleNet [48], and MobileNet [32]. We consider both homogeneous settings, where the teacher and student share the same backbone architecture, and heterogeneous settings, where they use different backbones.

We follow standard practices using an SGD optimizer with a momentum of 0.9. The training schedule and hyperparameters vary by dataset. For CIFAR-100-LT and TinyImageNet-LT, we train for 240 epochs with a batch size of 64 and a weight decay of 5e-4. The initial learning rate is set to 0.05 (or 0.01 for ShuffleNetV1 and MobileNetV2 students), and it is reduced by 0.1 at [150, 180, 210] epochs. For ImageNet-LT, we train for 100 epochs with a batch size of 256, a weight decay of 1e-4, and an initial learning rate of 0.1, which is reduced by 0.1 at [30, 60, 90] epochs.

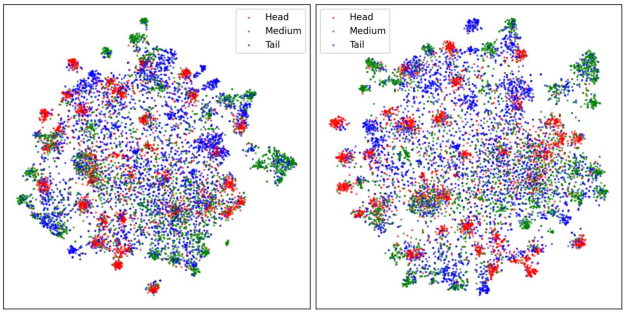
D. Additional results

D.1. Long-tailed protocols

We integrate a representative long-tailed learning technique, Logit Adjustment (LA) [27], into the training pipeline. Tab. 7 shows that LTKD consistently achieves additional performance gains on top of LA-enhanced baselines. This indicates that LTKD is not tied to vanilla protocols, but instead serves as a general and complementary distillation framework that can be effectively combined with state-of-the-art LT recognition approaches.

Table 7. Overall accuracy on CIFAR-100-LT.

T-S Pairs	R32×4−R8×4		VGG13−VGG8		WRN402−SV1		R50−MV2	
γ	10	100	10	100	10	100	10	100
LA	62.36	48.36	59.04	45.46	55.45	39.23	46.27	31.17
KD	62.85	44.50	61.41	43.56	62.37	42.56	53.55	36.17
LA + KD	63.29	44.94	61.84	44.42	62.69	42.76	54.34	36.01
LTKD	66.76	51.08	63.04	47.66	65.42	48.60	57.79	42.45
LA + LTKD	**67.01**	**51.69**	**63.67**	**48.92**	**65.75**	**49.63**	**58.45**	**43.20**



(a) KD (b) LTKD

Figure 6. tSNE visualization of student feature representations. (a) Student trained with standard KD. (b) Student trained with proposed LTKD. Points are colored by class group: head (red), medium (green), and tail (blue). Compared to KD, LTKD yields more compact and well-separated clusters for tail classes.

D.2. Visualization

To further investigate the quality of the learned representations, we visualize the feature embeddings of the student model (ShuffleNetV1) trained from the teacher model (ResNet32×4) using both KD and proposed LTKD. Fig. 6 shows the 2D projection of features from the CIFAR-100-LT validation set, where each point is colored according to its class group: head (red), medium (green), or tail (blue). In the KD case (Fig. 6a), the embeddings of tail-class samples appear scattered and poorly separated, indicating suboptimal feature learning for underrepresented classes. In contrast, LTKD (Fig. 6b) produces tighter and more distinct clusters, especially for the tail group. This result supports our claim that rebalanced cross-group and reweighted within-group knowledge transfer enables the student to learn richer and more discriminative representations, particularly for tail-class groups.

D.3. CIFAR-100-LT

Tab. 8 and Tab. 9 present extensive experimental results for CIFAR-100-LT, covering various homogeneous and heterogeneous teacher-student architecture pairs. Consistent with the findings in Sec. 4.3, LTKD demonstrates substantially superior performance over other methods in both overall and tail-class accuracy.

