Federated Personalization of Early-Exit Networks

Boyi Liu
City University of Hong Kong Shenzhen Research Institute
Shenzhen, China
Department of Data Science, City University of Hong Kong
Hong Kong, China
SKLCCSE, Beihang University
Beijing, China
boy.liu@my.cityu.edu.hk

Zimu Zhou
City University of Hong Kong Shenzhen Research Institute
Shenzhen, China
Department of Data Science, City University of Hong Kong
Hong Kong, China
zimuzhou@cityu.edu.hk

Cheng Fang
Department of Data Science, City University of Hong Kong
Hong Kong, China
cheng.fang@my.cityu.edu.hk

Yongxin Tong
SKLCCSE, Beihang University
Beijing, China
yxtong@buaa.edu.cn

Abstract

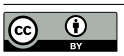
Personalized Federated Learning (PFL) excels at tailoring client-specific models, which is particularly critical for decentralized and heterogeneous data environments, yet existing methods produce static models with a fixed tradeoff between accuracy and efficiency. This inherent static nature limits their ability to adapt to inference demands that vary with context and resource availability, posing a challenge for real-world deployment. Early-exit networks (EENs), which enable adaptive inference via intermediate classifiers, offer a promising solution. However, integrating EENs into PFL introduces two intertwined conflicts: client-wise heterogeneity across clients and depth-wise interference arising from conflicting exit objectives. Prior studies fail to resolve both conflicts simultaneously, leading to suboptimal performance. In this paper, we propose X-FED, a novel Conflict-Aware Cross-Client Federated Exit Distillation framework that jointly addresses both client- and depth-wise conflicts while extending PFL to early-exit networks. At its core, X-FED employs a progressive, depth-prioritized student coordination mechanism that mitigates interference among shallow and deep exits while enabling effective personalized knowledge transfer across clients. Furthermore, we introduce a client-decoupled formulation that reduces communication overhead with theoretical soundness. Extensive evaluations on various datasets show that compared to state-of-the-art PFL and PFL-EE methods, X-FED achieves higher accuracy while reducing inference costs by 30.79%–46.86%.

CCS Concepts

• **Computing methodologies** → **Learning paradigms**.

Keywords

Personalized Federated Learning; Early-Exit Networks



CIKM '26, Rome, Italy.

ACM ISBN 979-8-4007-2539-5/2026/11
https://doi.org/10.1145/3799682.3840572

ACM Reference Format:
Boyi Liu, Zimu Zhou, Cheng Fang, and Yongxin Tong. 2026. Federated Personalization of Early-Exit Networks. In *Proceedings of the 35th ACM International Conference on Information and Knowledge Management (CIKM '26), November 7–11, 2026, Rome, Italy*. ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3799682.3840572

1 Introduction

Personalized Federated Learning (PFL) [44] excels at leveraging the *decentralized*, *heterogeneous* data generated from IoT devices for intelligent applications such as smart healthcare [35], traffic prediction [55] and personalized assistants [29]. It enables collaborative model training across a network of IoT devices (clients) while tailoring models to each client's unique data distribution. PFL often achieves higher accuracy than both generic federated learning (GFL) of a single global model [33, 46, 47, 51] and local training on limited data, as it effectively integrates shared knowledge with client-specific insights.

Despite their improved accuracy, these personalized models are inherently *static*, with a fixed tradeoff between accuracy and efficiency. IoT applications, however, often process continuous data streams in dynamic environments where inference requirements change with context, workload, or resource availability [13]. For instance, an image classifier in autonomous driving must prioritize ultra-low-latency predictions to identify immediate hazards, such as nearby pedestrians or vehicles, while ensuring accurate recognition of distant objects or under complex lighting. Dynamic models, like early-exit networks (EENs) [41, 45], address this challenge by attaching intermediate classifiers (*i.e.*, early exits) to the backbone, enabling inference to terminate at shallower exits when appropriate. This reduces latency while preserving accuracy for simpler inputs, making EENs ideal for IoT applications with fluctuating demands.

Can we effectively train both *personalized* and *dynamic* models in federated learning (see Fig. 1)? Consider n clients, each training personalized EENs with m exits. This setup requires optimizing mn objectives simultaneously, which introduces challenges beyond the *scale*. Two types of exit *conflicts* complicate the training process. The first is *client-wise heterogeneity*, where variations in local data distributions necessitate careful decisions on what to share across clients versus what to personalize [44]. The second is *depth-wise*

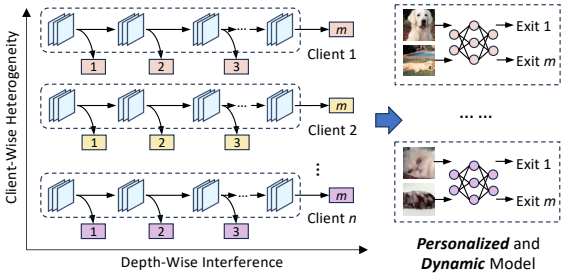


Figure 1: Motivation for personalized federated learning of early-exit networks (PFL-EE).

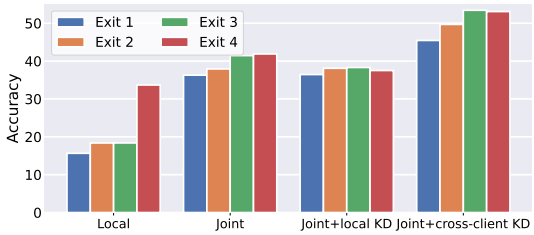


Figure 2: An empirical comparison of post-PFL exit training (local), direct extension of PFL to EENs (joint), existing GFL-EE (joint + local KD), and our solution (joint + cross-client KD) to train personalized 4-exit EENs for 100 clients on CIFAR-100 under Dir(0.3).

interference, caused by conflicting backpropagation signals between shallow and deep exits during joint training [23]. These conflicts create intricate interdependence among exits, making personalized federated training of EENs (PFL-EE) distinct from both personalized federated learning of single-exit networks (PFL) [1, 8, 17, 34, 52] and generic federated learning of EENs (GFL-EE) [18, 20, 24].

Existing solutions fail to address both conflicts simultaneously. One naive approach is applying PFL to the last exit (and the backbone) [1], followed by local training of shallower exits [45]. However, this method underperforms because shallow exits also require knowledge across clients to compensate for the sparse local datasets (*local* vs. *joint* in Fig. 2). Another option is to adopt recent federated training methods for global EENs [18], which employ local knowledge distillation (KD) [38] to supervise the training of shallow exits. Yet, under severe data heterogeneity, which is intrinsic in PFL, local KD can degrade accuracy, performing worse than simple joint training (*joint + local KD* vs. *joint* in Fig. 2). Both strategies fail because they implicitly treat exits as analogous to clients, reducing PFL-EE to standard PFL with more clients, thereby overlooking the distinctions between client- and depth-induced exit conflicts.

In this paper, we present X-FED, a Conflict-Aware Cross-Client Federated Exit Distillation framework for personalized federated learning of EENs. X-FED effectively resolves both client- and depth-wise conflicts, enabling shallow exits to distill personalized knowledge from the last exits across all clients. At its core is a *progressive, depth-prioritized* coordination strategy for student exits. Specifically, X-FED gradually incorporates exits along the depth dimension rather than the client dimension, encouraging shallow exits to align with their corresponding teachers first, thereby reducing conflicts during subsequent distillation. The incremental inclusion also maintains manageable conflict levels throughout the training. Following student selection, X-FED matches teacher exits to student exits based on inferred *data similarity*, as is standard in PFL, to facilitate personalized knowledge transfer across clients. Furthermore, to avoid the communication overhead of cross-client KD, X-FED introduces a *client-decoupled* formulation that achieves theoretical equivalence to cross-client KD while incurring a communication cost comparable to standard FedAvg [33].

Our main contributions are summarized as follows.

- To our knowledge, this is the first work on personalized federated learning of early-exit networks (PFL-EE). It advances PFL from static to dynamic models capable of high-fidelity and low-latency inference in evolving environments.
- We propose X-FED, a unified distillation-based framework that addresses the dual challenges of client-wise heterogeneity and depth-wise interference in PFL-EE. It harmonizes exit training through progressive, depth-prioritized student coordination and introduces a client-decoupled formulation to eliminate the communication overhead of direct cross-client distillation.
- Evaluations on four datasets show that X-FED outperforms state-of-art PFL [1, 5, 8, 17, 26, 34, 48, 52] and GFL-EE [18, 20] baselines in accuracy, while reducing the inference cost by up to 30.79-46.86%.

2 Related Work

Personalized Federated Learning. PFL [44] trains personalized models rather than a single global model to handle non-IID data across clients. Each model holds a client-specific classifier [1, 6, 8, 34, 36] or is a different sub-model of the global model [3, 32, 56] to retrain personalized parameters. For example, FedRep [8] divides the model into a global feature extractor and a local classifier. Only the feature extractor is aggregated at the server. pFedGate [3] employs personalized masks to generalize different sub-models for clients. Personalization can also be enforced via client clustering [43], model regularization [26], knowledge distillation [54], meta-learning [10], etc.

Existing PFL research trains personalized, static (single-exit) models, whereas we learn both personalized and dynamic models exemplified by early-exit networks. For simplicity, we adopt the classifier-based strategy [1, 6, 8, 34, 36] for personalization.

Early-Exit Networks. EENs [41] are dynamic neural networks [13] that condition their computational depths on inputs. They insert intermediate exits to the model, allowing early termination of inference on easy samples, thus improving inference efficiency. An EEN can be optimized in model architectures [15, 45, 50], exit policies [16, 19, 42], and training strategies [11, 38, 53]. We employ the classic multi-branch architecture with a simple confidence-based exit policy [45], and mainly focus on EEN training.

The exits in EENs are often jointly trained to improve accuracy [13, 15], which can be further enhanced by knowledge distillation (KD) [38]. Originally, EEN training assumes centralized settings. Yet recent efforts [18, 20, 24, 40] have explored federated learning of a global EEN with clients of heterogeneous resource constraints

[9]. We investigate an orthogonal problem that trains personalized EENs with clients holding non-IID data. Our solution is inspired by these studies [18, 20, 24] that utilize KD to harmonize exit training. However, their KD is confronted within each local EEN. We show the necessity and propose a novel formulation of cross-client KD for federated learning of personalized EENs.

