RETHINKING INTERMEDIATE SUPERVISION IN FEDERATED LEARNING WITH ADAPTIVE DISTILLATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Federated Learning (FL) enables multiple clients to collaboratively train a shared model while keeping their data local. Recent FL methods increasingly introduce supervision at intermediate layers, yet the effects of different forms of supervision in FL remain unclear. We examine Cross-Entropy (CE) and Knowledge Distillation (KD) for intermediate branch supervision in FL. While CE directly uses hard labels, KD provides richer guidance through the final classifier's soft predictions. Our analysis shows that this distinction becomes particularly important under federated local training, where CE is more sensitive to limited local data while KD provides more stable guidance. Motivated by these findings, we adopt KD for intermediate supervision. Its appropriate strength, however, can vary across federated settings and during training, so we propose adaptive KD weighting that adjusts the distillation strength to local training conditions. Experiments across multiple datasets and FL algorithms show that the proposed approach provides consistent performance across diverse federated settings.

1 INTRODUCTION

Federated Learning (FL) is a learning framework in which multiple clients jointly train a shared model using their locally stored data (McMahan et al., 2017). During training, each client updates the model on its own data, and the resulting updates are aggregated at a central server to update the shared model. Since the raw data remain on the clients rather than being collected at the server, FL allows collaborative model training while preserving data privacy and reducing the need for centralized data storage and processing.

Recent FL studies have increasingly incorporated intermediate layers into local training rather than relying solely on the final classifier. For example, FedACL (Gao et al., 2022) uses auxiliary classifiers in a multi-distillation framework for heterogeneous clients, while HYDRA-FL (Khan et al., 2024) distributes distillation through shallow auxiliary classifiers to improve robustness against model poisoning. Such branches are commonly supervised using Cross-Entropy (CE) with ground-truth labels or Knowledge Distillation (KD) from a teacher. However, their distinct effects on intermediate branch learning in FL remain underexplored.

We examine how CE and KD differ as supervision objectives for intermediate branches. CE directly trains each branch toward the ground-truth class, whereas KD guides the branch using the prediction distribution of the final classifier. Our gradient analysis shows that branch CE adds a hard-label correction based on the same prediction residual as final CE, whereas KD provides a distinct teacher-guided signal. This distinction is particularly relevant when intermediate representations are not yet sufficiently discriminative: hard-label supervision can be restrictive, whereas KD retains the teacher's uncertainty and relative class predictions.

In FL, each client optimizes the branch objectives using only a limited local dataset. As a result, branch CE becomes more sensitive to particular local samples. In contrast, KD produces local gradients that remain more aligned with full-data gradients and reduces the sensitivity of intermediate predictions to differences in local class frequency. These findings support the use of KD as the supervision objective for intermediate branch predictions.

However, the appropriate strength of KD can vary across federated settings and with local training conditions across clients and communication rounds. We therefore propose an adaptive weighting

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

strategy that adjusts the KD strength based on training progress, teacher reliability, and intermediate prediction discrepancy. By using locally available signals, the proposed approach provides a practical way to adapt the distillation strength to changing client and training conditions without requiring additional client-server communication.

Our main contributions are summarized as follows:

- We analyze how CE and KD differ as intermediate branch supervision. Our analysis reveals distinct learning signals from the two objectives and shows when direct hard-label supervision from CE can become restrictive at intermediate depths.
- We extend this analysis to federated local training, where limited client data amplify the limitations of branch CE. Our results show that KD provides more reliable intermediate supervision under varying local data conditions.
- Building on these findings, we propose an adaptive KD weighting strategy that adjusts the distillation strength to local training conditions. Experiments across diverse federated settings show that adaptive weighting provides more consistent performance than fixed weighting.

2 Related Work

Intermediate supervision has been widely explored to provide additional learning signals at different network depths. Early approaches such as GoogLeNet (Szegedy et al., 2015) and Deeply-Supervised Nets (Lee et al., 2015) attach auxiliary classifiers to intermediate layers and optimize their predictions using ground-truth labels. Beyond direct label supervision, Zhang et al. (2019) introduce self-distillation across multiple depths by supervising intermediate branches with both ground-truth labels and final-classifier predictions. Several studies have also examined the limitations of direct intermediate classification. HSD (Hu et al., 2021) replaces hard-label supervision with soft predictions from deeper classifiers, while Contrastive Deep Supervision (Zhang et al., 2022) uses contrastive objectives instead of direct intermediate classification. These studies show that the choice of supervision objective can influence intermediate learning.

Building on these developments, intermediate supervision has also been adopted in federated learning. While knowledge distillation in FL is commonly used for knowledge transfer between global and local models (Lee et al., 2022), recent studies have also applied auxiliary supervision across network depths within local models. FedACL (Gao et al., 2022) uses auxiliary classifiers and self-distillation to address model and statistical heterogeneity, while early-exit approaches exploit predictions at multiple depths for resource-heterogeneous clients (Ilhan et al., 2023; Lee et al., 2024). Intermediate supervision has also been explored for personalized federated learning (LI et al., 2025) and robust federated learning (Khan et al., 2024). These studies apply intermediate supervision to different challenges in FL, but they typically adopt a particular supervision objective without analyzing its effect on intermediate branch learning. We address this gap by directly comparing CE and KD and analyzing how their supervision signals differ under federated local training.

3 Preliminaries

3.1 Federated Learning

We consider a federated learning setting with K clients, where client k holds a local dataset $\mathcal{D}_k$ of size n_k. At communication round t, the server distributes the global model θ^t to a set of participating clients $\mathcal{S}_t$. Each client initializes its local model from θ^t and performs local optimization on its own data. Under FedAvg, the resulting local models are aggregated as

$$\theta^{t+1} = \sum_{k \in \mathcal{S}_t} \frac{n_k}{\sum_{j \in \mathcal{S}_t} n_j} \theta_k^{t+1}, \tag{1}$$

where θ_k^{t+1} denotes the model obtained by client k after local training.

3.2 BRANCH-LEVEL SELF-DISTILLATION

We consider a network with auxiliary branches attached to intermediate representations. Let z_b denote the logits produced by branch b, and let z_T denote the logits of the final classifier, which serves as the teacher for the intermediate branches. For a distillation temperature τ, we define

$$p_{b,\tau} = \mathrm{softmax}(z_b/\tau), \qquad p_{T,\tau} = \mathrm{softmax}(z_T/\tau), \tag{2}$$

and use $p_b = p_{b,1}$ and $p_T = p_{T,1}$ to denote the original prediction distributions.

For a sample (x, Y), let e_Y denote the one-hot ground-truth label. We consider Cross-Entropy (CE) and Knowledge Distillation (KD) for intermediate branch supervision:

$$\mathcal{L}_b^{\mathrm{CE}} = -e_Y^\top \log p_b, \qquad \mathcal{L}_b^{\mathrm{KD}} = -\left(p_{T,\tau}\right)^\top \log p_{b,\tau}. \tag{3}$$

CE uses the ground-truth label as the target, whereas KD uses the final classifier's prediction distribution adjusted by the distillation temperature.

The local objective for client k is

$$\mathcal{L}_k = \mathbb{E}_{(x,Y)\sim\mathcal{D}_k}\left[\mathcal{L}_T^{\mathrm{CE}} + \sum_b \left[(1-\alpha)\mathcal{L}_b^{\mathrm{CE}} + \alpha\mathcal{L}_b^{\mathrm{KD}}\right]\right], \tag{4}$$

where $\mathcal{L}_T^{\mathrm{CE}}$ trains the final classifier and $\alpha \in [0, 1]$ balances branch CE and KD.

4 ANALYSIS

We analyze how CE and KD differ as intermediate branch supervision and how these differences appear under federated training. We first examine their gradient relation to final CE and hard-label supervision at intermediate representations, then consider the effects of limited local data, and finally evaluate their behavior under federated training.

4.1 ANALYSIS OF BRANCH CE AND KD SUPERVISION

Gradient relation to final CE. For simplicity, we set the distillation temperature to $\tau = 1$ and write the CE and KD gradients with respect to the logits of branch b as

$$g_b^{\mathrm{CE}} = p_b - e_Y, \qquad g_b^{\mathrm{KD}} = p_b - p_T. \tag{5}$$

Accordingly, the gradient of the branch prediction objective in Eq. 4 can be written as

$$g_{b,\alpha}^{\mathrm{pred}} = (1-\alpha)\underbrace{\left[(p_b - p_T) + (p_T - e_Y)\right]}_{g_b^{\mathrm{CE}}} + \alpha\underbrace{(p_b - p_T)}_{g_b^{\mathrm{KD}}} = (p_b - p_T) + (1-\alpha)(p_T - e_Y). \tag{6}$$

This decomposition separates the branch gradient into the KD term $p_b - p_T$ and the hard-label correction $(p_T - e_Y)$, which is also the final CE logit gradient. Decreasing α therefore increases the contribution of the final CE residual, making branch CE more closely related to final CE while branch KD provides a more distinct signal.

We next examine whether this relationship extends to backbone gradients. For branch b and objective $o \in \{\mathrm{CE}, \mathrm{KD}\}$, let G_b^o denote the branch gradient over the preceding backbone parameters and $G_{T,b}^{\mathrm{CE}}$ the corresponding final CE gradient. We measure their cosine similarity on ResNet18 trained on CIFAR-100 under centralized and federated training. As shown in Table 1, branch CE is more aligned with final CE than branch KD across all branches and settings, consistent with Eq. 6. Branch CE provides a signal more closely related to final CE, whereas branch KD provides a more distinct direction.

Hard-Label Supervision at Intermediate Representations. The effect of hard-label supervision depends on the information available at each depth. Earlier representations are generally less task-specific and may not yet separate classes well, while branch CE applies the same hard-label objective across depths. This can make direct CE supervision restrictive at earlier branches.

Table 1: Gradient alignment with final CE on CIFAR-100 with KD at $\tau = 1$. B1–B3 denote the three auxiliary branches, and Combined denotes their summed gradient.

	Centralized		IID		$\beta = 0.1$	
	CE	KD	CE	KD	CE	KD
B1	0.43	0.26	0.54	0.08	0.61	-0.13
B2	0.44	0.28	0.68	0.03	0.84	0.14
B3	0.84	-0.28	0.93	0.01	0.97	0.10
Combined	0.62	0.04	0.66	0.04	0.82	0.04

To characterize how this affects the CE gradient, let U_b denote the representation at depth b and $\eta_b(u) = P(Y \mid U_b = u)$ its conditional label distribution. For a given representation $U_b = u$, the CE gradient has conditional mean $p_b(u) - \eta_b(u)$ and variance

$$\mathbb{E}\left[\left\|g_b^{\mathrm{CE}} - \mathbb{E}[g_b^{\mathrm{CE}} \mid U_b = u]\right\|_2^2 \mid U_b = u\right] = 1 - \|\eta_b(u)\|_2^2. \tag{7}$$

When $\eta_b(u)$ is concentrated, samples sharing u tend to have the same label and receive similar hard-label corrections. As $\eta_b(u)$ becomes more distributed, similar representations can receive different one-hot targets, increasing the variability in Eq. 7. KD instead uses the final prediction distribution, so the same variability from hard labels does not arise.