D.4. TinyImageNet-LT

Tab. 10 and Tab. 11 summarize the results on TinyImageNet-LT, which presents a more challenging benchmark than CIFAR-100-LT due to its greater class diversity. Even on this more complex dataset, the results strongly reinforce the conclusions from Sec. 4.3, consistently outperforming all competing methods in both overall and tail-class accuracy.

Table 8. Accuracy (%) on both the tail group classes ($\mathcal{T}$) and the overall classes (All) in the CIFAR-100-LT test sets when using teacher and student models with homogeneous architectures. The best result is highlighted in **bold**, and the second-best result is indicated with underline. Δ denotes the performance gap between the best and the second-best results.

T-S Pairs	ResNet32×4 − ResNet8×4						VGG13 − VGG8					
γ	10		20		100		10		20		100	
Group	$\mathcal{T}$	All	$\mathcal{T}$	All	$\mathcal{T}$	All	$\mathcal{T}$	All	$\mathcal{T}$	All	$\mathcal{T}$	All
Teacher	50.72	64.95	39.19	58.82	15.28	45.35	45.67	60.77	36.43	55.10	14.01	43.11
Student	47.32	60.59	36.99	55.44	13.38	42.48	43.67	57.43	33.89	52.29	13.13	40.70
KD [16]	48.76	62.85	36.81	57.41	11.43	44.50	47.84	61.41	37.29	56.13	14.02	43.56
FitNet [30]	48.02	60.86	36.72	55.60	13.01	42.82	44.47	58.30	35.90	54.08	14.42	42.48
DKD [52]	49.86	64.55	37.87	58.78	13.25	46.11	48.00	61.84	37.65	56.68	14.42	44.22
ReviewKD [6]	52.08	64.71	40.12	59.17	15.09	45.91	47.75	61.43	37.69	56.51	14.76	44.19
DIST [17]	50.28	63.74	38.69	58.28	13.86	45.21	45.57	60.53	34.36	54.68	12.46	42.12
CAT-KD [12]	49.83	64.74	37.67	58.73	12.83	45.33	48.53	62.01	37.95	56.78	14.22	44.33
LTKD	**58.66**	**66.76**	**49.70**	**62.54**	**27.21**	**51.08**	**53.95**	**63.04**	**45.77**	**58.86**	**23.30**	**47.66**
Δ	**+6.58**	**+2.02**	**+9.58**	**+3.37**	**+12.12**	**+4.97**	**+5.42**	**+1.03**	**+7.82**	**+2.08**	**+8.54**	**+3.33**
T-S Pairs	WRN–40–2 − WRN–40–1						ResNet110 − ResNet32					
γ	10		20		100		10		20		100	
Group	$\mathcal{T}$	All	$\mathcal{T}$	All	$\mathcal{T}$	All	$\mathcal{T}$	All	$\mathcal{T}$	All	$\mathcal{T}$	All
Teacher	49.77	63.05	39.88	58.27	14.88	44.78	47.28	61.18	36.63	55.49	13.32	42.45
Student	44.61	59.02	34.44	54.24	12.92	42.29	44.04	59.13	34.52	53.82	13.32	41.68
KD [16]	47.10	62.17	35.63	56.71	10.09	43.66	44.63	60.98	33.10	55.39	9.33	41.89
FitNet [30]	44.81	59.18	34.41	53.92	13.60	42.42	44.05	58.02	32.50	52.00	12.72	40.28
DKD [52]	47.01	62.53	36.87	57.52	11.74	44.17	46.21	61.42	34.72	55.70	11.41	42.54
ReviewKD [6]	48.71	62.70	38.44	57.78	14.69	44.71	46.67	60.87	36.41	55.56	13.48	42.70
DIST [17]	48.18	62.32	37.00	56.98	13.19	44.09	46.47	61.15	35.77	55.93	12.68	42.54
CAT-KD [12]	48.26	62.61	37.45	57.73	12.82	44.18	45.63	60.92	34.26	54.90	12.23	42.07
LTKD	**54.23**	**64.21**	**45.74**	**59.91**	**20.57**	**47.73**	**54.24**	**62.90**	**45.55**	**58.53**	**22.72**	**46.54**
Δ	**+5.52**	**+1.51**	**+7.30**	**+2.13**	**+5.88**	**+3.02**	**+7.57**	**+1.48**	**+9.14**	**+2.60**	**+9.24**	**+3.84**