3 Problem Statement

Personalized Federated Learning (PFL). Assume n clients $\{c_1, c_2, \ldots, c_n\}$ holding local datasets $\{D_1, D_2, \ldots, D_n\}$ which are often not independent and identically distributed (non-IID). PFL trains n personalized, *single-exit* models $\{\theta_1, \ldots, \theta_n\}$ in a standard federated setup [33], by optimizing the following objective [44]:

$$\min \sum_{i=1}^{n} p_i F_i(\theta_i; D_i) \tag{1}$$

where $p_i = \frac{|D_i|}{\sum_{i=1}^{n} |D_i|}$, and F_i is the local objective of client i.

Early-Exit Networks (EENs). An EEN [45] is a dynamic neural network that adjusts its computational depths at inference time based on inputs. It consists of a backbone ϕ and m exits $\{h_1, \ldots, h_m\}$. The m exits are often jointly trained on dataset D to mitigate cross-exit interference and boost accuracy [23, 41].

$$\min \sum_{j=1}^{m} w_j F(\phi, h_j; D) \tag{2}$$

where $\forall j$, the exit-wise weight $w_j = 1/m$, assuming each exit is equally important for inference [12, 15, 20, 24, 38].

Personalized Federated Learning of EENs (PFL-EE). We train for each client i a personalized, *multi-exit* model $\theta_i = (\phi; h_{i1}, \ldots, h_{im})$ in the federated manner, where ϕ is a backbone *shared* among the n clients, and $\{h_{i1}, \ldots, h_{im}\}$ are m personalized exits for client i, by optimizing the following objective:

$$\min \sum_{i=1}^{n} p_i \sum_{j=1}^{m} w_j F_i(\phi, h_{ij}; D_i) \tag{3}$$

It differs from prior studies as follows.

- Traditional **PFL** [1, 8, 17, 34, 52] optimizes n *client-wise* objectives, while we simultaneously optimize mn objectives in both the *client* and *exit* dimensions.
- EENs have been used to align features among local models of different depths in FL [18, 20, 24], which we term as **GFL-EE**. It is primarily designed for IID data and learns a *global* multi-exit model whereas we train n *personalized* multi-exit models.

Challenges. PFL-EE is more challenging than standard PFL and GFL-EE not only because of the increased *scale of objectives* but also the need for a unified approach to handle two *types of conflicts*. *(i)* There is *interference* among *exits* due to the interplay of multiple backpropagation signals in joint exit training [23]. *(ii)* There is *data heterogeneity* among *clients* that demands careful balance between similarity and complementarity for effective federated training [49].

Scope. We focus on the training of EENs. We attach an exit per block to construct a classic multi-branch EEN architecture and apply a simple threshold-based exit policy [45]. For simplicity, FedAvg [33] is used for model aggregation.

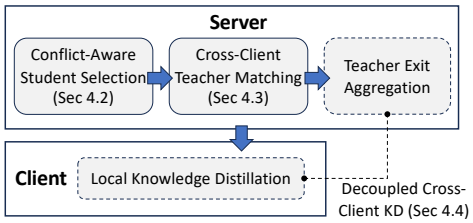


Figure 3: Overview of X-FED.

4 Method

4.1 X-FED Overview

X-FED is a Conflict-Aware Cross-Client Federated Exit Distillation framework for effective personalized federated learning of EENs.

Principles. X-FED is inspired by [18] that exploits the last exit as an anchor to harmonize the training of shallow exits. This is achieved via knowledge distillation (KD) from the last exit to shallow ones *within* the local EEN at each client. We extend the idea to PFL-EE by allowing distillation from the last exits *across* all clients to shallow exits *across* all clients.

Specifically, let $\theta_{ij} = (\phi, h_{ij})$ be the sub-model that corresponds to backbone ϕ and exit h_{ij} of client i at depth j. For a student model θ, our *cross-client* KD leverages all teacher models $\{\theta_{1m}, \ldots, \theta_{nm}\}$ to improve θ via the following KD loss:

$$\mathcal{L}_{XKD}(\theta; x) = \sum_{i=1}^{n} k_i D_{KL}\left(\theta_{im}(x) \| \theta(x)\right) \tag{4}$$

where D_{KL} is the KL divergence [22], and k_i is the distillation weight of teacher i in *cross-client* KD, satisfying $\sum_{i=1}^{n} k_i = 1$.

- Cross-client KD is necessary for PFL-EE. Naively integrating local KD into PFL as GFL-EE [18, 20, 24] even yields a lower accuracy than PFL without KD (see Fig. 2).
- Cross-client KD necessitates new student-teacher coordination mechanisms due to the increased scale and type of conflicts as explained in Sec. 3.

X-FED coordinates the distillation by *prioritizing conflict management among exits over clients*. *(i)* In the exit dimension, we fix the last exits from all clients as candidate teachers, and progressively include shallow exits as students, to mitigate the conflict exerted to the shared backbone. *(ii)* In the client dimension, each student exit learns from the last exit beyond its own client, and it is matched with teacher exits from other clients based on their (estimated) data similarity for effective distillation.

Workflow. X-FED integrates the above principles into a typical PFL pipeline (see Fig. 3). In each round, client i trains the shared backbone θ and its personalized exits $\{h_{ij}\}$ on local dataset D_i as:

$$\min \sum_{j \in \mathbb{S}_i} w_j \left(\mathcal{L}_{CE}(\theta_{ij}; D_i) + \lambda \mathcal{L}_{XKD}(\theta_{ij}; D_i)\right) \tag{5}$$

where the first term is the standard cross-entropy loss, the second term is the cross-client KD loss defined in Eq. (4), λ is the hyperparameter controlling the proportion of KL loss, and $\mathbb{S}_i$ is the set of exits included for training at client i.

The key novelty of X-FED lies in the *effective conflict management* when optimizing Eq. (5) via standard gradient descent.

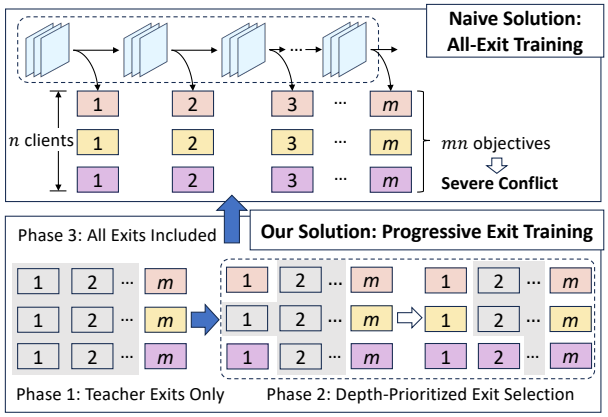


Figure 4: Conflict-aware student selection.

- We progressively expand $\mathbb{S}_i$ for client i via a *conflict-aware student selection* scheme (Sec. 4.2) to control exit-wise interference during joint exit training and distillation.
- We adaptively set the distillation weights $\mathbf{k} = [k_1, \ldots, k_n]$ based on the inferred similarities between student and teach exits via a *cross-client teacher matching* strategy (Sec. 4.3).
- Naive implementation of the cross-client KD loss, *i.e.*, second term of Eq. (5), needs exit parameters of all n clients, which incurs excessive communication and violates privacy. Hence, we propose an equivalent *decoupled cross-client distillation* formulation (Sec. 4.4) for practical deployment.

After local training, the server collects and aggregates the shared backbone from the clients as in other classifier-based PFL [1, 6, 8, 34, 36]. When the training converges, the shared backbone and the personalized exits are assembled as the final EEN for deployment to each client (see Fig. 1).

4.2 Conflict-Aware Student Selection

This module determines the set of exits $\mathbb{S}_i$ for local training and distillation at every client i ($1 \le i \le n$) in round t ($1 \le t \le T$), with T denoting the total number of rounds. As mentioned in Sec. 4.1, the last exits of all clients are employed as teachers. Hence, $\mathbb{S}_i$ is initialized as $\{m\}$ for all client i and round t. We design a *two-tier* selection scheme that gradually involves the remaining $(m-1)n$ exits as students for distillation (see Fig. 4).

Let the proportion of student exits in round t be $R(t)$, which corresponds to $(m-1)nR(t)$ student exits. We empirically set $R(t) = \min(\frac{2t}{T}, 1)$. Other options are evaluated in Sec. 5.4.3. We gradually include more exits for training over rounds to avoid the substantial conflicts to optimize mn exits all at once. The concrete exit selection operates as follows.

Depth-Prioritized Exit Selection. In round t, we first include the $\lfloor (m-1)R(t) \rfloor$ *shallowest* exits as students for every client i. This step is essential for two reasons.

- If the $(m-1)n$ exits are treated as $(m-1)n$ clients rather than prioritize the depth, similarity-based client selection in PFL [4] tend to pick exits from a single client, since the data heterogeneity across clients is larger than that among exits of the same client. Yet this would cause the shared backbone to bias towards the selected client.
- Including shallower exits and aligning them with the teacher first will mitigate the conflicts in later distillation for deeper exits, since it encourages deeper exits to only focus on the knowledge not well-learned by shallow ones [53].

Similarity-Aware Exit Selection. For the remaining $K = (m-1)nR(t) - n\lfloor (m-1)R(t) \rfloor$ exits, we pick them from depth $\lfloor (m-1)R(t) \rfloor + 1$ as standard similarity-based client selection in PFL. We focus on client similarity because we have included the shallow exits of all exits, and thus sufficient client complementarity [49]. For two clients i and j, we measure the similarity of their exits l as $\delta_{ij}^l = \min(0, \cos(\theta_{il}, \theta_{jl}))$. The gradient-based proxy is common to measure data similarity among clients in FL [30, 31, 43].

Then we iteratively remove the least similar exits from all exits $\mathcal{E}^l$ at depth l. Specifically, we initialize the exit set $\mathcal{E} = \mathcal{E}^l$. In each iteration, we greedily remove exit e via the following criterion,

$$e = \arg\min_{e \in \mathcal{E}} \sum_{i \in \mathcal{E}} \delta_{ie}^l \tag{6}$$

until the number of remaining exits $|\mathcal{E}| = K$. We use the similarity of final exits $\cos(\theta_{im}, \theta_{jm})$ to approximate that of shallow exits $\cos(\theta_{il}, \theta_{jl})$ to reduce communication cost.