To assess the classification capability of intermediate representations, we use linear probes following prior analyses of representations across network depths (Alain & Bengio, 2018). On a backbone trained only with the final classification objective, probe accuracy is 5.49% at B1 and 13.19% at B2, compared with 76.90% at the final representation; full results are provided in Appendix B.1. This provides empirical evidence that earlier representations can have limited classification capability, where direct branch CE can be restrictive.

4.2 Branch Supervision with Limited Local Data

In federated learning, each client estimates the branch objectives from a finite local dataset. We examine the finite-sample effects of the two properties in Section 4.1 under IID sampling to isolate the effect of limited local data.

For the relation to final CE, let $\epsilon_{b,k}^{o} = \hat{g}_{b,k}^{o} - g_b^{o,*}$ denote the finite-sample error for objective $o \in \{\mathrm{CE}, \mathrm{KD}\}$, and let $\epsilon_{T,k}^{\mathrm{CE}}$ denote the corresponding error for final CE. Using Eq. 6, the difference in their finite-sample error alignment with final CE can be written as

$$\mathbb{E}\left[\epsilon_{b,k}^{\mathrm{CE}\top}\epsilon_{T,k}^{\mathrm{CE}}\right] - \mathbb{E}\left[\epsilon_{b,k}^{\mathrm{KD}\top}\epsilon_{T,k}^{\mathrm{CE}}\right] = \frac{1}{n_k}\operatorname{tr}\operatorname{Var}\left(g_T^{\mathrm{CE}}\right). \tag{8}$$

Thus, branch CE contains an additional finite-sample error component shared with final CE, whose magnitude increases as n_k decreases.

For hard-label variability, the local CE gradient estimation error satisfies

$$\mathbb{E}\left[\|\epsilon_{b,k}^{\mathrm{CE}}\|_2^2\right] = \frac{1}{n_k}\operatorname{tr}\operatorname{Var}\left(g_b^{\mathrm{CE}}\right). \tag{9}$$

This variance includes $\mathbb{E}_{U_b}[1 - \|\eta_b(U_b)\|_2^2]$ from Eq. 7. As n_k decreases, the contribution of this hard-label variability to the local CE gradient estimation error becomes larger. Detailed derivations are provided in Appendix A.1 and Appendix A.2.

Together, these results show that limited local data can strengthen the finite-sample effects associated with the two properties identified in Section 4.1. As a result, branch CE can become more sensitive to the particular local samples and provide a more variable estimate of the full-data gradient.

4.3 Branch CE and KD under Federated Training

Client-Level Gradient and Prediction Behavior. We first examine whether the finite-sample effects identified in Section 4.2 also appear in client gradients during federated training. We compare

Table 2: Client-level gradient agreement and rare–frequent JS gap on CIFAR-100 with KD at $\tau = 1$.

(a) Gradient Agreement ↑

Objective	IID	$\beta = 0.1$
CE	0.38	0.36
KD	**0.63**	**0.74**

(b) Rare–Frequent JS Gap ↓ ($\beta = 0.1$)

Objective	B1	B2	B3	Final
CE	0.25	0.27	0.26	0.26
KD	**0.23**	**0.24**	**0.23**	0.26

Table 3: Early-branch and final accuracy (%) on CIFAR-100. B1 and B2 are reported as representative earlier branches. Values in parentheses denote τ.

	Central			IID			$\beta = 0.1$		
Objective	B1	B2	Final	B1	B2	Final	B1	B2	Final
CE	61.90	72.10	77.55	54.16	64.52	71.69	**39.23**	**45.70**	51.67
KD (1.0)	**62.34**	**73.07**	77.41	**55.92**	**65.53**	71.19	36.16	44.61	52.18
KD (0.5)	58.00	70.32	**77.61**	52.53	62.87	**72.42**	33.96	41.93	**52.62**

each client's branch gradient computed from local data with the corresponding gradient computed on the full test set at the same global checkpoint. Their agreement is measured by the mean cosine similarity across clients. As shown in Table 2(a), KD shows higher agreement than CE under both IID and $\beta = 0.1$, with a larger difference under $\beta = 0.1$. This indicates that KD preserves the corresponding full-data gradient direction more consistently when estimated from clients' local data. Branch-wise results across communication rounds are provided in Appendix B.2.

Under $\beta = 0.1$, we further examine prediction consistency across clients using the rare–frequent JS gap, which measures how much larger prediction divergence is for locally rare classes than for frequent classes; its detailed definition is provided in Appendix B.3. As shown in Table 2(b), KD shows a smaller gap than CE at the intermediate branches, while the final predictor remains unchanged. This indicates that KD intermediate predictions are less sensitive to local class frequency. This is relevant in FL, where the same class can be frequent on some clients but rare on others.

Performance under Federated Training. Finally, we compare branch CE and branch KD in model performance under centralized and federated training. As shown in Table 3, their final accuracy is similar under centralized training, whereas KD shows a clearer advantage under FL.

The performance pattern does not directly follow gradient alignment in Table 1. Although KD at $\tau = 1$ provides a more distinct direction from final CE, its final accuracy remains close to CE, whereas $\tau = 0.5$ yields a clearer improvement. Lowering the temperature sharpens the target distribution, which can reduce the influence of uncertainty in the final predictor. In addition, this improvement does not come with higher branch accuracy: KD with $\tau = 0.5$ achieves lower B1 and B2 accuracy than CE and KD with $\tau = 1$ in most cases, while providing comparable or higher final accuracy. Thus, effective intermediate supervision is not determined by gradient distinctness or branch classification strength alone.

Overall, branch CE can become more sensitive to local samples under limited data, especially when intermediate representations have limited classification capability, whereas KD provides more stable intermediate supervision across clients.

5 Proposed Method

5.1 KD-Only Branch Supervision

Following the preceding analysis, we use KD as the prediction objective for intermediate branches and remove direct branch CE:

$$\mathcal{L}_{k,t} = \mathcal{L}_T^{\mathrm{CE}} + \sum_b \lambda \mathcal{L}_b^{\mathrm{KD}}, \tag{10}$$

Table 4: CIFAR-100 accuracy (%) with different fixed KD weights. Best results are bolded and second-best results are underlined.

	KD weight λ					
Partition	0	0.1	0.3	0.5	1	3
IID	69.61	71.71	**72.39**	<u>71.96</u>	71.16	70.23
$\beta = 0.3$	63.85	<u>65.73</u>	**66.33**	65.71	65.17	64.55
$\beta = 0.1$	<u>54.69</u>	**54.71**	52.92	52.24	52.50	52.65

where $\lambda \geq 0$ controls the strength of branch KD. Unlike α in Eq. 4, which balances branch CE and KD, λ directly scales KD after branch CE is removed.

As shown in Table 4, performance does not consistently improve with larger KD weights and the optimal weight varies across data partitions. However, selecting λ separately for each setting is difficult in practice because the underlying degree of data heterogeneity is generally unknown in advance. Moreover, even under a fixed data partition, the appropriate KD strength can change across clients and communication rounds. We therefore determine the KD weight adaptively from information available during local training.

5.2 Adaptive KD Weighting

For client k at communication round t, we define

$$\lambda_{k,t} = \lambda_t \cdot \tilde{r}_{k,t} \cdot d_{k,t}, \tag{11}$$

where λ_t controls the round-wise KD strength, $\tilde{r}_{k,t}$ reflects teacher reliability, and $d_{k,t}$ captures the intermediate–final prediction discrepancy.

Round-level warm-up. During the early communication rounds, the final classifier is not yet sufficiently trained and its predictions can change substantially. Strong KD at this stage may therefore provide unreliable guidance to the intermediate branches. We therefore gradually increase the base KD strength as

$$\lambda_t = \lambda_{\max} \min\left(\frac{t}{t_{\text{warm}}}, 1\right), \tag{12}$$

where $\lambda_{\max}$ is the maximum KD weight and t_{warm} is the warm-up period. This limits KD early in training and gradually increases its influence as the final classifier improves.

Teacher reliability. Even at the same communication round, the quality of the final classifier can differ across clients. We use the average probability assigned to the ground-truth class as a local proxy for teacher reliability:

$$r_{k,t} = \frac{1}{|\mathcal{D}_k|} \sum_{(x_i, y_i) \in \mathcal{D}_k} p_{T,k,t}(y_i \mid x_i), \tag{13}$$

where $p_{T,k,t}$ denotes the final classifier prediction for client k at round t.

Since frequently occurring classes can have greater influence on $r_{k,t}$, we additionally account for the class distribution of teacher predictions. We compute their mean distribution and normalized entropy as

$$\bar{p}_{k,t} = \frac{1}{|\mathcal{D}_k|} \sum_{x_i \in \mathcal{D}_k} p_{T,k,t}(\cdot \mid x_i), \qquad b_{k,t} = \frac{-\sum_{c=1}^{C} \bar{p}_{k,t,c} \log \bar{p}_{k,t,c}}{\log C}. \tag{14}$$

A smaller $b_{k,t}$ indicates stronger concentration on a limited subset of classes. We then define

$$g_{k,t} = \sigma\left(\frac{b_{k,t} - \kappa}{T_s}\right), \qquad s_{k,t} = (1 - g_{k,t}) b_{k,t}^2 + g_{k,t}, \qquad \tilde{r}_{k,t} = r_{k,t} s_{k,t}, \tag{15}$$

where κ determines the transition point and T_s controls its sharpness. Thus, $\tilde{r}_{k,t}$ remains close to $r_{k,t}$ for balanced predictions and decreases as they become more concentrated.

Intermediate prediction discrepancy. The appropriate KD strength also depends on the remaining discrepancy between the intermediate and final predictions. When the intermediate predictions are already close to the final classifier, strong KD may unnecessarily constrain them toward the final prediction. We therefore measure the remaining discrepancy using the average normalized Jensen–Shannon (JS) divergence across branches:

$$d_{k,t} = \frac{1}{B}\sum_{b=1}^{B}\frac{1}{|\mathcal{D}_k|}\sum_{x_i\in\mathcal{D}_k}\frac{\mathrm{JS}\left(p_{T,k,t}(\cdot \mid x_i), p_{b,k,t}(\cdot \mid x_i)\right)}{\log 2}, \tag{16}$$

where B is the number of intermediate branches and $p_{b,k,t}$ is the prediction distribution of branch b. A larger $d_{k,t} \in [0,1]$ retains more of the KD weight, while a smaller value reduces it when the intermediate predictions are already close to the teacher.

Since all adaptive factors are computed using information available during local training, adaptive weighting itself requires no additional client–server communication and leaves the server-side aggregation procedure unchanged. We report the computational and communication overhead of auxiliary-branch training in Appendix F.