Table 9. Accuracy (%) on both the tail group classes ($\mathcal{T}$) and the overall classes (All) in the CIFAR-100-LT test sets when using teacher and student models with heterogeneous architectures. The best result is highlighted in **bold**, and the second-best result is indicated with underline. Δ denotes the performance gap between the best and the second-best results.

T-S Pairs	ResNet32×4 − ShuffleNetV1						VGG13 − MobileNetV2					
γ	10		20		100		10		20		100	
Group	$\mathcal{T}$	All	$\mathcal{T}$	All	$\mathcal{T}$	All	$\mathcal{T}$	All	$\mathcal{T}$	All	$\mathcal{T}$	All
Teacher	50.72	64.95	39.19	58.82	15.28	45.35	45.67	60.77	36.43	55.10	14.01	43.11
Student	40.04	54.06	29.91	48.22	10.74	36.21	30.47	44.58	22.32	39.25	7.04	27.56
KD [16]	47.08	60.88	36.13	55.03	13.07	42.04	39.78	53.64	29.05	47.69	8.99	34.56
FitNet [30]	44.82	58.70	33.77	52.78	13.36	40.66	29.38	43.82	21.38	38.54	7.26	28.31
DKD [52]	50.23	63.32	39.00	57.81	14.98	44.76	41.81	55.84	31.48	50.27	10.94	38.22
ReviewKD [6]	50.02	63.31	38.38	57.64	15.18	44.69	40.03	53.17	29.70	48.06	11.75	36.31
DIST [17]	46.05	61.11	34.25	54.71	11.99	42.33	37.14	51.91	26.85	46.34	8.92	34.38
CAT-KD [12]	50.49	63.86	39.27	58.42	16.37	45.28	39.40	53.56	29.57	47.89	10.08	34.76
LTKD	**54.87**	**64.60**	**45.94**	**59.62**	**23.93**	**48.59**	**46.49**	**57.18**	**38.33**	**52.03**	**18.24**	**41.04**
Δ	**+4.38**	**+0.74**	**+6.67**	**+1.20**	**+7.56**	**+3.31**	**+4.68**	**+1.34**	**+6.85**	**+1.76**	**+6.49**	**+2.82**
T-S Pairs	WRN–40–2 − ShuffleNetV1						ResNet50 − MobileNetV2					
γ	10		20		100		10		20		100	
Group	$\mathcal{T}$	All	$\mathcal{T}$	All	$\mathcal{T}$	All	$\mathcal{T}$	All	$\mathcal{T}$	All	$\mathcal{T}$	All
Teacher	49.77	63.05	39.88	58.27	14.88	44.78	49.74	63.51	37.74	56.70	14.42	42.26
Student	40.04	54.06	29.91	48.22	10.74	36.21	30.47	44.58	22.32	39.25	7.04	27.56
KD [16]	49.10	62.37	38.30	56.61	12.59	42.56	39.59	53.55	30.22	48.87	10.00	36.17
FitNet [30]	45.02	58.72	34.56	53.40	13.30	41.27	29.40	42.86	21.21	38.22	7.44	28.01
DKD [52]	50.86	63.65	39.94	58.28	15.04	45.24	43.29	57.20	33.23	52.20	12.45	39.21
ReviewKD [6]	51.24	63.90	40.44	58.63	15.81	45.40	33.68	47.75	24.80	42.08	9.75	31.86
DIST [17]	48.40	62.47	37.48	56.92	12.23	41.95	37.86	52.36	27.11	46.50	9.81	34.96
CAT-KD [12]	51.02	63.68	40.23	58.26	14.68	44.84	43.18	57.23	33.17	51.90	11.61	38.45
LTKD	**57.40**	**65.42**	**48.42**	**60.94**	**23.99**	**48.60**	**48.43**	**57.79**	**40.82**	**53.70**	**21.04**	**42.45**
Δ	**+6.16**	**+1.52**	**+7.98**	**+2.31**	**+8.18**	**+3.20**	**+5.14**	**+0.56**	**+7.59**	**+1.50**	**+8.59**	**+3.24**