4.3 Cross-Client Teacher Matching

This module assigns distillation weights $\mathbf{k} = [k_1, \ldots, k_n]$ to the n teacher exits for the student exits $\{h_{ij}\}_{j=1}^{m-1}$ of client i. As shown in Sec. 1, learning from other clients is critical to boost the accuracy of shallow exits, yet the teachers should be properly selected to avoid negative knowledge transfer due to inter-client data heterogeneity. We maximize the knowledge sources by associating each student exit to all the n teachers, and only distill matched knowledge via similarity-based teacher weighing. As mentioned in Sec. 4.1, only the final exits in each EEN serve as teachers.

Similarity-Based Teacher Weighting. We assign the weights $\mathbf{k} = [k_1, \ldots, k_n]$ based on the inferred similarity in training data distributions between the student and teacher exits. A higher similarity results in a higher weight. We apply the same gradient-based proxy as Sec. 4.2 and formulate the following objective:

$$\arg\max_{\mathbf{k}} \sum_{i=1}^{n} k_i \cos(\theta_{im}, \theta) - \mu \sum_{i=1}^{n} \|k_i - \frac{1}{n}\|^2 \;\; s.t. \sum_{i=1}^{n} k_i = 1 \tag{7}$$

where θ and θ_{im} are the student model and the teacher model, respectively. The second term is a barrier regularization of $\mathbf{k}$ controlled by hyperparameter μ [54].

Note that Eq. (7) approximately minimizes the transfer risk of KD, *i.e.*, student's risk after KD. This is because the transfer risk bound for KD is related to the cosine angle of the teacher model and the student data space [37], which correlates to $\cos(\theta_{im}, \theta)$.

Intra-Client Weight Approximation. Eq. (7) demands solving $\mathbf{k}$ for all student exits $\{h_{ij}\}_{j=1}^{m-1}$ of client i, which can be time-consuming. Since the exit-wise heterogeneity is less notable than the client-wise heterogeneity in PFL [2], we enforce all student exits at client i to share the same distillation weights $\mathbf{k}$, which accelerates the preparation for distillation.

Specifically, we utilize the similarity between the teacher exit and the deepest student exit of client i to approximate $\cos(\theta_{im}, \theta)$.

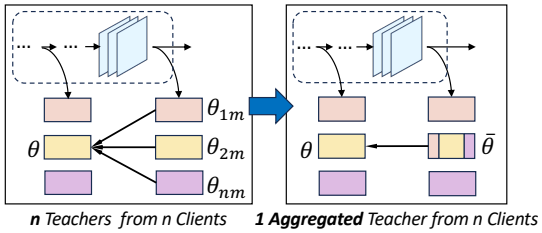


Figure 5: Decomposition of cross-client KD.

Accordingly, we reduce Eq. (7) from exit-wise to client-wise:

$$\arg\max_{\mathbf{k}} \mathbf{k}^\top c - \mu(\mathbf{k}I - \frac{1}{n}I)^\top(\mathbf{k}I - \frac{1}{n}I) \quad s.t.\ \ \mathbf{1}^\top\mathbf{k} = 1; \mathbf{k} \geq 0 \tag{8}$$

where $c = [\cos(h_{1m}, h_{sm}), \ldots, \cos(h_{nm}, h_{sm})]$ for any student exit s of client i, and we solve Eq. (8) via quadratic program.

4.4 Decoupled Cross-Client Distillation

As mentioned in Sec. 4.1, the cross-client KD formulated as Eq. (4) (second term in Eq. (5)) is impractical at the clients since it needs transmitting teacher exits of all other clients, which imposes high communication cost and violates privacy [51]. One alternative is to shift the KD from clients to the server. Yet it requires either a public dataset [25, 54] or a data generator [58] at the server, which is less favorable than client-side KD. Instead, we propose an equivalent cross-client KD formulation without the need for other clients' raw model parameters at each client (see Fig. 5).

Client-Decoupled KD. We reformulate the cross-client KD in Eq. (4) as a two-step pipeline.

- *Teacher Exit Aggregation at Server*. Since all clients share the same teacher exits (*i.e.*, the last exit of the n clients) but differ in the distillation weights $\mathbf{k}$, we can aggregate the n teacher exits at the server:

$$\bar{h} = \sum_{i=1}^{n} k_i h_{im} \tag{9}$$

  where the distillation weights $\mathbf{k}$ are still calculated as Sec. 4.3.
- *Local KD at Clients*. The aggregated teacher exit $\bar{h}$ and the shared backbone ϕ are sent to each client i as the teacher model $\bar{\theta} = (\phi, \bar{h})$ for distillation with the student model θ:

$$\mathcal{L}_{XKD}(\theta; x) = D_{KL}(\bar{\theta}(x) \| \theta(x)) \tag{10}$$

  Replacing the KD loss Eq. (5) by Eq. (10), client i performs local training and distillation as follows:

$$\mathcal{L}_i = \sum_{j \in \mathbb{S}_i} w_j \mathcal{L}_{CE}(\theta_{ij}; D_i) + \lambda \sum_{j \in \mathbb{S}_i} D_{KL}(\bar{\theta}(D_i) \| \theta(D_i)) \tag{11}$$

  where we empirically set $\lambda = 1$.

Equivalence to Cross-Client KD. The two-step KD is equivalent to the objective in Eq. (4) via the following proposition.

Proposition 1 (Equivalence of KD). *Given an input sample x, the cross-client KD in Eq. (4) is equivalent to knowledge distillation from an aggregated teacher model $\bar{\theta} = (\phi, \bar{h})$ where $\bar{h} = \sum_{i=1}^{n} k_i h_{im}$.*

$$\theta^* = \arg\min_{\theta} \sum_{i=1}^{n} k_i D_{KL}(\theta_{im}(x) \| \theta(x)) = \arg\min_{\theta} D_{KL}(\bar{\theta}(x) \| \theta(x)) \tag{12}$$

That is, θ^* obtained through cross-client KD can be derived by minimizing the KL divergence between $\bar{\theta}_m$ and θ. The proof is deferred to Appendix A.1.

Algorithm 1: X-FED workflow

Input: Clients' model parameters $\theta_1, \ldots, \theta_n$
Output: Personalized models $\theta_1^t, \ldots, \theta_n^t$

1 **for** *round t* **do**
2 // Server Execute
3 sample clients set $\mathbb{C}$ for this round
4 select exits $\{\mathbb{S}_i | i \in \mathbb{C}\}$ as Sec. 4.2
5 **for** *client i* $\in \mathbb{C}$ **do**
6 solve $\mathbf{k}$ via Eq. (8)
7 teacher exit $\bar{h}_i \leftarrow$ aggregate via Eq. (9)
8 // Client Execute
9 **for** *client i* $\in \mathbb{C}$ **do**
10 receive $\phi_i^t, \bar{h}_i$ from server
11 update local final exit $h_{im} \leftarrow \bar{h}_i$
12 update local backbone $\phi_i^t \leftarrow \phi^t$
13 calculate cross-client KD loss $\mathcal{L}_{XKD}$ via Eq. (10)
14 calculate overall training objective $\mathcal{L}_i$ via Eq. (11)
15 local training $\theta_i^{t+1} \leftarrow \theta_i^t - \eta\nabla\mathcal{L}_i(\theta_i^t; D_i)$
16 upload ϕ_i^t, h_{im} to server
17 aggregate backbone $\phi^{t+1} \leftarrow \sum_{i \in \mathbb{C}} p_i \phi_i^t$

4.5 Putting It Together

Algorithm 1 shows the workflow of X-FED. In each round, the server randomly samples a client set $\mathbb{C}$ to participate in training (line 3). For each sampled client, we further decide the student exits via conflict-aware student selection (line 4), and assemble a personalized teacher exit $\bar{h}_i$ for each student exit (line 5-7). On receiving the shared backbone ϕ^t and the teacher exit $\bar{h}_i$, client i updates its local model (line 10-15). Then the backbone ϕ_i^t and final exit h_{im} are uploaded to server for aggregation (line 16). The aggregation of backbone aligns with FedAvg [33] (line 17). The final exits are kept at the server for future cross-client teacher matching.

Convergence. Under mild assumptions as [28], X-FED converges to a data heterogeneity related constant after sufficient training round T (details and proofs in Appendix A.2).

Communication Cost. In X-FED, only the backbone and the last exit are transferred between the server and clients. All shallow exits are kept locally. The data exchanged per round is the same as FedAvg [33], *i.e.*, transmission of the full model. Compared with classifier-based PFL [1, 6, 8, 34, 36], X-FED transfers slightly more data, *i.e.*, the teacher exits. Yet the extra communication overhead is negligible. For example, our experiments show that the communication cost of X-FED is merely 0.45% higher than classifier-based PFL [1, 6, 8, 34, 36] with ResNet-18 [14].

Computational Cost. X-FED adds extra server-side computation but matches GFL-EE baselines (*e.g.* ScaleFL [18]) in client-side overhead, for they share the same number of local KD objectives. The server-side computational cost of X-FED mainly comes from (i) conflict-aware student selection with $O(s^2n^2)$ (see Sec. 4.2), (ii) cross-client teacher matching with $O(s^3n^3)$ (see Sec. 4.3), and (iii) teacher exit aggregation with $O(sn)$ (see Sec. 4.4), where s is the sampling rate, and n is number of clients. Teacher matching with cubic level complexity dominates among the three. However, since it operates only over *sampled clients* ($sn \ll n$), the actual cost is small. Sec. 5.3.3 further provides an empirical study on the computational latency of teacher matching.

5 Experiments

5.1 Experimental Setup

Baselines. We mainly compare X-FED with representative GFL and PFL methods, including Local training, FedAvg [33], FedProx [27], FedPer [1], FedRep [8], FedBABU [34], FedAMP [17], pFedGraph [52], FedRoD [5], Ditto [26], and FedPAC [48]. For a fair comparison, we extend them to EENs. We also include GFL-EE methods dedicated to train a global EEN as baselines, including ScaleFL [18] and DepthFL [20].

Datasets and Models. We experiment with CIFAR-10/CIFAR-100 [21], TinyImageNet [7], and AgNews [57]. The data distribution adheres to a Dirichlet distribution, quantified by a hyperparameter α. We set $\alpha = 0.3$ and 0.1 following [56]. For CIFAR-10, we apply a 3-layer ConvNet. For CIFAR-100 and TinyImageNet, we apply ResNet18. For AgNews, we apply a 4-block Transformer.