6 Experiments

6.1 Experimental Setup

We evaluate the proposed method on four image classification datasets: CIFAR-10 and CIFAR-100 (Krizhevsky et al., 2009), TinyImageNet (Le et al., 2015), and ImageNet100-64, constructed from ImageNet (Deng et al., 2009). We consider both IID and Dirichlet non-IID data partitions, using $\beta = 0.1$ for the non-IID setting. Unless otherwise specified, we use 100 clients, with 10% of the clients participating in each communication round and performing five local epochs before aggregation. For the main performance evaluations, results are reported as the average global test accuracy over the last 30 communication rounds. Additional implementation details are provided in Appendix C.

We use ResNet18 (He et al., 2016) as the default backbone and attach three auxiliary branches after the first three residual stages. The final classifier serves as the teacher for all intermediate branches, and we set the KD temperature to $\tau = 1$, adjusting only the strength of the distillation signal through adaptive weighting. We evaluate five FL algorithms: FedAvg (McMahan et al., 2017), FedProx (Li et al., 2020), MOON (Li et al., 2021), SCAFFOLD (Karimireddy et al., 2020), and FedLC (Lee & Seo, 2023). Additional results with ResNet50 and MobileNetV2 are provided in Appendix D.

6.2 Performance across Datasets and FL Algorithms

We compare three training schemes: *Plain*, which uses only the final classification objective; *Fixed*, which applies intermediate KD with a fixed weight $\lambda = 0.3$; and *Adaptive*, which uses the proposed adaptive weighting strategy. Using ResNet18, we evaluate the three methods across four datasets and five FL algorithms under both IID and non-IID data partitions. As shown in Table 5, adaptive weighting improves over Plain in most settings and outperforms Fixed in nearly all settings. The same pattern is observed across the evaluated FL algorithms. Fixed weighting can perform slightly better in some settings, particularly on CIFAR-100 under IID, but its effectiveness varies more across datasets and data partitions. This is consistent with Table 4, where the preferred KD weight differs across data partitions. Overall, adaptive weighting provides more consistent improvements without setting-specific tuning.

The benefit of adaptive intermediate KD is particularly pronounced on the more challenging datasets. On TinyImageNet and ImageNet100-64, both fixed and adaptive weighting provide substantial improvements over Plain, with adaptive weighting achieving the highest accuracy across all evaluated settings. On CIFAR-10 and CIFAR-100, the improvements are smaller and more dependent on the data partition. This pattern is consistent with our analysis that intermediate supervision becomes more useful when classification at intermediate depths is difficult.

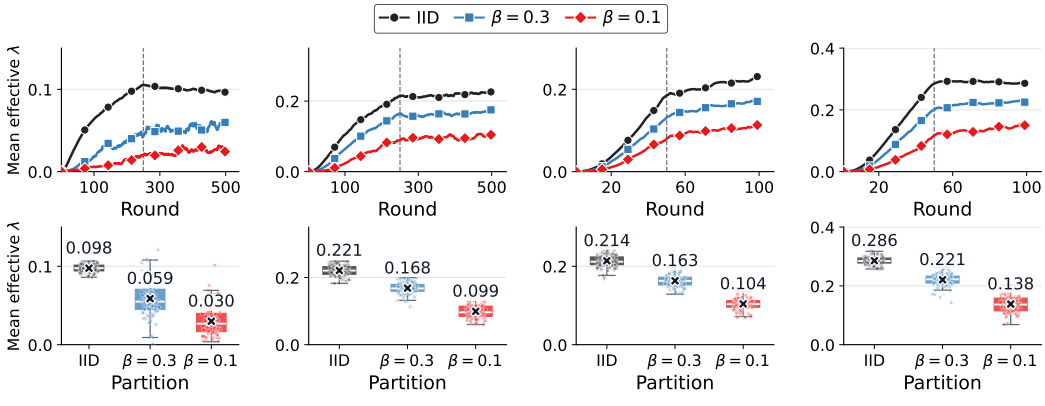
Table 5: Global test accuracy (%) with ResNet18 over three seeds. Standard deviations are shown in smaller text; best and second-best means are bolded and underlined. Tiny-IN and IN100-64 denote TinyImageNet and ImageNet100-64.

	CIFAR-10		CIFAR-100		Tiny-IN		IN100-64	
Method	IID	$\beta = 0.1$	IID	$\beta = 0.1$	IID	$\beta = 0.1$	IID	$\beta = 0.1$
FedAvg	$92.0_{\pm 0.05}$	$\underline{31.6}_{\pm 2.27}$	$70.3_{\pm 0.04}$	$\mathbf{55.0}_{\pm 0.91}$	$41.1_{\pm 0.23}$	$34.0_{\pm 0.21}$	$56.9_{\pm 0.34}$	$43.7_{\pm 0.40}$
+ Fixed	$\underline{92.4}_{\pm 0.09}$	$18.0_{\pm 0.19}$	$\mathbf{72.4}_{\pm 0.06}$	$52.5_{\pm 1.26}$	$\underline{45.5}_{\pm 0.09}$	$\underline{35.3}_{\pm 0.14}$	$\underline{61.4}_{\pm 0.38}$	$\underline{44.1}_{\pm 0.41}$
+ Adaptive	$\mathbf{92.5}_{\pm 0.07}$	$\mathbf{32.4}_{\pm 3.41}$	$\underline{72.2}_{\pm 0.06}$	$\underline{54.8}_{\pm 0.75}$	$\mathbf{45.9}_{\pm 0.43}$	$\mathbf{36.6}_{\pm 0.15}$	$\mathbf{61.6}_{\pm 0.16}$	$\mathbf{45.6}_{\pm 0.43}$
FedProx	$92.1_{\pm 0.02}$	$\mathbf{40.3}_{\pm 3.87}$	$70.4_{\pm 0.16}$	$\mathbf{56.3}_{\pm 0.77}$	$40.8_{\pm 0.28}$	$34.2_{\pm 0.18}$	$56.6_{\pm 0.58}$	$43.9_{\pm 0.45}$
+ Fixed	$\underline{92.5}_{\pm 0.15}$	$27.5_{\pm 1.86}$	$\mathbf{72.5}_{\pm 0.12}$	$54.5_{\pm 1.27}$	$\underline{45.4}_{\pm 0.15}$	$\underline{35.5}_{\pm 0.20}$	$\underline{61.1}_{\pm 0.13}$	$\underline{44.6}_{\pm 0.69}$
+ Adaptive	$\mathbf{92.5}_{\pm 0.10}$	$\underline{40.0}_{\pm 4.02}$	$\underline{72.3}_{\pm 0.16}$	$\underline{56.3}_{\pm 0.92}$	$\mathbf{45.9}_{\pm 0.45}$	$\mathbf{36.7}_{\pm 0.26}$	$\mathbf{61.4}_{\pm 0.06}$	$\mathbf{46.1}_{\pm 0.73}$
MOON	$92.1_{\pm 0.06}$	$\mathbf{31.7}_{\pm 2.10}$	$70.3_{\pm 0.23}$	$\mathbf{55.1}_{\pm 1.08}$	$41.1_{\pm 0.07}$	$34.0_{\pm 0.27}$	$57.0_{\pm 0.57}$	$43.8_{\pm 0.42}$
+ Fixed	$\underline{92.4}_{\pm 0.09}$	$18.5_{\pm 0.54}$	$\mathbf{72.5}_{\pm 0.15}$	$52.6_{\pm 0.96}$	$\underline{45.5}_{\pm 0.09}$	$\underline{35.2}_{\pm 0.04}$	$\underline{61.5}_{\pm 0.15}$	$\underline{44.1}_{\pm 0.50}$
+ Adaptive	$\mathbf{92.5}_{\pm 0.10}$	$\underline{31.3}_{\pm 2.63}$	$\underline{72.3}_{\pm 0.03}$	$\underline{54.7}_{\pm 0.78}$	$\mathbf{45.9}_{\pm 0.40}$	$\mathbf{36.6}_{\pm 0.30}$	$\mathbf{61.6}_{\pm 0.41}$	$\mathbf{45.7}_{\pm 0.39}$
SCAFFOLD†	$93.2_{\pm 0.05}$	$\underline{81.4}_{\pm 0.62}$	$72.8_{\pm 0.13}$	$68.6_{\pm 0.35}$	$25.7_{\pm 0.16}$	$18.8_{\pm 0.12}$	$37.7_{\pm 0.21}$	$28.5_{\pm 0.34}$
+ Fixed	$\underline{94.1}_{\pm 0.05}$	$78.4_{\pm 0.99}$	$\underline{75.9}_{\pm 0.17}$	$\underline{70.0}_{\pm 0.36}$	$\underline{36.8}_{\pm 0.23}$	$\underline{30.2}_{\pm 0.22}$	$\underline{52.4}_{\pm 0.50}$	$\underline{41.0}_{\pm 0.80}$
+ Adaptive	$\mathbf{94.2}_{\pm 0.05}$	$\mathbf{82.3}_{\pm 0.92}$	$\mathbf{76.1}_{\pm 0.16}$	$\mathbf{71.2}_{\pm 0.32}$	$\mathbf{37.7}_{\pm 0.10}$	$\mathbf{31.4}_{\pm 0.15}$	$\mathbf{53.1}_{\pm 0.35}$	$\mathbf{43.5}_{\pm 0.65}$
FedLC	$92.2_{\pm 0.11}$	$\underline{31.1}_{\pm 2.44}$	$70.4_{\pm 0.33}$	$\mathbf{53.7}_{\pm 1.18}$	$41.2_{\pm 0.31}$	$33.2_{\pm 0.27}$	$57.5_{\pm 0.38}$	$43.1_{\pm 0.53}$
+ Fixed	$\underline{92.3}_{\pm 0.12}$	$18.2_{\pm 0.06}$	$\mathbf{72.3}_{\pm 0.07}$	$51.0_{\pm 0.93}$	$\underline{45.4}_{\pm 0.18}$	$\underline{34.4}_{\pm 0.04}$	$\underline{61.4}_{\pm 0.04}$	$\underline{43.3}_{\pm 0.99}$
+ Adaptive	$\mathbf{92.6}_{\pm 0.09}$	$\mathbf{32.1}_{\pm 3.85}$	$\underline{72.2}_{\pm 0.17}$	$\underline{52.9}_{\pm 0.87}$	$\mathbf{46.0}_{\pm 0.35}$	$\mathbf{35.7}_{\pm 0.09}$	$\mathbf{61.7}_{\pm 0.26}$	$\mathbf{44.8}_{\pm 0.52}$

† SCAFFOLD follows its canonical setting with zero momentum and uniform client averaging.

6.3 Analysis of Adaptive KD Weight

We next examine how the adaptive KD weight $\lambda_{k,t}$ changes during training. We additionally consider $\beta = 0.3$ as an intermediate level of data heterogeneity. Figure 1 shows that the weight decreases as data heterogeneity becomes stronger, while the overall weight also differs across datasets. In particular, CIFAR-10 receives substantially smaller weights than the other evaluated datasets, consistent with the weaker benefit of strong intermediate KD observed on CIFAR-10. The weight remains small early in training and gradually increases during warm-up, limiting KD before the final classifier becomes sufficiently reliable. After warm-up, it continues to adapt based on local training signals, with clients in the same partition receiving different effective weights. We further examine this behavior across FL algorithms and backbone architectures in Appendix E.