Table 10. Accuracy (%) on both the tail group classes ($\mathcal{T}$) and the overall classes (All) in the TinyImageNet-LT test sets when using teacher and student models with homogeneous architectures. The best result is highlighted in **bold**, and the second-best result is indicated with underline. Δ denotes the performance gap between the best and the second-best results.

T-S Pairs	ResNet32×4 − ResNet8×4						VGG13 − VGG8					
γ	10		20		100		10		20		100	
Group	$\mathcal{T}$	All	$\mathcal{T}$	All	$\mathcal{T}$	All	$\mathcal{T}$	All	$\mathcal{T}$	All	$\mathcal{T}$	All
Teacher	38.47	52.64	28.74	47.49	9.53	35.37	32.53	45.23	21.85	39.75	6.29	29.71
Student	29.72	44.60	21.46	40.25	4.73	30.62	31.12	43.76	22.19	39.00	6.88	29.96
KD [16]	27.34	45.38	18.11	41.35	3.38	31.42	31.92	47.16	20.60	41.49	3.99	30.95
FitNet [30]	29.65	44.64	20.86	40.24	5.51	30.66	32.04	44.43	23.43	39.47	6.87	30.09
DKD [52]	34.70	48.93	26.58	44.84	9.09	34.61	33.20	48.01	22.65	42.44	5.88	31.82
ReviewKD [6]	32.85	49.13	23.43	44.62	5.39	33.51	34.39	47.66	24.84	42.36	7.61	32.18
DIST [17]	34.81	50.14	25.71	45.52	7.30	33.98	33.48	47.22	23.19	41.46	5.98	31.01
CAT-KD [12]			-				32.56	47.57	22.57	42.17	5.96	31.72
LTKD	**40.66**	**51.33**	**31.33**	**47.05**	**10.48**	**36.21**	**38.90**	**49.43**	**29.30**	**44.22**	**9.73**	**33.78**
Δ	**+5.85**	**+1.19**	**+4.75**	**+1.53**	**+1.39**	**+1.60**	**+4.51**	**+1.42**	**+4.46**	**+1.78**	**+2.12**	**+1.60**

T-S Pairs	WRN–40–2 − WRN–40–1						ResNet110 − ResNet32					
γ	10		20		100		10		20		100	
Group	$\mathcal{T}$	All	$\mathcal{T}$	All	$\mathcal{T}$	All	$\mathcal{T}$	All	$\mathcal{T}$	All	$\mathcal{T}$	All
Teacher	34.21	49.09	23.50	44.15	6.05	33.66	32.78	47.01	23.36	42.08	6.31	32.11
Student	28.97	44.34	20.27	40.16	4.79	30.31	28.54	44.02	20.45	40.10	4.61	30.47
KD [16]	27.35	45.72	17.12	40.67	2.82	31.59	28.87	45.47	19.77	40.60	3.87	30.86
FitNet [30]	28.85	44.25	20.02	39.92	3.74	29.98	28.51	43.52	20.31	39.12	4.97	29.86
DKD [52]	31.92	47.32	21.52	42.83	5.08	33.04	31.26	46.79	22.06	42.03	5.32	32.09
ReviewKD [6]	33.53	48.56	24.00	43.71	5.84	33.15	31.80	47.21	22.33	42.47	5.36	32.23
DIST [17]	32.35	48.26	22.08	43.53	5.26	33.61	32.03	47.07	21.97	42.19	5.44	32.26
CAT-KD [12]	9.13	29.32	3.38	26.06	0.09	19.97	2.12	18.83	0.16	15.79	0.10	14.14
LTKD	**36.55**	**48.86**	**27.70**	**44.49**	**8.91**	**34.80**	**37.19**	**47.85**	**29.08**	**43.68**	**9.68**	**33.80**
Δ	**+3.02**	**+0.30**	**+3.70**	**+0.78**	**+3.07**	**+1.19**	**+5.16**	**+0.64**	**+6.75**	**+1.21**	**+4.24**	**+1.54**