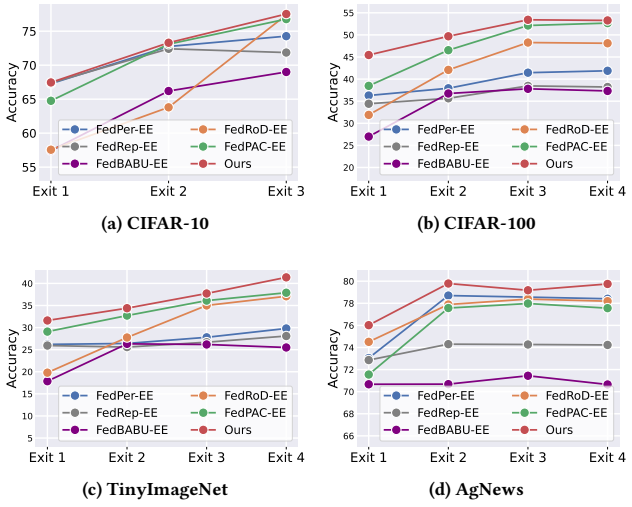
Configurations. We simulate a federation of 100 clients, with a sampling rate of 0.1. The communication round is set to $T = 300$ for CIFAR-10, CIFAR-100, TinyImageNet, and $T = 100$ for AgNews. The local epoch is set to $E = 5$ for CIFAR-10, CIFAR-100, TinyImageNet and $E = 1$ for AgNews. The batch size is fixed to 64. The learning rate is set to 0.1, with a decay of 0.99 per round.

Metrics. We assess the methods with three performance metrics: *(i) Accuracy*: The model accuracy on local datasets measured at the terminated exit during inference. *(ii) Averaged Accuracy*: The average accuracy of all exits. *(iii) Inference Efficiency*: The average number of Multiply-Accumulate (MAC) operations per input sample.

5.2 Main Results

Averaged Accuracy. Table. 1 reports the averaged accuracy of all exits across seven non-IID settings on four datasets. The GFL-EE methods with local KD (*e.g.* DepthFL and ScaleFL) do not perform well under severe non-IID settings. This validates the observations in Fig. 2. Most PFL-EE methods outperform GFL-EE schemes, except FedAMP-EE and pFedGraph-EE. This is because these two baselines adopt personalized aggregation, which becomes ineffective with a low client sampling rate. X-FED improves the averaged accuracy over other PFL-EE methods by up to 9.57%, 21.29%, 12.19% and 16.48% for each dataset.

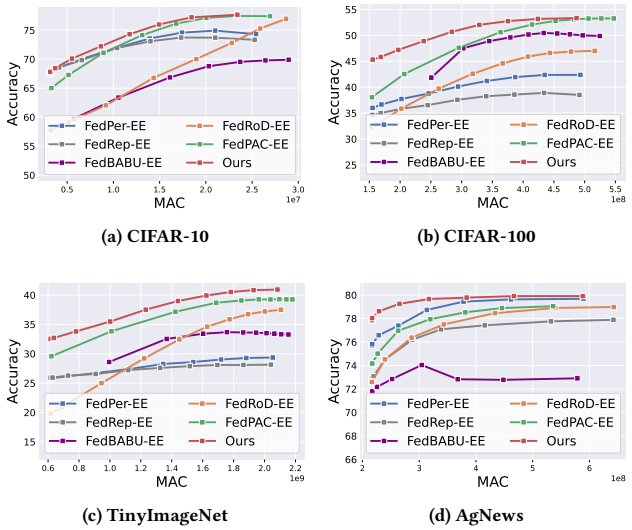
Exit-Wise Accuracy. Fig. 6 zooms into the per-exit accuracy of the top five performing baselines in Table. 1, *i.e.*, FedPer-EE, FedRep-EE, FedBABU-EE, FedRoD-EE, FedPAC-EE. We show the results under Dir(0.3). X-FED outperforms the baselines *in all exits*. For example, on CIFAR-100, X-FED yields accuracy gains of 6.98-18.46%, 3.15-14.04%, 1.29-18.46% and 0.61-16.11% in each exit.



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

(c) TinyImageNet (d) AgNews

Figure 6: Per-exit accuracy of X-FED and five best-performing baselines from Table. 1.



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

(c) TinyImageNet (d) AgNews

Figure 7: Accuracy-efficiency tradeoff of X-FED and five best-performing baselines from Table. 1.

Accuracy-Efficiency Tradeoff. We apply a naive exit policy [15] where the model terminates when the output confidence, *i.e.*, the maximum of the softmax output of an exit, exceeds a threshold ϵ. We compare X-FED with the same five baselines under the exit policy with different thresholds ϵ to understand how they tradeoff between inference accuracy and efficiency.

Fig. 7 shows the measured accuracy and MAC. On all four datasets, X-FED achieves a *higher accuracy* at *fewer MACs*. Specifically, on CIFAR-10, for an accuracy of 70%, X-FED saves 0.2-5.5× MACs.

Table 1: Overall performance of federated training of EENs. *Averaged accuracy* of all exits are measured.

Type	Method	CIFAR-10		CIFAR-100		TinyImageNet		AgNews
		Dir(0.3)	Dir(0.1)	Dir(0.3)	Dir(0.1)	Dir(0.3)	Dir(0.1)	Dir(0.3)
N/A	Local-EE	$68.43_{\pm 12.52}$	$81.92_{\pm 13.08}$	$28.55_{\pm 5.76}$	$48.79_{\pm 9.74}$	$22.96_{\pm 12.91}$	$38.28_{\pm 14.69}$	$71.91_{\pm 21.08}$
GFL-EE	FedAvg-EE [33]	$49.46_{\pm 14.85}$	$42.20_{\pm 17.11}$	$31.53_{\pm 5.56}$	$16.98_{\pm 5.94}$	$19.55_{\pm 4.53}$	$11.58_{\pm 5.96}$	$25.11_{\pm 33.16}$
	FedProx-EE [27]	$62.05_{\pm 6.78}$	$43.71_{\pm 19.43}$	$30.88_{\pm 6.18}$	$16.42_{\pm 5.61}$	$16.80_{\pm 6.87}$	$12.83_{\pm 4.82}$	$27.67_{\pm 32.94}$
	ScaleFL [18]	$38.69_{\pm 7.71}$	$26.75_{\pm 10.54}$	$23.13_{\pm 4.56}$	$9.83_{\pm 4.54}$	$11.19_{\pm 3.28}$	$7.87_{\pm 3.58}$	$23.69_{\pm 31.96}$
	DepthFL [20]	$10.23_{\pm 17.99}$	$9.87_{\pm 23.71}$	$20.44_{\pm 7.52}$	$12.56_{\pm 5.72}$	$0.90_{\pm 1.49}$	$0.55_{\pm 0.71}$	$24.75_{\pm 34.06}$
PFL-EE	FedPer-EE [1]	$71.39_{\pm 11.42}$	$82.58_{\pm 11.77}$	$39.39_{\pm 5.66}$	$59.06_{\pm 8.20}$	$27.55_{\pm 11.92}$	$43.78_{\pm 13.24}$	$77.18_{\pm 15.56}$
	FedRep-EE [8]	$70.56_{\pm 11.49}$	$82.71_{\pm 11.38}$	$36.68_{\pm 5.77}$	$59.26_{\pm 8.37}$	$26.58_{\pm 11.96}$	$44.03_{\pm 13.33}$	$73.92_{\pm 19.50}$
	FedBABU-EE [34]	$64.21_{\pm 13.51}$	$75.23_{\pm 18.46}$	$34.71_{\pm 9.07}$	$53.45_{\pm 11.36}$	$23.97_{\pm 13.70}$	$37.63_{\pm 16.38}$	$70.86_{\pm 22.17}$
	FedAMP-EE [17]	$47.25_{\pm 20.09}$	$64.75_{\pm 25.76}$	$24.98_{\pm 4.99}$	$14.24_{\pm 9.50}$	$16.04_{\pm 15.54}$	$26.05_{\pm 17.21}$	$62.08_{\pm 30.13}$
	pFedGraph-EE [52]	$50.19_{\pm 18.57}$	$67.66_{\pm 24.10}$	$10.52_{\pm 5.20}$	$22.07_{\pm 8.75}$	$9.20_{\pm 17.92}$	$13.73_{\pm 20.91}$	$71.27_{\pm 22.04}$
	FedRoD-EE [5]	$66.36_{\pm 12.37}$	$81.51_{\pm 13.35}$	$42.59_{\pm 6.63}$	$62.19_{\pm 8.35}$	$28.91_{\pm 12.09}$	$45.09_{\pm 12.85}$	$77.24_{\pm 15.64}$
	Ditto-EE [26]	$69.79_{\pm 12.66}$	$82.60_{\pm 13.15}$	$29.12_{\pm 5.71}$	$49.19_{\pm 9.56}$	$24.08_{\pm 12.68}$	$40.23_{\pm 14.76}$	$72.83_{\pm 20.10}$
	FedPAC-EE [48]	$71.60_{\pm 10.80}$	$82.73_{\pm 13.70}$	$47.45_{\pm 5.65}$	$61.92_{\pm 7.89}$	$33.96_{\pm 11.21}$	$46.47_{\pm 12.65}$	$76.17_{\pm 17.70}$
Ours	X-FED	$\mathbf{72.67}_{\pm 10.71}$	$\mathbf{83.68}_{\pm 13.55}$	$\mathbf{50.41}_{\pm 5.47}$	$\mathbf{62.83}_{\pm 8.64}$	$\mathbf{36.27}_{\pm 10.86}$	$\mathbf{48.22}_{\pm 12.59}$	$\mathbf{78.69}_{\pm 14.63}$

On AgNews, for an accuracy of 78%, X-FED saves 0.2-2.8× MACs. On CIFAR-100, given a budget of 3.5×10^8 MACs, the accuracy of X-FED is 2.5-14.5% higher. On TinyImageNet, with 1.4×10^9 MACs, the accuracy of X-FED is 2-11% higher.

5.3 Micro-Benchmarks

5.3.1 Benefit of EENs. Complementary to the main results, we directly compare X-FED with FL baselines that train a single-exit model to highlight the benefits of EENs. We exclude DepthFL and ScaleFL in this experiment since they are designed for EENs. For X-FED, we set $\epsilon = 0.8$ for CIFAR-10 and AgNews, and $\epsilon = 0.6$ for CIFAR-100 and TinyImageNet. We use different thresholds ϵ since datasets with fewer classes tend to reach higher confidence levels more easily.