(a) CIFAR-10 (b) CIFAR-100 (c) TinyImageNet (d) ImageNet100-64

Figure 1: Effective KD weight across datasets and data partitions. **Top:** round-wise mean λ, with dashed lines marking the end of warm-up. **Bottom:** client-wise mean λ over the last 30 rounds.

Table 6: Ablation of adaptive KD weighting components. Best results are bolded and second-best results are underlined.

	CIFAR-10		CIFAR-100		TinyImageNet	
Variant	IID	$\beta = 0.1$	IID	$\beta = 0.1$	IID	$\beta = 0.1$
Full	**92.563**	**31.553**	72.150	**54.973**	<u>46.260</u>	<u>36.544</u>
w/o warm-up	92.455	<u>29.702</u>	**72.492**	54.278	**46.290**	**36.622**
w/o teacher reliability	<u>92.480</u>	27.214	71.751	54.061	45.912	36.091
w/o discrepancy	92.234	28.608	<u>72.305</u>	<u>54.683</u>	45.815	36.320

6.4 Ablation Study

We examine the contribution of each factor in the adaptive KD weight by removing the warm-up schedule, teacher reliability, and intermediate prediction discrepancy one at a time. Table 6 reports the results across different datasets and data partitions.

Teacher reliability provides the most consistent contribution. Removing it degrades performance across all evaluated settings, with the effect particularly pronounced on CIFAR-10, where the drop increases from 0.083 points under IID to 4.339 points under $\beta = 0.1$. Since teacher reliability reflects both the final predictor's probability on the ground-truth class and its prediction concentration, these results highlight the importance of adapting the KD strength to teacher reliability.

The discrepancy factor is also beneficial in most settings, especially on CIFAR-10 under $\beta = 0.1$. On this relatively easier task, intermediate predictions can become informative early in training and may already be closely aligned with the final prediction. In such cases, a large KD weight is less necessary, and the discrepancy factor helps reduce excessive intermediate supervision.

The effect of warm-up varies more across settings. Removing it substantially degrades CIFAR-10 under $\beta = 0.1$, while its effect is small elsewhere. This variation is related to its overlap with teacher reliability during early training. When the final predictor remains less reliable, teacher reliability already reduces the KD strength, leaving a smaller role for warm-up. When the final predictor becomes reliable earlier, warm-up can still limit KD according to training progress. Overall, the three factors provide complementary signals for determining when and how strongly intermediate KD should be applied.

7 Conclusion

In this work, we analyzed CE and KD as intermediate supervision in federated learning. CE directly trains auxiliary classifiers toward hard labels, whereas KD guides them using the final classifier's prediction distribution. Because earlier representations may have limited classification capability, direct CE can be restrictive, particularly in FL where each client learns from limited local data. Under such conditions, CE becomes more sensitive to local samples, whereas KD maintains better alignment between gradients computed on local data and the full dataset.

Based on these findings, we use KD for intermediate supervision and propose adaptive weighting that adjusts its strength using local training signals. Across datasets and FL algorithms, adaptive weighting shows stable behavior under different federated settings. Overall, our findings highlight the importance of understanding how different supervision objectives behave under federated local training and using this understanding to adapt KD supervision accordingly.

AI use statement

Generative AI tools were used to assist with manuscript preparation, experimental code modification, and the refinement of research analysis, experimental design, and result interpretation. All AI-assisted content and code were reviewed and verified by the authors, who take full responsibility for the final work.

REFERENCES

Guillaume Alain and Yoshua Bengio. Understanding intermediate layers using linear classifier probes, 2018. URL https://arxiv.org/abs/1610.01644.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Zhipeng Gao, Futeng Feng, Zijia Mo, Zijian Xiong, and Yang Yang. Fedacl: Federated multi-distillation with auxiliary classification layers in iot. In *2022 IEEE 24th Int Conf on High Performance Computing & Communications; 8th Int Conf on Data Science & Systems; 20th Int Conf on Smart City; 8th Int Conf on Dependability in Sensor, Cloud & Big Data Systems & Application (HPCC/DSS/SmartCity/DependSys)*, pp. 1208–1215. IEEE, 2022.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Yutao Hu, Xiaolong Jiang, Xuhui Liu, Xiaoyan Luo, Yao Hu, Xianbin Cao, Baochang Zhang, and Jun Zhang. Hierarchical self-distilled feature learning for fine-grained visual categorization. *IEEE Transactions on Neural Networks and Learning Systems*, 36(3):4005–4018, 2021.

Fatih Ilhan, Gong Su, and Ling Liu. Scalefl: Resource-adaptive federated learning with heterogeneous clients. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 24532–24541. IEEE, 2023.

Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. Scaffold: Stochastic controlled averaging for federated learning. In *International conference on machine learning*, pp. 5132–5143. PMLR, 2020.

Momin A Khan, Yasra Chandio, and Fatima M Anwar. Hydra-fl: Hybrid knowledge distillation for robust and accurate federated learning. *Advances in Neural Information Processing Systems*, 37: 50469–50493, 2024.

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

Yann Le, Xuan Yang, et al. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7):3, 2015.

Chen-Yu Lee, Saining Xie, Patrick Gallagher, Zhengyou Zhang, and Zhuowen Tu. Deeply-supervised nets. In *Artificial intelligence and statistics*, pp. 562–570. Pmlr, 2015.

Gihun Lee, Minchan Jeong, Yongjin Shin, Sangmin Bae, and Se-Young Yun. Preservation of the global knowledge by not-true distillation in federated learning. *Advances in Neural Information Processing Systems*, 35:38461–38474, 2022.

Hunmin Lee and Daehee Seo. Fedlc: Optimizing federated learning in non-iid data via label-wise clustering. *IEEE Access*, 11:42082–42095, 2023.

Royson Lee, Javier Fernandez-Marques, Shell Xu Hu, Da Li, Stefanos Laskaridis, Łukasz Dudziak, Timothy Hospedales, Ferenc Huszár, and Nicholas Donald Lane. Recurrent early exits for federated learning with heterogeneous clients. In *Proceedings of the 41st International Conference on Machine Learning*, pp. 26568–26588, 2024.

Qinbin Li, Bingsheng He, and Dawn Song. Model-contrastive federated learning. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10708–10717. IEEE, 2021.

Shuyuan LI, Boyi LIU, Zimu ZHOU, and Jin DONG. Fedaims: adaptive intermediate supervision for personalized federated learning. *Frontiers of Computer Science*, 21(3):2103601, 2025.

Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *Proceedings of Machine learning and systems*, 2:429–450, 2020.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pp. 1273–1282. Pmlr, 2017.

Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *2018 IEEE/CVF conference on computer vision and pattern recognition*, pp. 4510–4520. Ieee, 2018.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9, 2015.

Linfeng Zhang, Jiebo Song, Anni Gao, Jingwei Chen, Chenglong Bao, and Kaisheng Ma. Be your own teacher: Improve the performance of convolutional neural networks via self distillation. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 3712–3721. IEEE, 2019.

Linfeng Zhang, Xin Chen, Junbo Zhang, Runpei Dong, and Kaisheng Ma. Contrastive deep supervision, 2022. URL https://arxiv.org/abs/2207.05306.

A Additional Theoretical Analysis

This section provides detailed derivations supporting the analytical results in Section 4.

A.1 Hard-Label Variability and Local Gradient Estimation

We also provide the derivation of Eq. 7. Let U_b denote the representation at branch b and $\eta_b(u) = P(Y \mid U_b = u)$ the conditional label distribution given representation u. Conditioning on $U_b = u$, the CE gradient has mean

$$\mathbb{E}\left[g_b^{\mathrm{CE}} \mid U_b = u\right] = p_b(u) - \eta_b(u). \tag{17}$$

Therefore,

$$\begin{aligned}
&\mathbb{E}\left[\left\|g_b^{\mathrm{CE}} - \mathbb{E}[g_b^{\mathrm{CE}} \mid U_b = u]\right\|_2^2 \mid U_b = u\right] \\
&= \mathbb{E}\left[\|e_Y - \eta_b(u)\|_2^2 \mid U_b = u\right] \\
&= 1 - \|\eta_b(u)\|_2^2.
\end{aligned} \tag{18}$$

The last equality follows from $\mathbb{E}[e_Y \mid U_b = u] = \eta_b(u)$ and $\|e_Y\|_2^2 = 1$. This gives the variability of the CE gradient for samples sharing the same representation. To relate this conditional variability to local gradient estimation, we first express the overall variance of the CE gradient using the law of total variance.

$$\begin{aligned}
\operatorname{tr}\operatorname{Var}\left(g_b^{\mathrm{CE}}\right) = \mathbb{E}_{U_b}\left[1 - \|\eta_b(U_b)\|_2^2\right] \\
+ \operatorname{tr}\operatorname{Var}\left(p_b(U_b) - \eta_b(U_b)\right).
\end{aligned} \tag{19}$$

The first term captures the conditional hard-label variability derived above, while the second captures variation in the conditional mean across representations.

We next consider how this per-sample variability affects the gradient estimated by a client from its local data. For client k with n_k IID samples, define the local empirical gradient as

$$\hat{g}_{b,k}^{\mathrm{CE}} = \frac{1}{n_k}\sum_{i=1}^{n_k} g_{b,i}^{\mathrm{CE}}, \tag{20}$$

and let $g_b^{\mathrm{CE},*} = \mathbb{E}[g_b^{\mathrm{CE}}]$ denote the corresponding population gradient. The finite-sample estimation error is then $\epsilon_{b,k}^{\mathrm{CE}} = \hat{g}_{b,k}^{\mathrm{CE}} - g_b^{\mathrm{CE},*}$. Under IID sampling,

$$\mathbb{E}\left[\|\epsilon_{b,k}^{\mathrm{CE}}\|_2^2\right] = \frac{1}{n_k}\operatorname{tr}\operatorname{Var}\left(g_b^{\mathrm{CE}}\right). \tag{21}$$

Thus, the hard-label variability in Eq. 18 contributes to the local gradient estimation error through the overall CE gradient variance. With fewer local samples, the empirical average is more strongly influenced by sample-specific variations in the CE gradient, resulting in a more variable estimate of the population gradient.

A.2 Finite-Sample Error Relation

We next derive Eq. 8 and clarify how the additional final-CE component in branch CE appears under finite local sampling. Let $g_{b,i}^{o}$ denote the per-sample branch gradient for $o \in \{\mathrm{CE}, \mathrm{KD}\}$ and $g_{T,i}^{\mathrm{CE}}$ the corresponding final CE gradient.