Table 11. Accuracy (%) on both the tail group classes ($\mathcal{T}$) and the overall classes (All) in the TinyImageNet-LT test sets when using teacher and student models with heterogeneous architectures. The best result is highlighted in **bold**, and the second-best result is indicated with underline. Δ denotes the performance gap between the best and the second-best results.

T-S Pairs	ResNet32×4 − ShuffleNetV1						VGG13 − MobileNetV2					
γ	10		20		100		10		20		100	
Group	$\mathcal{T}$	All	$\mathcal{T}$	All	$\mathcal{T}$	All	$\mathcal{T}$	All	$\mathcal{T}$	All	$\mathcal{T}$	All
Teacher	38.47	52.64	28.74	47.49	9.53	35.37	32.53	45.23	21.85	39.75	6.29	29.71
Student	24.43	37.10	16.80	31.85	4.71	22.77	26.98	39.73	17.38	33.14	3.97	22.99
KD [16]	34.67	49.05	24.12	42.81	5.62	30.74	31.24	45.60	19.53	39.50	3.27	28.44
FitNet [30]	25.23	36.79	16.39	32.04	4.84	23.17	25.81	37.98	16.51	31.90	4.07	22.71
DKD [52]	36.83	50.22	26.64	44.38	8.34	33.23	33.07	46.79	22.06	40.87	5.70	29.97
ReviewKD [6]	36.30	49.27	27.09	44.72	8.49	33.12	32.37	44.99	22.77	39.36	7.02	29.20
DIST [17]	36.49	50.07	25.74	43.59	7.23	31.19	32.92	46.09	21.77	40.05	5.52	29.08
CAT-KD [12]			-				27.89	40.45	18.50	34.89	4.27	24.63
LTKD	**42.12**	**51.64**	**33.06**	**46.41**	**12.85**	**35.09**	**39.04**	**48.71**	**28.28**	**43.22**	**9.52**	**32.30**
Δ	**+5.29**	**+1.42**	**+5.97**	**+1.69**	**+4.36**	**+1.86**	**+5.97**	**+1.92**	**+5.51**	**+2.35**	**+2.50**	**+2.33**

T-S Pairs	WRN–40–2 − ShuffleNetV1						ResNet50 − MobileNetV2					
γ	10		20		100		10		20		100	
Group	$\mathcal{T}$	All	$\mathcal{T}$	All	$\mathcal{T}$	All	$\mathcal{T}$	All	$\mathcal{T}$	All	$\mathcal{T}$	All
Teacher	34.21	49.09	23.50	44.15	6.05	33.66	39.43	52.86	29.37	46.36	10.71	35.49
Student	24.43	37.10	16.80	31.85	4.71	22.77	26.98	39.73	17.38	33.14	3.97	22.99
KD [16]	32.28	47.64	21.72	42.54	4.64	32.09	34.09	48.70	23.33	42.23	5.49	30.23
FitNet [30]	24.48	36.57	16.59	31.59	5.03	23.02	25.66	38.16	16.53	32.07	4.00	22.29
DKD [52]	34.45	48.68	23.40	43.78	6.55	33.25	37.07	50.37	27.11	44.30	8.56	33.01
ReviewKD [6]	35.89	48.94	26.54	43.93	8.67	33.61	30.26	42.84	20.97	36.99	5.50	26.67
DIST [17]	33.71	48.74	23.19	43.61	5.83	32.74	36.31	50.03	25.73	43.32	7.38	31.13
CAT-KD [12]	28.69	43.16	18.88	37.70	1.85	27.14	30.15	43.99	20.35	38.04	0.76	26.58
LTKD	**39.66**	**50.19**	**30.05**	**45.58**	**11.13**	**35.39**	**42.59**	**51.95**	**31.95**	**45.66**	**11.97**	**34.60**
Δ	**+3.77**	**+1.25**	**+3.51**	**+1.65**	**+2.46**	**+1.78**	**+5.52**	**+1.58**	**+4.84**	**+1.36**	**+3.41**	**+1.59**