Table. 2 shows the accuracy X-FED improves the accuracy over the PFL baselines by 0.11-35.30%, 4.88-40.61%, 0.99-28.92% and 6.73-16.11%. Due to the adaptive inference of EENs, the accuracy gains come with fewer MACs. The averaged MAC per sample of the PFL baselines is 30.39M, 557.94M, 2231.64M and 811.57M on the four tasks, because they produce the same static model. For X-FED, its averaged MAC per sample is 16.15M, 300.45M, 1545.69M, 446.58M, which reduces the inference cost by 46.15%, 44.73%, 30.74% and 48.49% for each dataset.

5.3.2 Necessity of Cross-Client KD. Following the motivating experiment in Sec. 1, we provide a more detailed counterexample to justify the need for cross-client KD in PFL of EENs. Specifically, we perform local KD on the PFL-EE baselines.

Table. 3 summarizes the per-exit accuracy with and without local KD on CIFAR-100 under Dir(0.3). With local KD, the accuracy of shallow exits (*e.g.* Exit-1) is slightly improved. Yet the accuracy of deeper exits degrade. It implies that with severe data heterogeneity, local KD cannot effectively train personalized EENs. The result aligns with our motivation of cross-client KD.

Table 2: Accuracy of PFL methods.

Type	Method	CIFAR10	CIFAR100	TinyImageNet	AgNews
N/A	Local	$68.67_{\pm 13.06}$	$22.82_{\pm 5.37}$	$22.07_{\pm 13.35}$	$72.49_{\pm 20.85}$
GFL	FedAvg [33]	$63.20_{\pm 9.66}$	$31.60_{\pm 4.94}$	$19.21_{\pm 5.80}$	$27.66_{\pm 32.95}$
	FedProx [27]	$66.21_{\pm 6.98}$	$30.80_{\pm 4.83}$	$19.90_{\pm 5.41}$	$26.69_{\pm 31.26}$
PFL	FedPer [1]	$73.98_{\pm 10.82}$	$35.14_{\pm 6.10}$	$26.87_{\pm 11.90}$	$72.55_{\pm 20.66}$
	FedRep [8]	$74.53_{\pm 10.70}$	$30.62_{\pm 5.94}$	$22.72_{\pm 12.73}$	$71.79_{\pm 21.58}$
	FedBABU [34]	$75.20_{\pm 11.32}$	$34.14_{\pm 6.72}$	$28.14_{\pm 13.27}$	$67.44_{\pm 25.78}$
	FedAMP [17]	$41.31_{\pm 22.16}$	$26.08_{\pm 5.79}$	$20.61_{\pm 13.35}$	$63.17_{\pm 28.90}$
	pFedGraph [52]	$44.39_{\pm 22.19}$	$10.60_{\pm 5.21}$	$10.71_{\pm 17.50}$	$63.28_{\pm 33.51}$
	FedRoD [5]	$75.33_{\pm 11.50}$	$35.90_{\pm 6.67}$	$27.18_{\pm 12.85}$	$71.79_{\pm 21.58}$
	Ditto [26]	$69.43_{\pm 13.11}$	$23.96_{\pm 5.59}$	$19.91_{\pm 13.37}$	$71.03_{\pm 22.27}$
	FedPAC [48]	$76.50_{\pm 9.51}$	$46.33_{\pm 6.25}$	$38.64_{\pm 10.83}$	$72.14_{\pm 21.25}$
Ours	X-FED	$\mathbf{76.61}_{\pm 9.16}$	$\mathbf{51.21}_{\pm 5.61}$	$\mathbf{39.63}_{\pm 10.33}$	$\mathbf{79.28}_{\pm 14.95}$

Table 3: Impact of local KD on PFL-EE baselines.

Methods	KD	Accuracy			
		Exit 1	Exit 2	Exit 3	Exit 4
FedPer-EE [1]	✗	$36.29_{\pm 5.94}$	$37.92_{\pm 6.36}$	$41.45_{\pm 6.02}$	$41.88_{\pm 5.79}$
	✓	$36.94_{\pm 5.81}$	$38.79_{\pm 5.62}$	$39.15_{\pm 6.04}$	$38.33_{\pm 5.94}$
FedRep-EE [8]	✗	$34.42_{\pm 6.44}$	$35.65_{\pm 6.16}$	$38.46_{\pm 6.00}$	$38.21_{\pm 5.95}$
	✓	$34.96_{\pm 6.30}$	$36.14_{\pm 6.39}$	$39.07_{\pm 6.27}$	$38.27_{\pm 5.93}$
FedBABU-EE [34]	✗	$26.99_{\pm 7.29}$	$36.74_{\pm 9.97}$	$37.80_{\pm 10.66}$	$37.32_{\pm 10.71}$
	✓	$31.02_{\pm 7.11}$	$32.56_{\pm 8.17}$	$32.37_{\pm 8.71}$	$30.82_{\pm 8.30}$
FedRoD-EE [5]	✗	$31.90_{\pm 7.30}$	$42.08_{\pm 7.46}$	$48.27_{\pm 8.23}$	$48.11_{\pm 7.15}$
	✓	$31.68_{\pm 7.70}$	$41.80_{\pm 7.96}$	$47.62_{\pm 8.73}$	$46.82_{\pm 6.85}$
FedPAC-EE [48]	✗	$38.47_{\pm 6.18}$	$46.54_{\pm 6.15}$	$52.13_{\pm 5.92}$	$52.67_{\pm 6.16}$
	✓	$38.49_{\pm 6.34}$	$46.10_{\pm 6.40}$	$51.65_{\pm 6.25}$	$52.10_{\pm 6.35}$

5.3.3 Computational Cost of Teacher Matching. We measured the teacher matching latency under different sampling scales (see Table. 4).

Empirically, teacher matching in our experimental setup (*i.e.*, 100 clients with a sampling rate of 0.1) takes 0.02 seconds. The results show that even with 200 sampled clients (*e.g.* 10,000 clients

Table 4: Computational time of teacher matching under different numbers of sampled clients.

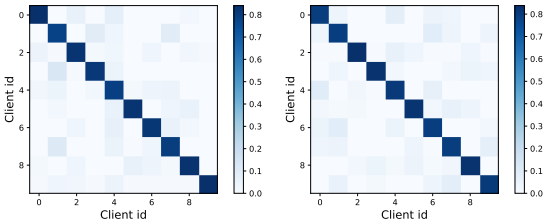
Sampled Clients	10	20	50	100	200
Time (s)	0.02	0.02	0.05	0.48	1.43

Table 5: Contributions of individual components. KD: naive local knowledge distillation; TM: cross-client teacher matching; SS: conflict-aware student selection.

Dataset	Options			Accuracy			
	KD	TM	SS	Exit 1	Exit 2	Exit 3	Exit 4
CIFAR-100	✗	✗	✗	$36.29_{\pm 5.94}$	$37.92_{\pm 6.36}$	$41.45_{\pm 6.02}$	$41.88_{\pm 5.79}$
	✓	✗	✗	$36.45_{\pm 5.65}$	$38.10_{\pm 5.67}$	$38.31_{\pm 5.63}$	$37.51_{\pm 5.66}$
	✓	✓	✗	$42.06_{\pm 6.14}$	$45.64_{\pm 6.26}$	$48.21_{\pm 5.54}$	$51.91_{\pm 5.18}$
	✓	✓	✓	$45.45_{\pm 6.06}$	$49.69_{\pm 5.96}$	$53.42_{\pm 5.96}$	$53.08_{\pm 6.26}$
Tiny-ImageNet	✗	✗	✗	$26.20_{\pm 12.34}$	$26.41_{\pm 12.08}$	$27.80_{\pm 11.98}$	$29.80_{\pm 11.65}$
	✓	✗	✗	$25.27_{\pm 12.12}$	$25.75_{\pm 11.97}$	$26.35_{\pm 12.09}$	$26.52_{\pm 12.22}$
	✓	✓	✗	$30.44_{\pm 11.95}$	$32.16_{\pm 11.58}$	$34.85_{\pm 10.98}$	$38.28_{\pm 10.45}$
	✓	✓	✓	$31.61_{\pm 11.71}$	$34.39_{\pm 11.42}$	$37.72_{\pm 10.73}$	$41.37_{\pm 10.09}$

with a sampling rate of 0.02), teacher matching takes only 1.43s, confirming negligible server overhead.

5.3.4 Visualization of Cross-client KD Weight. We visualize the cross-client KD weight on CIFAR-100 dataset under both Dir(0.3) and Dir(0.1) in Fig. 8. For all clients, the weight assigned to the client itself is approximately 0.8. This indicates that, despite cross-client KD improves the performance as demonstrated in Sec. 5.4, it still primarily depends on the client's own teacher exit.



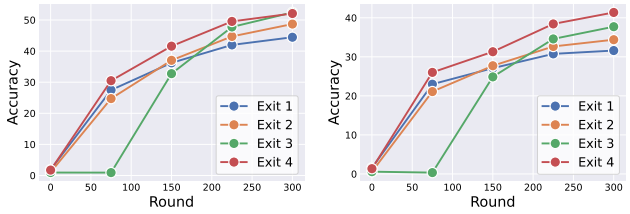
(a) Weight under Dir(0.3) (b) Weight under Dir(0.1)

Figure 8: Visualization of cross-client KD weight.

5.4 Ablation Studies

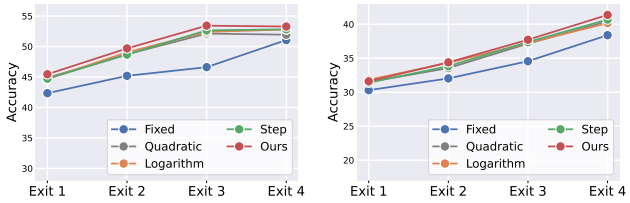
5.4.1 Contributions of Individual Components. Table. 5 compares four variants of X-FED on CIFAR-100 and TinyImageNet under Dir(0.3). Naively adding local KD does not improves accuracy, which aligns with Sec. 5.3.2. With cross-client KD (*i.e.*, cross-client teacher matching in Sec. 4.3), the accuracy of all exits increases by 5.77-10.03% on CIFAR-100 and 4.24-8.48% on TinyImageNet. Adding conflict-aware student selection in Sec. 4.2 further improves the performance of all exits by 1.17-5.21% and 1.17-3.09%.

5.4.2 Round-to-Accuracy. Fig. 9 plots the round-to-accuracy curves of all exits. The accuracy of each exit increases asynchronously. The teacher exit *i.e.*, Exit 4, consistently yields the highest accuracy.



(a) CIFAR-100 (b) TinyImageNet

Figure 9: Round-Accuracy curves.