For client k with n_k IID samples, define

$$\hat{g}_{b,k}^{o} = \frac{1}{n_k}\sum_{i=1}^{n_k} g_{b,i}^{o}, \qquad \hat{g}_{T,k}^{\mathrm{CE}} = \frac{1}{n_k}\sum_{i=1}^{n_k} g_{T,i}^{\mathrm{CE}}. \tag{22}$$

Let $g_b^{o,*} = \mathbb{E}[g_b^{o}]$ and $g_T^{\mathrm{CE},*} = \mathbb{E}[g_T^{\mathrm{CE}}]$ denote the corresponding population gradients. We define the finite-sample estimation errors as

$$\epsilon_{b,k}^{o} = \hat{g}_{b,k}^{o} - g_b^{o,*}, \qquad \epsilon_{T,k}^{\mathrm{CE}} = \hat{g}_{T,k}^{\mathrm{CE}} - g_T^{\mathrm{CE},*}. \tag{23}$$

From the gradient decomposition in Eq. 6, at $\tau = 1$ we have $g_b^{\mathrm{CE}} = g_b^{\mathrm{KD}} + g_T^{\mathrm{CE}}$ sample-wise. Therefore, the same relation holds for both the finite-sample averages and their population expectations:

$$\hat{g}_{b,k}^{\mathrm{CE}} = \hat{g}_{b,k}^{\mathrm{KD}} + \hat{g}_{T,k}^{\mathrm{CE}},$$
$$g_b^{\mathrm{CE},*} = g_b^{\mathrm{KD},*} + g_T^{\mathrm{CE},*}.$$

Thus,

$$\epsilon_{b,k}^{\mathrm{CE}} = \epsilon_{b,k}^{\mathrm{KD}} + \epsilon_{T,k}^{\mathrm{CE}}. \tag{24}$$

Branch CE therefore includes the final CE estimation error as an additional finite-sample component beyond the KD estimation error. To quantify the contribution of this component, we take the inner product with $\epsilon_{T,k}^{\mathrm{CE}}$:

$$\mathbb{E}\left[\epsilon_{b,k}^{\mathrm{CE}\top}\epsilon_{T,k}^{\mathrm{CE}}\right] - \mathbb{E}\left[\epsilon_{b,k}^{\mathrm{KD}\top}\epsilon_{T,k}^{\mathrm{CE}}\right] = \mathbb{E}\left[\left\|\epsilon_{T,k}^{\mathrm{CE}}\right\|_2^2\right]. \tag{25}$$

Let $\delta_{T,i} = g_{T,i}^{\mathrm{CE}} - g_T^{\mathrm{CE},*}$, so that $\epsilon_{T,k}^{\mathrm{CE}} = \frac{1}{n_k}\sum_{i=1}^{n_k}\delta_{T,i}$ and $\mathbb{E}[\delta_{T,i}] = 0$. Then,

$$\begin{aligned}\mathbb{E}\left[\left\|\epsilon_{T,k}^{\mathrm{CE}}\right\|_2^2\right] &= \frac{1}{n_k^2}\mathbb{E}\left[\left\|\sum_{i=1}^{n_k}\delta_{T,i}\right\|_2^2\right] \\ &= \frac{1}{n_k^2}\left(\sum_{i=1}^{n_k}\mathbb{E}\left[\|\delta_{T,i}\|_2^2\right] + \sum_{i\neq j}\mathbb{E}\left[\delta_{T,i}^\top\delta_{T,j}\right]\right).\end{aligned} \tag{26}$$

Under IID sampling, the cross terms are zero because $\delta_{T,i}$ and $\delta_{T,j}$ are independent and have zero mean. Thus,

$$\begin{aligned}\mathbb{E}\left[\left\|\epsilon_{T,k}^{\mathrm{CE}}\right\|_2^2\right] &= \frac{1}{n_k}\mathbb{E}\left[\|\delta_T\|_2^2\right] \\ &= \frac{1}{n_k}\operatorname{tr}\operatorname{Var}\left(g_T^{\mathrm{CE}}\right).\end{aligned} \tag{27}$$

Substituting this result into Eq. 25 gives Eq. 8. Thus, branch CE contains an additional finite-sample error component aligned with the estimation error of final CE, and the magnitude of this shared component scales inversely with the local sample size n_k.

B Additional Empirical Analysis

B.1 Linear Probe Details

The main analysis reports representative linear probe results under centralized training. Here, we provide the full depth-wise results and additionally examine whether the same pattern holds under federated training. The backbone is frozen, and a new linear classifier is trained independently on the representation after global average pooling at each depth. To avoid directly training the intermediate representations for classification before probing them, we use a backbone trained only with the final classification objective.

As shown in Table 7, the depth-wise pattern is similar across centralized and federated settings: direct classification remains substantially more difficult at earlier intermediate representations than at the final representation. This shows that the limited classification capability observed at earlier depths also persists under federated training.

Table 7: Linear probe accuracy (%) on CIFAR-100 for backbones trained only with the final classification objective. Results are averaged over three seeds.

Setting	B1	B2	B3	Final
Central	5.49	13.19	28.73	76.90
FL (IID)	6.37	12.81	26.61	72.99
FL ($\beta = 0.1$)	6.50	12.74	25.28	67.72

B.2 Client-Wise Gradient Agreement

At communication round t, let $G^o_{k,t}$ denote the branch gradient for objective o computed from the local data of participating client k, and let $G^o_{\text{full},t}$ denote the corresponding gradient computed from the full test set at the same global checkpoint. We define client-wise gradient agreement as

$$A^o_t = \frac{1}{|\mathcal{S}_t|} \sum_{k \in \mathcal{S}_t} \frac{G^{o\top}_{k,t} G^o_{\text{full},t}}{\|G^o_{k,t}\|_2 \|G^o_{\text{full},t}\|_2}. \tag{28}$$

Table 2(a) reports the combined result, where $G^o_{k,t}$ is obtained from the sum of the B1, B2, and B3 branch losses before computing the cosine similarity, reflecting the joint contribution of all auxiliary branches to local optimization. Figure 2 shows the corresponding trajectories for each individual branch. Across most rounds and branches, KD shows higher client-wise agreement with the corresponding full-data gradient than CE under both IID and $\beta = 0.1$.

B.3 Local Class Frequency Metric

The rare–frequent analysis in Section 4.3 measures how client prediction consistency changes with the local frequency of the corresponding class. Let $n_{k,c}$ denote the number of class-c samples on client k, and let $\bar{n}_k = n_k/C$ denote the expected class count under a uniform class distribution. For each class c, we define

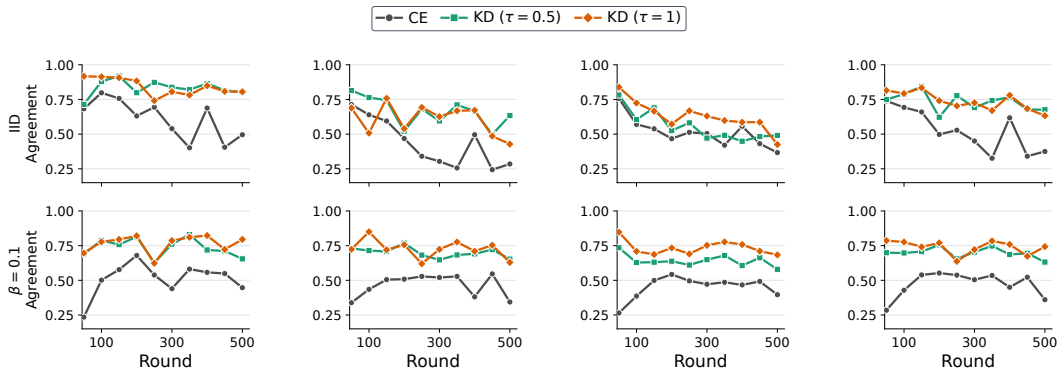
$$\mathcal{K}^{\text{rare}}_c = \{k : n_{k,c} < 0.5\bar{n}_k\}, \qquad \mathcal{K}^{\text{frequent}}_c = \{k : n_{k,c} > 1.5\bar{n}_k\}. \tag{29}$$

All participating client models are evaluated on the same class-balanced reference set from the test data, using up to eight samples per class. For a reference sample x with label c and head h, let $p_{k,h}(x)$ denote the prediction distribution of client k. For frequency group $G \in \{\text{rare}, \text{frequent}\}$, the group mean prediction is

$$\bar{p}^G_h(x) = \frac{1}{|\mathcal{K}^G_c|} \sum_{k \in \mathcal{K}^G_c} p_{k,h}(x). \tag{30}$$

We measure the prediction divergence for each reference sample as

$$D^G_h(x) = \frac{1}{|\mathcal{K}^G_c|} \sum_{k \in \mathcal{K}^G_c} \text{JS}\left(p_{k,h}(x) \,\|\, \bar{p}^G_h(x)\right). \tag{31}$$



(a) B1 (b) B2 (c) B3 (d) Combined

Figure 2: Branch-wise client gradient agreement with the corresponding full-data gradient on CIFAR-100. The top and bottom rows show IID and $\beta = 0.1$. CE and KD are evaluated on models trained with the corresponding objectives and temperatures.

Only samples with at least two clients in the corresponding frequency group are included. We average $D_h^G(x)$ over the valid reference samples and define the rare–frequent gap as

$$\Delta_h = D_h^{\text{rare}} - D_h^{\text{frequent}}. \tag{32}$$

The reported values are averaged over post-local checkpoints at rounds 470, 480, and 490. A larger positive gap indicates greater cross-client prediction divergence when the corresponding class is locally rare.

C Additional Experimental Details

C.1 Implementation Details

We use SGD for local optimization with weight decay 10^{-3}. The momentum is 0.9 for FedAvg, FedProx, MOON, and FedLC, and 0 for SCAFFOLD. The learning rate follows a round-wise exponential decay, $\eta_t = \eta_0(0.998)^{t-1}$, where η_0 denotes the initial learning rate. We train CIFAR-10 and CIFAR-100 for 500 communication rounds with $\eta_0 = 0.1$, and TinyImageNet and ImageNet100-64 for 100 rounds with $\eta_0 = 0.01$. We use a training batch size of 64 and set the test batch size to 512 for the default experiments and 256 for ResNet50.

For FedProx, we set the proximal coefficient to $\mu = 0.01$. For MOON, we use a contrastive coefficient of $\mu = 0.01$ and a contrastive temperature of 0.5. For FedLC, we use a calibration coefficient of 1.0 and apply the client-specific margin to the logits used for the supervised and KD objectives. The adaptive weighting factors are computed from the uncalibrated predictions. For the main ResNet18 performance evaluations, we use seeds $\{0, 1, 2\}$. The effective KD weight analysis, additional local training configurations, and ablation study use seed 0.

C.2 Auxiliary Branch Implementation

For ResNet18, auxiliary branches B1, B2, and B3 are attached after `layer1`, `layer2`, and `layer3`, respectively. Each branch consists of a convolutional bottleneck followed by global average pooling and a linear classifier. The final classifier follows `layer4` and serves as the teacher for all intermediate branches. During local training, the final classifier is optimized using hard-label CE, while the auxiliary branches are supervised only through KD. The losses from all auxiliary branches are summed during local training, while evaluation discards the auxiliary branches and measures test accuracy using only the final classifier.