E. PyTorch implementation

E.1. Long-tailed knowledge distillation

To ensure reproducibility, the proposed LTKD framework is implemented on top of the open-source *mdistiller* codebase[1].

```python
import torch
import torch.nn.functional as F
from ._base import Distiller

class LTKD(Distiller):
    """
    [Hyperparameters]
    - CIFAR-100-LT & TinyImageNet-LT : TEMPERATURE = 4.0, WARMUP = 20
    - ImageNet-LT                    : TEMPERATURE = 1.0, WARMUP = 1
    """
    def __init__(self, student, teacher, cfg):
        super(LTKD, self).__init__(student, teacher)
        self.alpha = cfg.LTKD.ALPHA
        self.beta = cfg.LTKD.BETA
        self.temperature = cfg.LTKD.TEMPERATURE
        self.warmup = cfg.LTKD.WARMUP
        self.dataset = cfg.DATASET.TYPE

    def forward_train(self, image, target, **kwargs):
        logits_student, _ = self.student(image)
        with torch.no_grad():
            logits_teacher, _ = self.teacher(image)

        # Cross-Entropy Loss
        loss_ce = F.cross_entropy(logits_student, target)

        # LTKD Loss (see Fig. 8)
        loss_ltkd = min(kwargs["epoch"] / self.warmup, 1.0) * ltkd_loss(
            self.dataset,
            logits_student,
            logits_teacher,
            self.alpha,
            self.beta,
            self.temperature,
        )

        losses_dict = {
            "loss_ce": loss_ce,
            "loss_kd": loss_ltkd,
        }

        return logits_student, losses_dict
```

Figure 7. PyTorch implementation of the proposed LTKD class based on the *mdistiller* codebase.

[1] https://github.com/megvii-research/mdistiller

```python
def ltkd_loss(dataset, logits_student, logits_teacher, alpha, beta, temperature):
    """
    Implementation of LTKD loss.
    """
    pred_teacher = F.softmax(logits_teacher / temperature, dim=1)
    pred_student = F.softmax(logits_student / temperature, dim=1)

    # Class indices for Head, Medium, and Tail groups
    if dataset == 'CIFAR-100-LT':
        head_idx = torch.arange(0, 33, device=logits_student.device)
        med_idx = torch.arange(33, 67, device=logits_student.device)
        tail_idx = torch.arange(67, 100, device=logits_student.device)
    # Other datasets (TinyImageNet-LT, ImageNet-LT) are handled similarly.

    # Within-Group Loss
    def group_softmax(logits, idx, temperature):
        return F.softmax(logits[:, idx] / temperature, dim=1)

    def group_loss(logits_student, logits_teacher, idx, temperature):
        pred_teacher_group = group_softmax(logits_teacher, idx, temperature)
        pred_student_group = group_softmax(logits_student, idx, temperature)
        loss = (F.kl_div(torch.log(pred_student_group), pred_teacher_group, size_average=False)
                * (temperature**2)
                / logits_student.shape[0])
        return loss

    head_loss = group_loss(logits_student, logits_teacher, head_idx, temperature)
    med_loss = group_loss(logits_student, logits_teacher, med_idx, temperature)
    tail_loss = group_loss(logits_student, logits_teacher, tail_idx, temperature)

    # Cross-Group Loss
    def group_sum(pred, idx):
        return pred[:, idx].sum(dim=1)

    b_teacher = torch.stack([group_sum(pred_teacher, head_idx),
                             group_sum(pred_teacher, med_idx),
                             group_sum(pred_teacher, tail_idx)],
                             dim=1)

    w_alpha, w_beta, w_gamma = b_teacher.sum(0).mean() / b_teacher.sum(0)

    weighted_b_teacher = torch.stack([group_sum(pred_teacher, head_idx)*w_alpha,
                             group_sum(pred_teacher, med_idx)*w_beta,
                             group_sum(pred_teacher, tail_idx)*w_gamma],
                             dim=1)
    weighted_b_teacher = weighted_b_teacher / weighted_b_teacher.sum(1)[:, None]

    b_student = torch.stack([group_sum(pred_student, head_idx),
                             group_sum(pred_student, med_idx),
                             group_sum(pred_student, tail_idx)],
                             dim=1)

    cross_group_loss = (F.kl_div(torch.log(b_student), weighted_b_teacher, size_average=False)
            * (temperature**2)
            / logits_student.shape[0])

    # LTKD Loss
    return alpha*cross_group_loss + beta*(head_loss + med_loss + tail_loss)
```