(a) CIFAR-100 (b) TinyImageNet

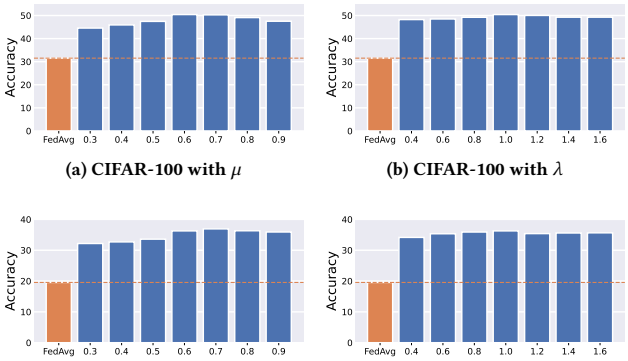
Figure 10: Impact of selection rate $R(t)$.

Initially, Exit 1 outperforms Exit 2 and Exit 3 because the latter two have not been selected yet. As the training progresses, Exit 3 gradually surpasses Exit 1 and 2. It implies that our exit selection mechanism does not compromise the performance of deeper exits, even though they are only involved in the later stage of training.

5.4.3 Impact of Selection Rate $R(t)$. This experiment compares different options for the selection rate $R(t)$. Given T rounds, X-FED increases $R(t)$ linearly, *i.e.*, $R(t) = \min(2t/T, 1)$ (see Sec. 4.2). We explore four alternatives: (1) *Fixed*: $R(t) = 1$. (2) *Quadratic*: $R(t) = \min((2t/T)^2, 1)$. (3) *Logarithm*: $R(t) = \min(\log(2(e-1)t/T + 1), 1)$. (4) *Step*: Every $T/20$ rounds, $R(t)$ increases by 0.1 until it reaches 1. Fig. 10 presents the per-exit accuracies of EENs trained with different $R(t)$. All strategies except *fixed* (*i.e.*, w/o exit selection) achieve comparable performance. The worst among the four strategies still outperforms *fixed* by 2.33%, 3.48%, 5.51% and 0.92% for CIFAR-100, and 1.10%, 1.52%, 2.58% and 1.78% for TinyImageNet. We pick the linear strategy for its simplicity.

5.4.4 Impact of Hyperparameters. Fig. 11 shows the hyperparameter sensitivity of X-FED. We test two key hyperparameters: μ that regularizes the cross-client KD weight $\mathbf{k}$ in Eq. (7), and λ for KD in Eq. (5). We vary μ from 0.3 to 0.9, and the optimal performance is attained when $\mu \in [0.6, 0.8]$. A large μ uniforms the cross-client KD weight. Teacher exits from irrelevant clients may have a greater influence on local training. Conversely, a small μ forces cross-client KD into local KD. Regarding λ, X-FED demonstrates robust performance across a range from 0.4 to 1.6.

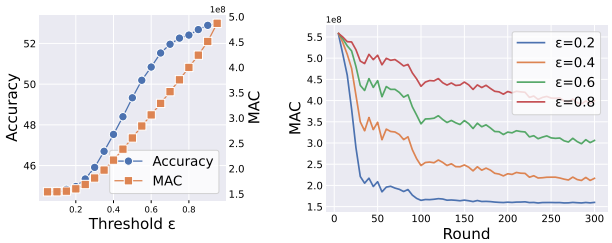
5.4.5 Impact of Exit Policy. We evaluate X-FED under different exit thresholds ϵ on CIFAR-100 with Dir(0.3). Fig. 12a shows that both accuracy and inference cost increase with the threshold ϵ. This is because the model is likely to terminate at a deeper exit given



(a) CIFAR-100 with μ (b) CIFAR-100 with λ

(c) TinyImageNet with μ (d) TinyImageNet with λ

Figure 11: Impact of hyperparameters.



(a) Accuracy vs. MAC (b) Round vs. MAC

Figure 12: Impact of threshold ϵ in exit policy.

high threshold. We find $\epsilon = 0.6$ better balance between inference accuracy and efficiency. Compared to full-model inference, there is an accuracy drop of 2.01% for X-FED, yet the number of MACs is reduced by 37.15%. Fig. 12b shows that the inference cost drops during training given different thresholds. This is because shallow exits become increasingly more confident in their outputs.

5.5 Case Study

Fig. 13 presents a case study by deploying the trained models of X-FED, FedAvg [33] and FedPer [1] on an edge device (Jetson-Nano with 8GB Memory, see Fig. 13a). For clear illustration, we only report the results of the first 10 clients out of 100 clients in training. We evaluate *(i) averaged accuracy* and *(ii) averaged inference latency* of each client on the corresponding testset. We choose ResNet-18 as model, and CIFAR-100 as dataset (see Fig. 13b). We also add Gaussian noise with $\sigma = 0.1$ to the test images to simulate noisy environments (see Fig. 13c). The exit confidence threshold is set to 0.5. Compared with the baselines, X-FED achieves higher accuracy with lower inference latency on most clients. Specifically, X-FED yields an accuracy gain of 15.17-24.81%, 9.36-26.66% and reduces inference latency by 2.05×, 1.99× in CIFAR-100, and CIFAR-100 with noise. The latency is slightly higher in the noisy setting because the model often fails to produce confident outputs at early exits.

(a) Deployment (b) CIFAR-100 (c) CIFAR-100 with Noise

Figure 13: Case study on edge devices: (a) deployment on Jetson Nano; accuracy-latency tradeoff on (b) CIFAR-100 and (c) CIFAR-100 with noise.

6 Conclusion

In this paper, we propose X-FED, a Conflict-Aware Cross-Client Federated Exit Distillation framework that extends personalized federated learning to early-exit networks. By jointly addressing client-wise heterogeneity and depth-wise interference, X-FED harmonizes exit training in a unified framework. Its progressive, depth-prioritized coordination strategy mitigates conflicts between shallow and deep exits and allows effective knowledge transfer across clients, while the client-decoupled formulation avoids excessive communication overhead. Extensive experiments on four benchmark datasets show that X-FED outperforms state-of-the-art PFL and GFL-EE baselines in accuracy and reduces inference costs by 30.79%–46.86%. We envision X-FED as a first step towards personalized and dynamic models from decentralized, heterogeneous IoT data. In the future, we plan to explore optimizations for client resource heterogeneity and expand X-FED to multimodal scenarios.

Acknowledgments

This research is partially supported by National Natural Science Foundation of China (NSFC) (No. 62572412), the RGC of Hong Kong SAR, China (Project No. CityU 11206425), and City University of Hong Kong Shenzhen Research Institute (CityUHKSRI). This work is partially supported by National Natural Science Foundation of China (NSFC) (Grant Nos. 62425202, 62336003), the Beijing Natural Science Foundation (Z230001), the Fundamental Research Funds for the Central Universities No.JKF-2026007844235, and the State Key Laboratory of Complex & Critical Software Environment (SKLCCSE). Zimu Zhou and Yongxin Tong are the corresponding authors.

A Appendix

A.1 Proof of Proposition 1

Proof. Taking the definition of KL divergence into Eq. (4):

$$\begin{aligned}\theta^* &= \arg\min_\theta \textstyle\sum_{i=1}^n k_i \theta_{im}(x) \log \frac{\theta_{im}(x)}{\theta(x)} \\ &= \arg\min_\theta - \textstyle\sum_{i=1}^n k_i \theta_{im}(x) \log \theta(x) \\ &= \arg\min_\theta - \log \theta(x) \textstyle\sum_{i=1}^n k_i \theta_{im}(x)\end{aligned} \tag{13}$$

The model θ_{im} can be further expressed as $\theta_{im} = (\phi, h_{im})$. The global backbone ϕ is shared by all clients. The linear classifier h_{im} is kept locally. Let the model with the aggregated classifier be

$\bar{\theta} = (\phi, \bar{h})$. The averaged classifier matrix is $\bar{h} = \sum_{i=1}^{n} k_i h_{im}$. Let the output of block m of backbone ϕ be $\phi(x)$, we have:

$$\bar{\theta}(x) = \bar{h}(\phi(x)) = \phi(x)^\top \bar{h} = \phi(x)^\top \sum_{i=1}^{n} k_i h_{im} = \sum_{i=1}^{n} k_i \phi(x)^\top h_{im} \tag{14}$$

Since the backbone ϕ is shared across all clients, the output of exit θ_{im} can be represented by $\theta_{im}(x) = h_{im}(\phi(x)) = \phi(x)^\top h_{im}$. Taking it into Eq. (14), there is $\bar{\theta}(x) = \sum_{i=1}^{n} k_i \theta_{im}(x)$. Taking it into Eq. (13), we have:

$$\begin{aligned} \theta^* &= \arg\min_\theta -\bar{\theta}(x) \log \theta(x) \\ &= \arg\min_\theta \bar{\theta}(x)(\log \bar{\theta}(x) - \log \theta(x)) = \arg\min_\theta D_{KL}(\bar{\theta} || \theta) \end{aligned} \tag{15}$$

Combining Eq. (4) and Eq. (15), we complete the proof. □

A.2 Convergence Analysis of X-FED

Theorem 1. *(Convergence of X-FED). Under Assumption 1–Assumption 3, X-FED satisfies:*

$$\frac{1}{T}\sum_{t=1}^{T}(\mathbb{E}\left\|\nabla_\phi F(\phi^t, h^t)\right\|^2 + s\mathbb{E}\left\|\nabla_h F_i(\phi^t, h_i^t)\right\|^2) \le \frac{2\Delta F_0}{E\eta T} + \eta\Phi_1 + \eta^2\Phi_2 + O(\delta_i) \tag{16}$$

with $\Phi_1 = \frac{L_\phi(1+\chi^2)E^2(G_\phi^2+\sigma_\phi^2)}{2} + \frac{2sL_h(1+\chi^2)}{n}(4(E-1)^2(G_h^2+\sigma_h^2) + E^2(G_h^2+\sigma_h^2))$ *and* $\Phi_2 = \frac{E^3}{3}(L_\phi^2(G_\phi^2+\sigma_\phi^2) + \chi^2 L_h L_\phi(G_h^2+\sigma_h^2)) + sE^3(\chi^2 L_\phi L_h(G_\phi^2+\sigma_\phi^2) + L_h^2(G_h^2+\sigma_h^2))$.