For ResNet50, auxiliary branches are likewise attached after `layer1`, `layer2`, and `layer3`, using the same convolutional bottleneck design as ResNet18. For MobileNetV2, they are attached after stages 2, 4, and 6, with each branch projecting the intermediate feature to 1280 channels before global average pooling and linear classification. For CCT, auxiliary branches are attached after transformer blocks 2, 4, and 5, with layer normalization, attention-based sequence pooling, and a linear classifier. For ConvNeXt, we use a lightweight four-stage variant with channel dimensions $\{32, 64, 128, 256\}$. Auxiliary branches are attached after the first three stages, with a 1×1 projection followed by layer normalization, GELU, global average pooling, and a linear classifier.

C.3 KD Weighting Configuration

For the fixed-weight baseline, we use $\lambda = 0.3$, selected from a preliminary weight sweep on CIFAR-100 and kept fixed across all subsequent settings without setting-specific tuning. For adaptive weighting, we set $\lambda_{\max} = 1$ and use a linear warm-up over the first 50% of communication rounds. For the prediction-concentration adjustment in the teacher reliability, we set the threshold to $\kappa = 0.85$, the temperature to $T_s = 0.05$, and the power to 2.

D PERFORMANCE ACROSS MODEL ARCHITECTURES

To examine the effect of adaptive weighting across different backbone architectures, we additionally evaluate ResNet50 (He et al., 2016) and MobileNetV2 (Sandler et al., 2018) under FedAvg using the same four datasets and both IID and $\beta = 0.1$ data partitions. For ResNet50, auxiliary branches are attached after `layer1`, `layer2`, and `layer3`, following the same convolutional bottleneck design as ResNet18. For MobileNetV2, auxiliary branches are attached after stages 2, 4, and 6, with each branch projecting the intermediate feature to 1280 channels before global average pooling and linear classification. The remaining training settings and adaptive weighting strategy follow the main experiments.

Table 8 shows that the effect of intermediate KD varies across backbone architectures. With ResNet50, adaptive weighting improves over Fixed in all evaluated settings and provides substantial gains on TinyImageNet and ImageNet100-64. Although adaptive weighting does not always surpass Plain, it recovers much of the degradation caused by Fixed on CIFAR-10 and CIFAR-100 under $\beta = 0.1$.

A similar pattern appears with MobileNetV2. Adaptive weighting improves over Fixed in all four non-IID settings, while the differences under IID are smaller and mixed. It does not consistently outperform Plain, particularly on TinyImageNet and ImageNet100-64, showing that intermediate KD itself is not uniformly beneficial across architectures. Overall, these results indicate that adaptive weighting more consistently mitigates the degradation from using a single fixed KD weight, particularly under non-IID data.

Table 8: Global test accuracy (%) of Plain, Fixed, and Adaptive across additional model architectures under FedAvg using seed 0. Best and second-best results are bolded and underlined, respectively.

	CIFAR-10		CIFAR-100		TinyImageNet		ImageNet100-64	
Model	IID	$\beta = 0.1$	IID	$\beta = 0.1$	IID	$\beta = 0.1$	IID	$\beta = 0.1$
ResNet50	<u>91.25</u>	**22.62**	70.17	**52.07**	37.10	31.05	51.00	38.01
+ Fixed	90.89	16.14	<u>72.23</u>	49.37	<u>47.82</u>	<u>31.47</u>	<u>63.26</u>	<u>40.17</u>
+ Adaptive	**91.89**	<u>21.28</u>	**72.26**	<u>51.92</u>	**48.20**	**33.17**	**63.63**	**43.31**
MobileNetV2	86.76	<u>14.60</u>	60.70	**35.65**	**31.31**	<u>21.30</u>	**45.31**	**30.15**
+ Fixed	**87.77**	11.79	**62.07**	32.76	29.73	20.43	43.96	26.85
+ Adaptive	<u>87.19</u>	**15.77**	<u>61.89</u>	<u>35.59</u>	<u>30.79</u>	**21.37**	<u>44.75</u>	<u>29.38</u>

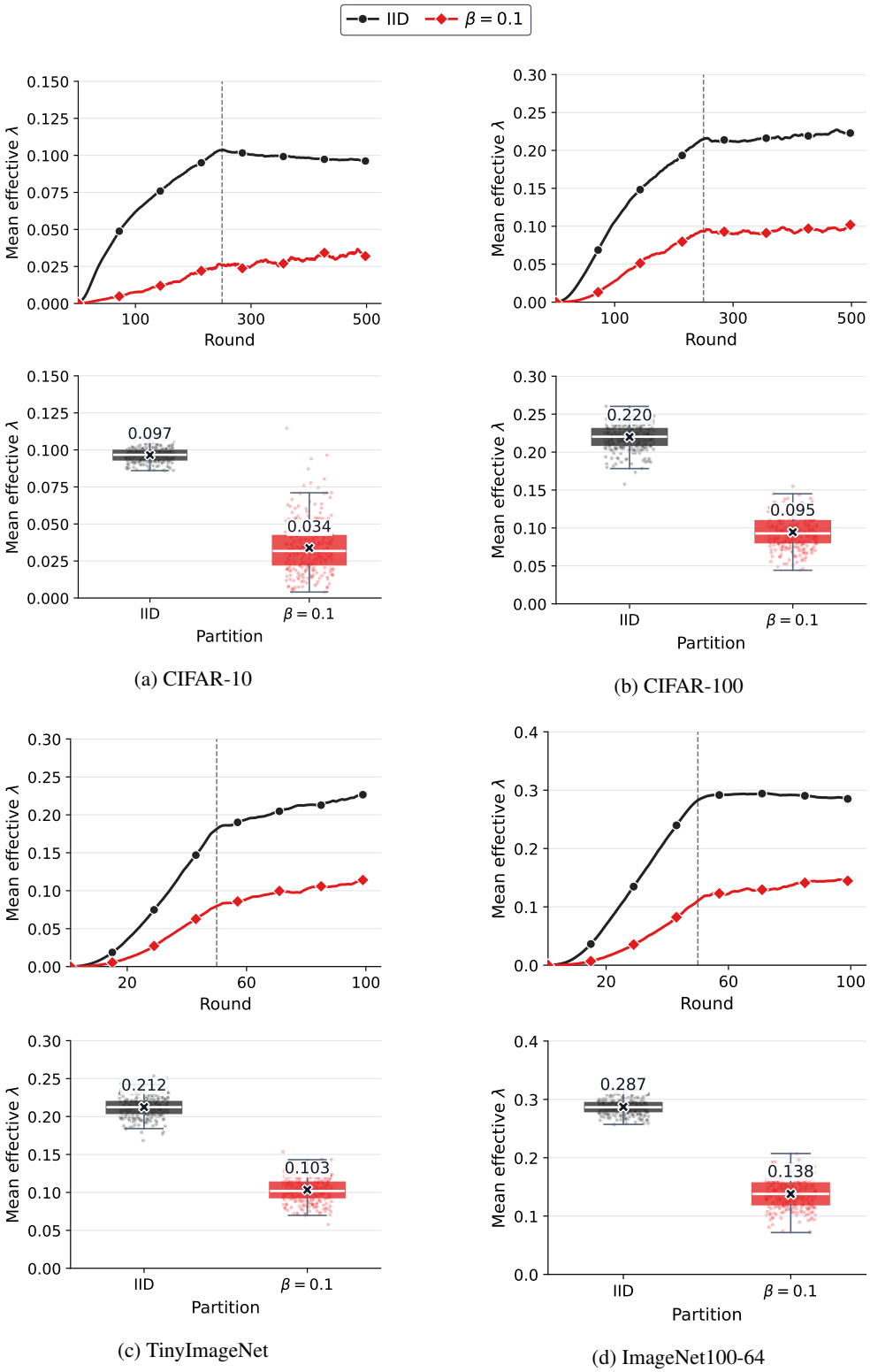
E Additional Effective KD Weight Analysis

To complement the effective KD weight analysis in Section 6.3, we provide additional results across FL algorithms and backbone architectures. The default FedAvg with ResNet18 is omitted because it is already presented in Figure 1.

We first examine the effective KD weight under FedProx, MOON, and SCAFFOLD. For each FL algorithm, we report the round-wise effective KD weight and the client-wise distribution across datasets and data partitions.

E.1 Across FL Algorithms

We further examine the effective KD weight under FedProx, MOON, SCAFFOLD, and FedLC. As shown in Figures 3–6, the effective KD weight generally remains lower under $\beta = 0.1$ than under IID across datasets, consistent with the FedAvg results in Figure 1. This indicates that the adaptive weighting behavior is maintained across different FL algorithms.