Figure 8. PyTorch implementation of the proposed LTKD loss.

E.2. Long-tailed dataset construction

For all long-tailed datasets (CIFAR-100-LT, TinyImageNet-LT, and ImageNet-LT), the imbalance is constructed by applying the exponential sub-sampling strategy to the standard training sets. While we provide the implementation of CIFAR-100-LT in Fig. 9, the same sub-sampling strategy is consistently applied to TinyImageNet-LT and ImageNet-LT.

During training, we apply dataset-specific standard augmentations: `RandomCrop(32, padding=4)` and `RandomHorizontalFlip()` for CIFAR-100-LT; `RandomRotation(20)` and `RandomHorizontalFlip(0.5)` for TinyImageNet-LT; `RandomResizedCrop(224)` and `RandomHorizontalFlip()` for ImageNet-LT.

```python
import numpy as np
from torch.utils.data import Dataset
from torchvision.datasets import CIFAR100

class CIFAR100_LT(Dataset):
    """
    Implementation of Long-Tailed CIFAR-100 dataset.
    The imbalance is constructed by exponentially decaying
    the number of training samples per class based on the `imb_factor`.
    """
    def __init__(self, root, download=True, train=True, transform=None, imb_factor=0.01):
        self.transform = transform
        self.train = train
        self.imb_factor = imb_factor
        self.num_classes = 100

        base_dataset = CIFAR100(root=root, train=train, download=download)
        self.data = base_dataset.data
        self.targets = np.array(base_dataset.targets)

        if train:
            self.gen_imbalanced_data()

    def get_img_num_per_cls(self, num_classes, imb_factor):
        img_max = len(self.data) / num_classes
        img_num_per_cls = []
        for cls_idx in range(num_classes):
            num = img_max * (imb_factor ** (cls_idx / (num_classes - 1.0)))
            img_num_per_cls.append(int(num))
        return img_num_per_cls

    def gen_imbalanced_data(self):
        img_num_per_cls = self.get_img_num_per_cls(self.num_classes, self.imb_factor)
        new_data, new_targets = [], []
        targets_np = np.array(self.targets, dtype=np.int64)
        classes = np.arange(self.num_classes)

        for cls_idx, img_num in zip(classes, img_num_per_cls):
            idx = np.where(targets_np == cls_idx)[0]
            np.random.shuffle(idx)
            selec_idx = idx[:img_num]

            new_data.append(self.data[selec_idx])
            new_targets.extend([cls_idx] * img_num)

        self.data = np.vstack(new_data)
        self.targets = np.array(new_targets)

    # Standard __len__ and __getitem__ functions are omitted for conciseness.
```

Figure 9. PyTorch implementation of the long-tailed dataset construction.