Assumptions. Let ϕ be the backbone and $h_i = \{h_{ij} | j \in \mathbb{S}_i\}$ the exits of client i. We adopt the standard assumptions in FL convergence analysis [28, 39], adapted to our two-block (backbone–exit) objective:

Assumption 1. *(Smoothness). F_i is block-wise L-smooth: $\phi \mapsto \nabla_\phi F_i(\phi, h_i)$ is L_ϕ-smooth, $h_i \mapsto \nabla_h F_i(\phi, h_i)$ is L_h-smooth, and the cross terms satisfy* $\max\{L_{\phi h}, L_{h\phi}\} \le \chi\sqrt{L_h L_\phi}$ *for some* $\chi > 0$.

Assumption 2. *(Unbiased gradient).* $\mathbb{E}_{\xi_i \sim D_i} \nabla F_i(\phi, h_i; \xi_i) = \nabla F_i(\phi, h_i)$ *for both* ∇_ϕ *and* ∇_h.

Assumption 3. *(Bounded variance and gradient). For both* ∇_ϕ *and* ∇_h, $\mathbb{E}\|\nabla F_i(\phi, h_i; \xi_i) - \nabla F_i(\phi, h_i)\|^2 \le \sigma_\bullet^2$ *and* $\mathbb{E}\|\nabla F_i(\phi, h_i)\|^2 \le G_\bullet^2$, *with* $\bullet \in \{\phi, h\}$.

Proof Outline. The overall objective is $F(\phi^t, h^t) = \sum_{i=1}^{n} p_i F_i(\phi^t, h_i^t)$. Following Lemma 20 of [39], we have the per-round descent:

$$\begin{aligned} &F(\phi^{t+1}, h^{t+1}) - F(\phi^t, h^t) \\ &\le \underbrace{\langle \nabla_\phi F(\phi^t, h^t), \phi^{t+1} - \phi^t \rangle}_{A_1} + \underbrace{\tfrac{1}{n}\sum_{i=1}^{n} \left\langle \nabla_h F_i(\phi^t, h_i^t), h_i^{t+1} - h_i^t \right\rangle}_{B_1} \\ &\quad + \underbrace{\tfrac{L_\phi(1+\chi^2)}{2}\left\|\phi^{t+1} - \phi^t\right\|^2}_{A_2} + \underbrace{\tfrac{1}{n}\sum_{i=1}^{n} \tfrac{L_h(1+\chi^2)}{2}\|h_i^{t+1} - h_i^t\|^2}_{B_2} \end{aligned} \tag{17}$$

A_1, A_2 concern the shared backbone and follow standard FedAvg arguments [28, 39]. The novelty lies in B_1, B_2, which involve X-FED's cross-client KD update of teacher exits with weights k_j^i.

Lemma 1. *(Local drift, standard [39]). For backbone* $\tilde{\phi}_i^{t,e}$ *and exits* $\tilde{h}_i^{t,e}$ *at local epoch* e*:*

$$\mathbb{E}\|\tilde{\phi}_i^{t,e} - \phi^t\|^2 \le (e-1)^2\eta^2(G_\phi^2+\sigma_\phi^2), \quad \mathbb{E}\|\tilde{h}_i^{t,e} - h_i^t\|^2 \le (e-1)^2\eta^2(G_h^2+\sigma_h^2) \tag{18}$$

The proof follows by writing each drift as a telescoping sum of one-step updates, applying Jensen's inequality, and using Assumption 2–Assumption 3; we omit it (cf. [39]).

$$\mathbb{E}\|\sum_{e=1}^{E} \nabla_\phi F_i(\tilde{\phi}_i^{t,e}, h_i; \xi_i)\|^2 \le E^2(G_\phi^2 + \sigma_\phi^2) \tag{19}$$

follows as a corollary by same argument for cumulative gradient.

Lemma 2. *(Bounding* A_1*).* $\mathbb{E}[A_1] \le -\frac{\eta E}{2}\mathbb{E}\|\nabla_\phi F(\phi^t, h^t)\|^2 + \frac{(E\eta)^3}{3}(L_\phi^2(G_\phi^2 + \sigma_\phi^2) + \chi^2 L_h L_\phi(G_h^2+\sigma_h^2))$.

Lemma 3. *(Bounding* A_2*).* $\mathbb{E}[A_2] \le \frac{L_\phi(1+\chi^2)\eta^2E^2(G_\phi^2+\sigma_\phi^2)}{2}$.

Lemma 2 follows by introducing a positive term, applying $-\langle a, b\rangle \le \frac{1}{2}\|a\|^2 + \frac{1}{2}\|b\|^2$, smoothness (Assumption 1), and Lemma 1—a derivation identical to standard FedAvg analysis on a single block [39]. Lemma 3 follows directly from Eq. (19) and Jensen.

Lemma 4. *(Bounding* B_1*).* $\mathbb{E}[B_1] \le -\frac{s\eta E}{2}\mathbb{E}\|\nabla_h F_i(\phi^t, h_i^t)\|^2 + \eta G_h \delta_i + s(E\eta)^3(\chi^2 L_\phi L_h(G_\phi^2+\sigma_\phi^2) + L_h^2(G_h^2+\sigma_h^2))$.

Proof. Only the s-fraction of clients in $\mathbb{C}$ update exits, so $h_i^{t+1} - h_i^t = 0$ for the rest. Split $B_1 = \frac{1}{n}\sum_{i\in\mathbb{C}}(B_{11} + B_{12})$ where $B_{11} = \langle \nabla_h F_i(\phi^t, h_i^t), h_{im}^{t+1} - h_{im}^{t,E}\rangle$ captures the cross-client KD aggregation (novel) and $B_{12} = \langle \nabla_h F_i(\phi^t, h_i^t), h_i^{t,E} - h_i^t\rangle$ captures the local drift (analogous to A_1).

For B_{11}, substituting $h_{im}^{t+1} = \sum_{j\in\mathbb{C}} k_j^i h_{jm}^{t,E}$ produces three inner products over $j \ne i$:

$$\begin{aligned} \mathbb{E}[B_{11}] = \sum_{j\in\mathbb{C}, j\ne i} k_j^i \big[&\langle \nabla_h F_i(\phi^t, h_i^t), h_{jm}^{t,E} - h_{jm}^t \rangle \\ &+ \langle \nabla_h F_i(\phi^t, h_i^t), h_{jm}^t - h_{im}^t \rangle + \langle \nabla_h F_i(\phi^t, h_i^t), h_{im}^t - h_{im}^{t,E} \rangle \big]. \end{aligned} \tag{20}$$

The first and third terms are bounded by $\frac{(E\eta)^3}{3}(\chi^2 L_\phi L_h(G_\phi^2+\sigma_\phi^2) + L_h^2(G_h^2+\sigma_h^2))$ each, via $-\langle a, b\rangle \le \frac{1}{2}\|a-b\|^2$, smoothness, and Lemma 1 (note $\nabla_h F_{im}$ is a special case of $\nabla_h F_i$). The second term is the cross-client heterogeneity contribution. Defining $\delta_i := \sum_{j\in\mathbb{C}, j\ne i} k_j^i \|h_{jm}^t - h_{im}^t\|^2/\eta$ as the system data heterogeneity, Cauchy–Schwarz and Assumption 3 give $\le \eta G_h \delta_i$. For B_{12}, the same argument as Lemma 2 yields $-\frac{\eta E}{2}\mathbb{E}\|\nabla_h F_i(\phi^t, h_i^t)\|^2 + \frac{(E\eta)^3}{3}(\chi^2 L_\phi L_h(G_\phi^2+\sigma_\phi^2) + L_h^2(G_h^2+\sigma_h^2))$. Summing B_{11} and B_{12} completes the proof. □

Lemma 5. *(Bounding* B_2*).* $\mathbb{E}[B_2] \le \frac{2sL_h(1+\chi^2)}{n}(2\eta\delta_i + 4(E-1)^2\eta^2(G_h^2+\sigma_h^2) + \eta^2E^2(G_h^2+\sigma_h^2))$.

Proof. Apply $\|h_i^{t+1} - h_i^t\|^2 \le 2\|h_{im}^{t+1} - h_{im}^{t,E}\|^2 + 2\|h_i^{t,E} - h_i^t\|^2$, denoting the two parts B_{21} and B_{22}. For B_{21}, expanding $h_{im}^{t+1} = \sum_{j\in\mathbb{C}} k_j^i h_{jm}^{t,E}$ and twice applying $\|a+b\|^2 \le 2\|a\|^2 + 2\|b\|^2$ gives

$$B_{21} \le 2\sum_{j\in\mathbb{C}} k_j^i (2\|h_{jm}^t - h_{im}^t\|^2 + 4\|h_{jm}^{t,E} - h_{jm}^t\|^2 + 4\|h_{im}^t - h_{im}^{t,E}\|^2), \tag{21}$$

which by the definition of δ_i and Lemma 1 is at most $4\eta\delta_i + 8(E-1)^2\eta^2(G_h^2+\sigma_h^2)$. B_{22} follows the same Jensen argument as Lemma 3 and is bounded by $2\eta^2E^2(G_h^2+\sigma_h^2)$. Combining yields the result. □

Substituting Lemma 2–Lemma 5 into Eq. (17) bounds $F(\phi^{t+1}, h^{t+1}) - F(\phi^t, h^t)$. Telescoping over $t = 0, \ldots, T-1$ completes the proof.

GenAI Usage Disclosure

The author used GenAI to revise code and polish the paper.

References

[1] Manoj Ghuhan Arivazhagan, Vinay Aggarwal, Aaditya Kumar Singh, and Sunav Choudhary. 2019. Federated learning with personalization layers. *arXiv preprint arXiv:1912.00818* (2019).

[2] Daoyuan Chen, Dawei Gao, Weirui Kuang, Yaliang Li, and Bolin Ding. 2022. pfl-bench: A comprehensive benchmark for personalized federated learning. *NeurIPS* 35 (2022), 9344–9360.

[3] Daoyuan Chen, Liuyi Yao, Dawei Gao, Bolin Ding, and Yaliang Li. 2023. Efficient personalized federated learning via sparse model-adaptation. In *ICML*. 5234–5256.

[4] Huancheng Chen and Haris Vikalo. 2024. Heterogeneity-Guided Client Sampling: Towards Fast and Efficient Non-IID Federated Learning. In *NeurIPS*.

[5] Hong-You Chen and Wei-Lun Chao. 2022. On Bridging Generic and Personalized Federated Learning for Image Classification. In *ICLR*.

[6] Siyao Cheng, Boyi Liu, Zimu Zhou, Zheng Wu, and Yongxin Tong. 2026. Progressive Collaboration Pruning for Federated Learning. In *CIKM*.