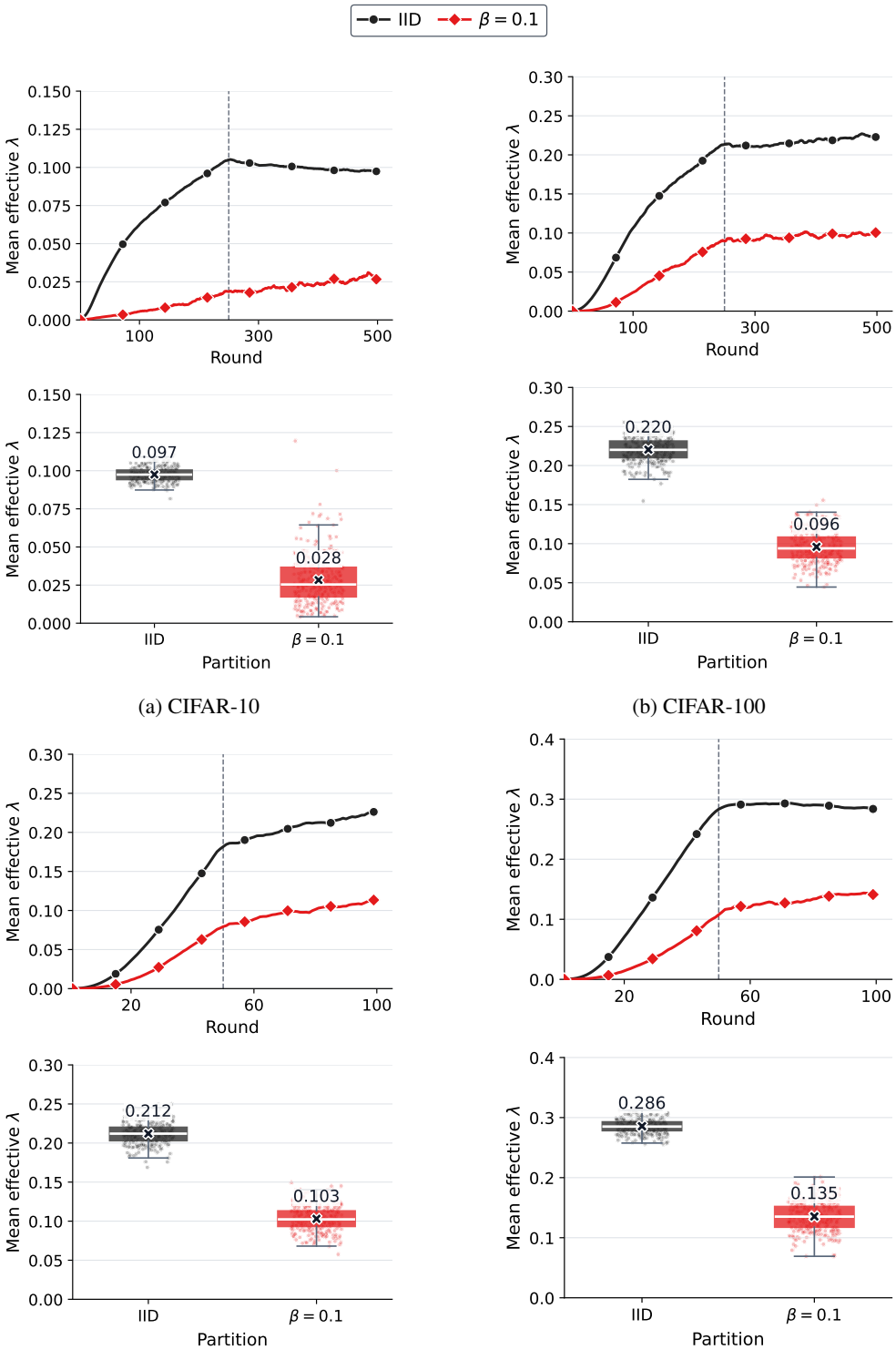
(a) CIFAR-10

(b) CIFAR-100

(c) TinyImageNet

(d) ImageNet100-64

Figure 3: Effective KD weight under FedProx across datasets and data partitions. For each dataset, the upper panel shows round-wise values and the lower panel shows client-wise distributions.



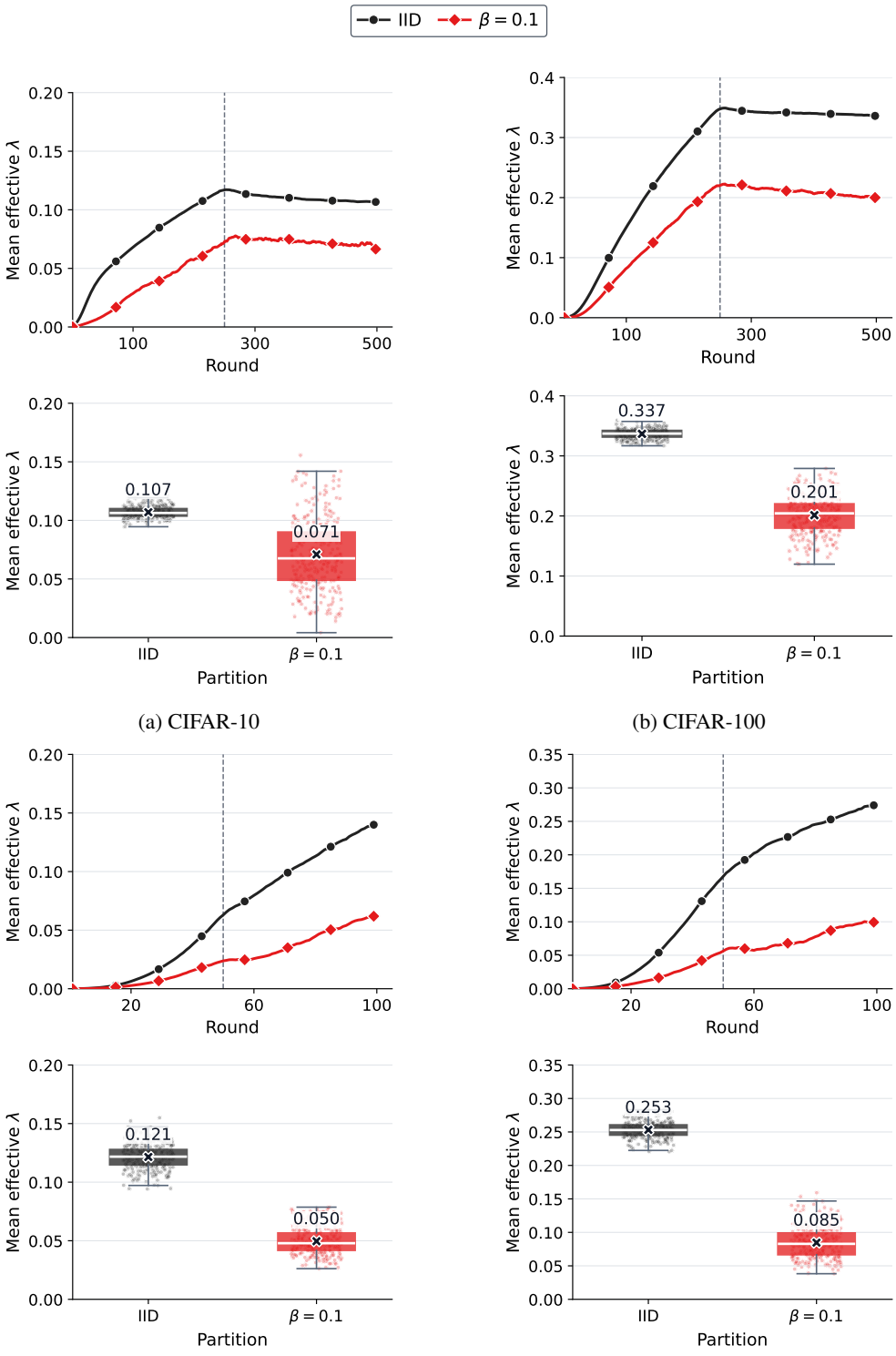
(a) CIFAR-10

(b) CIFAR-100

(c) TinyImageNet

(d) ImageNet100-64

Figure 4: Effective KD weight under MOON across datasets and data partitions. For each dataset, the upper panel shows round-wise values and the lower panel shows client-wise distributions.



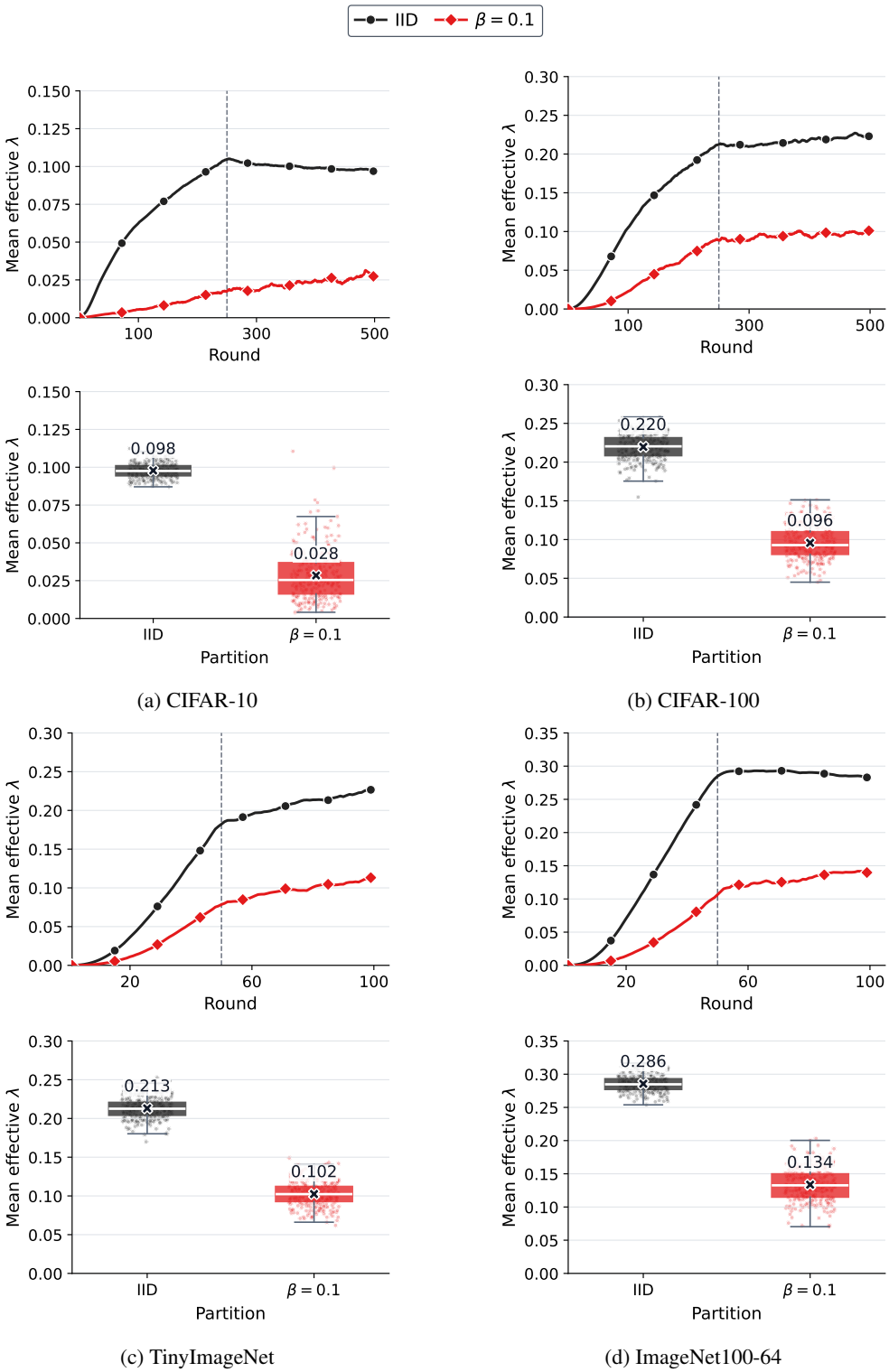
(a) CIFAR-10

(b) CIFAR-100

(c) TinyImageNet

(d) ImageNet100-64

Figure 5: Effective KD weight under SCAFFOLD across datasets and data partitions. For each dataset, the upper panel shows round-wise values and the lower panel shows client-wise distributions.