[7] Patryk Chrabaszcz, Ilya Loshchilov, and Frank Hutter. 2017. A downsampled variant of imagenet as an alternative to the cifar datasets. *arXiv preprint arXiv:1707.08819* (2017).

[8] Liam Collins, Hamed Hassani, Aryan Mokhtari, and Sanjay Shakkottai. 2021. Exploiting shared representations for personalized federated learning. In *ICML*. 2089–2099.

[9] Enmao Diao, Jie Ding, and Vahid Tarokh. 2020. HeteroFL: Computation and Communication Efficient Federated Learning for Heterogeneous Clients. In *ICLR*.

[10] Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. 2020. Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach. *NeurIPS* 33 (2020), 3557–3568.

[11] Cheng Gong, Yao Chen, Qiuyang Luo, Ye Lu, Tao Li, Yuzhi Zhang, Yufei Sun, and Le Zhang. 2024. Deep Feature Surgery: Towards Accurate and Efficient Multi-exit Networks. In *ECCV*. 435–451.

[12] Seokil Ham, Jungwuk Park, Dong-Jun Han, and Jaekyun Moon. 2024. NEO-KD: knowledge-distillation-based adversarial training for robust multi-exit neural networks. *NeurIPS* 36 (2024).

[13] Yizeng Han, Gao Huang, Shiji Song, Le Yang, Honghui Wang, and Yulin Wang. 2021. Dynamic neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 44, 11 (2021), 7436–7456.

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *CVPR*. 770–778.

[15] Gao Huang, Danlu Chen, Tianhong Li, Felix Wu, Laurens van der Maaten, and Kilian Weinberger. 2018. Multi-Scale Dense Networks for Resource Efficient Image Classification. In *ICLR*.

[16] Jiaming Huang, Yi Gao, and Wei Dong. 2024. Unlocking the Non-deterministic Computing Power with Memory-Elastic Multi-Exit Neural Networks. In *TheWebConf*.

[17] Yutao Huang, Lingyang Chu, Zirui Zhou, Lanjun Wang, Jiangchuan Liu, Jian Pei, and Yong Zhang. 2021. Personalized cross-silo federated learning on non-iid data. In *AAAI*, Vol. 35. 7865–7873.

[18] Fatih Ilhan, Gong Su, and Ling Liu. 2023. Scalefl: Resource-adaptive federated learning with heterogeneous clients. In *CVPR*. 24532–24541.

[19] Yigitcan Kaya, Sanghyun Hong, and Tudor Dumitras. 2019. Shallow-deep networks: Understanding and mitigating network overthinking. In *ICML*. 3301–3310.

[20] Minjae Kim, Sangyoon Yu, Suhyun Kim, and Soo-Mook Moon. 2023. DepthFL: Depthwise federated learning for heterogeneous clients. In *ICLR*.

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

[22] Solomon Kullback. 1997. *Information theory and statistics*. Courier Corporation.

[23] Stefanos Laskaridis, Alexandros Kouris, and Nicholas D Lane. 2021. Adaptive inference through early-exit networks: Design, challenges and directions. In *EMDL*. 1–6.

[24] Royson Lee, Javier Fernandez-Marques, Shell Xu Hu, Da Li, Stefanos Laskaridis, Łukasz Dudziak, Timothy Hospedales, Ferenc Huszár, and Nicholas Donald Lane. 2024. Recurrent Early Exits for Federated Learning with Heterogeneous Clients. In *ICML*.

[25] Daliang Li and Junpu Wang. 2019. Fedmd: Heterogenous federated learning via model distillation. *arXiv preprint arXiv:1910.03581* (2019).

[26] Tian Li, Shengyuan Hu, Ahmad Beirami, and Virginia Smith. 2021. Ditto: Fair and robust federated learning through personalization. In *ICML*. 6357–6368.

[27] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. 2020. Federated optimization in heterogeneous networks. *MLSys* 2 (2020), 429–450.

[28] Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. 2020. On the Convergence of FedAvg on Non-IID Data. In *ICLR*.

[29] Zhilin Liang, Yuxiang Wang, Zimu Zhou, Hainan Zhang, Boyi Liu, and Yongxin Tong. 2026. FedMosaic: Federated Retrieval-Augmented Generation via Parametric Adapters. *SIGIR* (2026).

[30] Boyi Liu, Yiming Ma, Zimu Zhou, Yexuan Shi, Shuyuan Li, and Yongxin Tong. 2024. CASA: Clustered Federated Learning with Asynchronous Clients. In *SIGKDD*. 1851–1862.

[31] Boyi Liu, Zimu Zhou, Pengfei Gao, Shuo Kang, and Yongxin Tong. 2026. Towards Asynchronous Client Collaboration in Personalized Federated Learning. In *INFOCOM*. 1–10.

[32] Yiming Ma, Boyi Liu, Zimu Zhou, Yanfeng Wang, and Yongxin Tong. 2026. Harnessing Asynchrony to Balance Modalities in Multi-modal Federated Learning. In *DASFAA*. 455–471.

[33] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-efficient learning of deep networks from decentralized data. In *AISTATS*. 1273–1282.

[34] Jaehoon Oh, SangMook Kim, and Se-Young Yun. 2022. FedBABU: Toward Enhanced Representation for Federated Image Classification. In *ICLR*.

[35] Xiaomin Ouyang, Xian Shuai, Yang Li, Li Pan, Xifan Zhang, Heming Fu, Sitong Cheng, Xinyan Wang, Shihua Cao, Jiang Xin, et al. 2024. ADMarker: A Multi-Modal Federated Learning System for Monitoring Digital Biomarkers of Alzheimer's Disease. In *MobiCom*. 404–419.

[36] Yibo Pan, Boyi Liu, Zimu Zhou, Fan Gao, Tianyi Wang, Yi Xu, and Yongxin Tong. 2026. HiMAC: Hierarchical Federated Learning with Mobility-Aware Collaboration. In *CIKM*.

[37] Mary Phuong and Christoph Lampert. 2019. Towards understanding knowledge distillation. In *ICML*. 5142–5151.

[38] Mary Phuong and Christoph H Lampert. 2019. Distillation-based training for multi-exit architectures. In *ICCV*. 1355–1364.

[39] Krishna Pillutla, Kshitiz Malik, Abdel-Rahman Mohamed, Mike Rabbat, Maziar Sanjabi, and Lin Xiao. 2022. Federated learning with partial model personalization. In *ICML*. 17716–17758.

[40] Lehao Qu, Shuyuan Li, Zimu Zhou, Boyi Liu, Yi Xu, and Yongxin Tong. 2025. Darkdistill: Difficulty-aligned federated early-exit network training on heterogeneous devices. In *SIGKDD*. 2374–2385.

[41] Haseena Rahmath P, Vishal Srivastava, Kuldeep Chaurasia, Roberto G Pacheco, and Rodrigo S Couto. 2024. Early-Exit Deep Neural Network-A Comprehensive Survey. *Comput. Surveys* 57, 3 (2024), 1–37.

[42] Florence Regol, Joud Chataoui, and Mark Coates. 2024. Jointly-learned exit and inference for a dynamic neural network: Jei-dnn. In *ICLR*.

[43] Felix Sattler, Klaus-Robert Müller, and Wojciech Samek. 2020. Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints. *IEEE Transactions on Neural Networks and Learning Systems* 32, 8 (2020), 3710–3722.

[44] Alysa Ziying Tan, Han Yu, Lizhen Cui, and Qiang Yang. 2022. Towards personalized federated learning. *IEEE Transactions on Neural Networks and Learning Systems* 34, 12 (2022), 9587–9603.

[45] Surat Teerapittayanon, Bradley McDanel, and Hsiang-Tsung Kung. 2016. Branchynet: Fast inference via early exiting from deep neural networks. In *ICPR*.

[46] Yansheng Wang, Yongxin Tong, Zimu Zhou, Ruisheng Zhang, Sinno Jialin Pan, Lixin Fan, and Qiang Yang. 2023. Distribution-regularized federated learning on non-iid data. In *ICDE*. 2113–2125.

[47] Shuyue Wei, Yongxin Tong, Zimu Zhou, Tianran He, and Yi Xu. 2025. Efficient data valuation approximation in federated learning: A sampling-based approach. In *ICDE*. 2922–2934.

[48] Jian Xu, Xinyi Tong, and Shao-Lun Huang. 2023. Personalized Federated Learning with Feature Alignment and Classifier Collaboration. In *ICLR*.

[49] Kunda Yan, Sen Cui, Abudukelimu Wuerkaixi, Jingfeng Zhang, Bo Han, Gang Niu, Masashi Sugiyama, and Changshui Zhang. 2024. Balancing Similarity and Complementarity for Federated Learning. In *ICML*.

[50] Le Yang, Yizeng Han, Xi Chen, Shiji Song, Jifeng Dai, and Gao Huang. 2020. Resolution adaptive networks for efficient inference. In *CVPR*. 2369–2378.

[51] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. 2019. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology* 10, 2 (2019), 1–19.

[52] Rui Ye, Zhenyang Ni, Fangzhao Wu, Siheng Chen, and Yanfeng Wang. 2023. Personalized federated learning with inferred collaboration graphs. In *ICML*. 39801–39817.

[53] Haichao Yu, Haoxiang Li, Gang Hua, Gao Huang, and Humphrey Shi. 2023. Boosted dynamic neural networks. In *AAAI*, Vol. 37. 10989–10997.

[54] Jie Zhang, Song Guo, Xiaosong Ma, Haozhao Wang, Wenchao Xu, and Feijie Wu. 2021. Parameterized knowledge transfer for personalized federated learning. *NeurIPS* 34 (2021), 10092–10104.

[55] Tianlong Zhang, Xiaoxi He, Yuxiang Wang, Yi Xu, Rendi Wu, Zhifei Wang, and Yongxin Tong. 2025. FedMetro: Efficient metro passenger flow prediction via federated graph learning. In *KDD*. 5215–5224.

[56] Wenhao Zhang, Zimu Zhou, Yansheng Wang, and Yongxin Tong. 2023. DM-PFL: Hitchhiking Generic Federated Learning for Efficient Shift-Robust Personalization. In *SIGKDD*. 3396–3408.

[57] Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *NeurIPS* 28 (2015).

[58] Zhuangdi Zhu, Junyuan Hong, and Jiayu Zhou. 2021. Data-free knowledge distillation for heterogeneous federated learning. In *ICML*. 12878–12889.