(a) CIFAR-10

(b) CIFAR-100

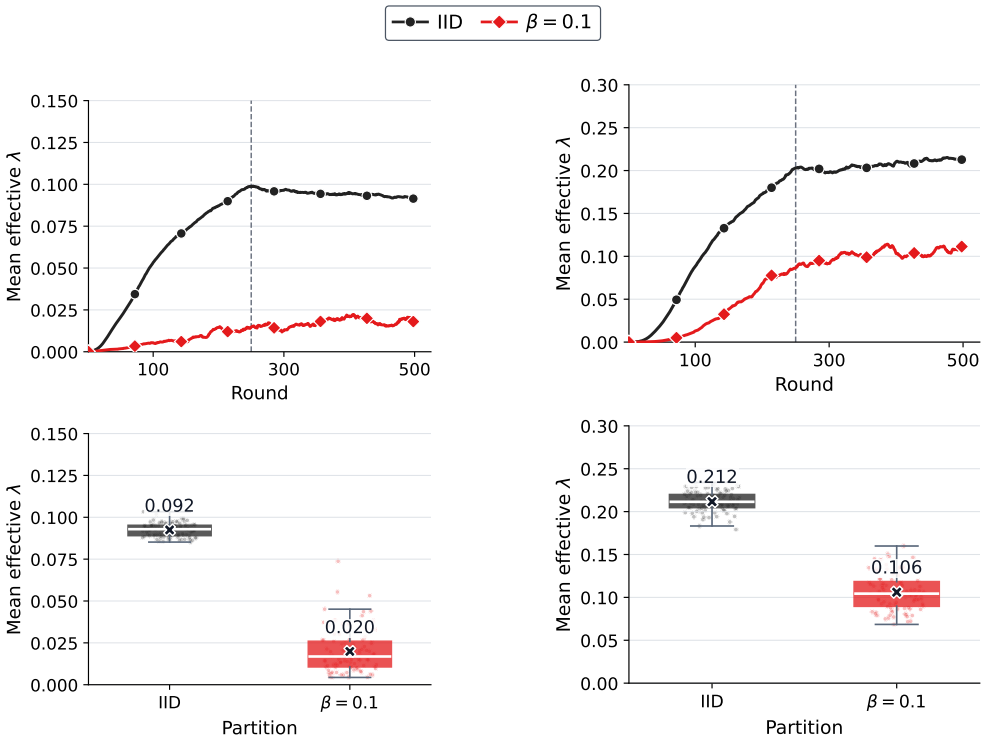
(c) TinyImageNet

(d) ImageNet100-64

Figure 6: Effective KD weight under FedLC across datasets and data partitions. For each dataset, the upper panel shows round-wise values and the lower panel shows client-wise distributions.

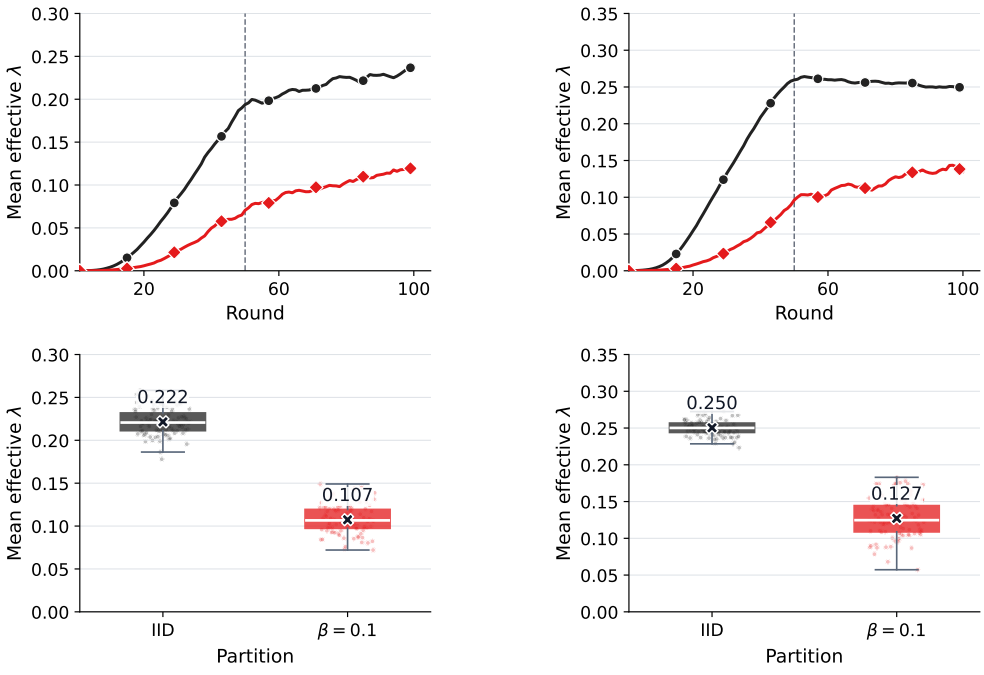
E.2 Across Backbone Architectures

We further examine the effective KD weight with ResNet50 and MobileNetV2 under FedAvg. Figures 7 and 8 show a similar tendency, with smaller effective KD weights under $\beta = 0.1$ than under IID across datasets. The magnitude of the effective weight varies with the backbone, indicating that the adaptive strategy adjusts the KD strength differently across model architectures.



(a) CIFAR-10

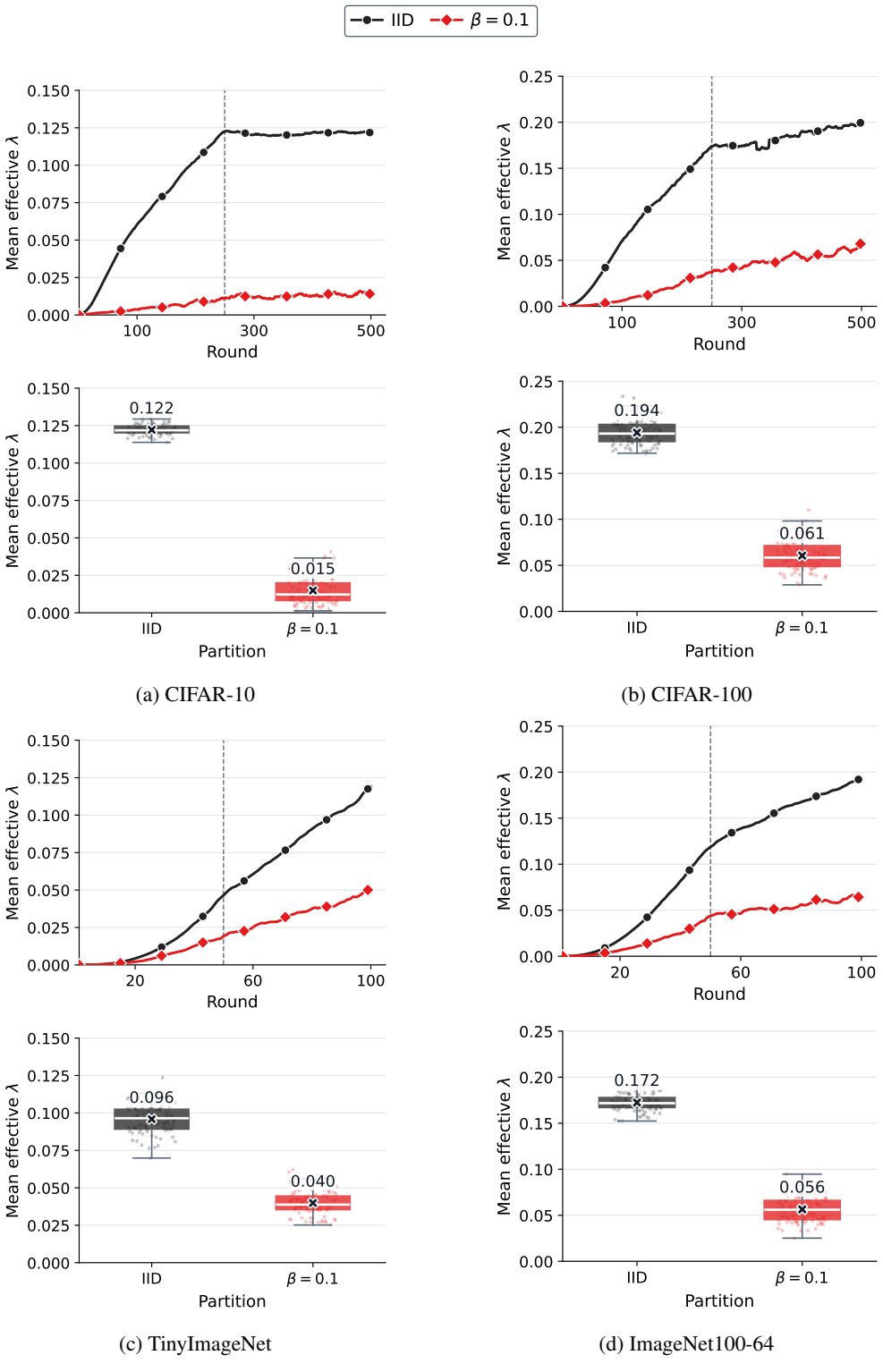
(b) CIFAR-100



(c) TinyImageNet

(d) ImageNet100-64

Figure 7: Effective KD weight under ResNet50 across datasets and data partitions. For each dataset, the upper panel shows round-wise values and the lower panel shows client-wise distributions.



(a) CIFAR-10

(b) CIFAR-100

(c) TinyImageNet

(d) ImageNet100-64

Figure 8: Effective KD weight under MobileNetV2 across datasets and data partitions. For each dataset, the upper panel shows round-wise values and the lower panel shows client-wise distributions.

F Computational and Communication Overhead

We additionally examine the overhead introduced by auxiliary branch supervision. Since Fixed and Adaptive use the same auxiliary-branch architecture, they incur the same parameter and communication overhead. Adaptive weighting introduces additional computation only during local training for estimating the reliability and discrepancy factors.

Parameter and communication overhead. Table 9 reports the parameter and communication costs for the architectures evaluated in this work. The auxiliary branches increase the parameter count by 18.03% for ResNet18, 128.28% for ResNet50, and 30.16% for MobileNetV2. Under our current implementation, clients communicate the complete model state, including the auxiliary branches. The resulting communication increase therefore closely follows the increase in model size.

Table 9: Parameter and communication overhead introduced by auxiliary branches. Communication is measured per round with 10 participating clients, where each client downloads and uploads one complete model state.

Architecture	Base Params	+ Branches	Param. Inc.	Comm. Inc.
ResNet18	11.22M	13.24M	18.03%	18.09%
ResNet50	23.71M	54.11M	128.28%	128.12%
MobileNetV2	2.35M	3.06M	30.16%	30.05%

Local training cost. We further measure the empirical local-training and aggregation cost using the default ResNet18 backbone on CIFAR-100 with FedAvg under IID partitioning. All measurements use five local epochs, 10% client participation, batch size 64, and seed 0. Round time includes local training and server aggregation but excludes global evaluation. As shown in Table 10, intermediate KD increases local training time and memory usage, while adaptive weighting introduces additional computation for estimating its local weighting factors.

Table 10: Empirical local-training and aggregation cost on CIFAR-100 with ResNet18. Runtime and memory ratios are reported relative to Plain.

Method	Sec./Round	Runtime	GPU Mem.	Memory
Plain	10.82	1.00×	0.64 GiB	1.00×
Fixed	13.67	1.26×	0.79 GiB	1.24×
Adaptive	18.60	1.72×	0.79 GiB	1.24×

Adaptive requires additional local computation over Fixed because the teacher reliability and intermediate prediction discrepancy factors are evaluated during training. This computation does not introduce additional model parameters or communication beyond those already required by the auxiliary branches.

Inference cost. The auxiliary branches are used only during local training and are not required for inference. The final prediction path follows the underlying backbone and final classifier. Thus, after removing the auxiliary branches when exporting the trained model, the deployed model has the same parameter count and inference architecture as the corresponding base model.