FEDGEO: FINE-TUNING-FREE FEDERATED PERSONALIZATION VIA GEOMETRIC STABILITY

Anonymous authors
Paper under double-blind review

ABSTRACT

Personalized federated learning addresses performance degradation under label distribution skew. Fine-tuning-based methods require local optimization for unseen clients. Meta-learning-based generation can avoid these updates, but implicit representations and unconstrained generators may produce unstable parameters and lower unseen-client accuracy. We propose Fine-Tuning-Free **Fed**erated Personalization via **Geo**metric Stability (**FedGeo**), a framework for generalization to unseen clients. Given a label prior estimated from local labeled support, FedGeo uses a geometry-preserving encoder and a spectrally constrained affine generator to produce personalized modulation parameters without local gradient updates. Together, distance preservation and bounded parameter sensitivity provide geometric stability. We establish generalization error bounds and conditional convergence guarantees under federated label-skewed distributions. Experiments demonstrate strong performance for both seen and unseen clients across multiple benchmarks. The anonymous source code is available at `https://anonymous.4open.science/r/FedGeo`.

1 INTRODUCTION

In federated learning (FL), a shared global model can struggle to generalize under severe label distribution skew. Statistical heterogeneity arises in medical image analysis (Yang et al., 2024) and Internet of Things (IoT) systems (Nguyen et al., 2021), where clients differ in label proportions, features, or both (Lu et al., 2024). Personalized federated learning (PFL) (Kairouz et al., 2021; Tan et al., 2023) optimizes models for individual client distributions. Recent work studies communication constraints (Huang & Shu, 2025; Yu et al., 2025) and distribution shifts, as in FLUX (Fenoglio et al., 2025) and FeRoMA (Li et al., 2026). We focus on fine-tuning-based personalization and meta-learning-based personalization.

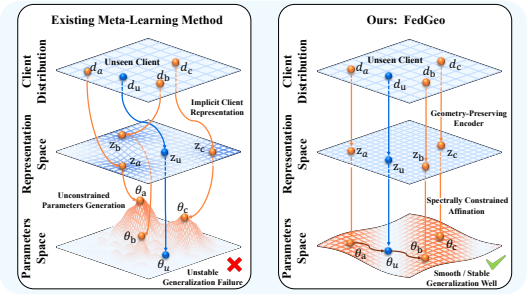


Figure 1: Motivation of FedGeo. *Left)* Implicit representations and unconstrained generation can distort distributional relationships and amplify parameter changes. *Right)* A Geometry-Preserving Encoder and Spectrally Constrained Affine Generator preserve prior geometry and control parameter sensitivity.

Fine-tuning-based personalization uses local optimization to adapt client parameters. Ditto (Li et al., 2021b) regularizes personalized models, FedRep (Collins et al., 2021) separates shared representations from local heads, and pFedMMA (Ghiasvand et al., 2026) trains shared and private adapters. Prototype methods such as FedProto (Tan et al., 2022) and FedCPD (Jin et al., 2025) guide local learning with class-level information. Constructing personalized models for unseen clients can require additional local optimization. Its computation and energy costs motivate efficient adaptation, as in FedSPU (Niu et al., 2025), and deployment with fewer client-side updates (Thakur et al., 2025).

Meta-learning-based personalization learns shared rules for constructing client models. pFedHN (Shamsian et al., 2021) adapts an embedding for each new client, whereas PeFLL (Scott

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.

et al., 2024) and HyperFedZero (Zhou et al., 2026) infer representations from data for deployment without fine-tuning. The encoder–generator approach in the left panel of Figure 1 represents each client and uses a hypernetwork to generate personalized parameters. Client distributions d_a, d_b, and d_c are mapped to representations z_a, z_b, and z_c, then to parameters θ_a, θ_b, and θ_c. Without explicit distance preservation, encoding can distort label-distribution geometry, and unconstrained generation can amplify representation differences. Nearby distributions may therefore produce distant personalized parameters for an unseen client d_u.

We propose Fine-Tuning-Free **Fed**erated Personalization via **Geo**metric Stability (**FedGeo**). As depicted in the right panel of Figure 1, we call the combination of distance-preserving encoding and bounded parameter sensitivity *geometric stability*. FedGeo uses (i) a Geometry-Preserving Encoder preserving prior distances up to a fixed scale when encoding d_u as z_u, and (ii) a Spectrally Constrained Affine Generator bounding parameter changes to stabilize θ_u for unseen clients. At deployment, local support labels determine the client prior; the trained encoder, generator, and backbone remain fixed. The main contributions are:

- We propose FedGeo, a fine-tuning-free PFL framework that stabilizes parameter generation with geometry-preserving client representations and spectrally constrained affine transformations.
- We provide theoretical analysis for FedGeo by establishing an unseen-client risk bound and a conditional excess-risk transfer result under label-skewed federated settings.
- Extensive experiments demonstrate strong performance for both seen and unseen clients across CIFAR-10/100, four backbones, and naturally partitioned FEMNIST.

2 Related Work

Fine-tuning-based personalization. Local optimization is central to many FL methods. FedProx (Li et al., 2020a) constrains local updates with a proximal term, while FedDyn (Acar et al., 2021) aligns local and global objectives through dynamic regularization. For personalization, Ditto (Li et al., 2021b) regularizes client models toward a global model; FedRep (Collins et al., 2021) trains local heads over shared representations; and FedPAC (Xu et al., 2023) combines feature alignment with classifier collaboration. FedBN (Li et al., 2021c) retains local normalization statistics, FedProto (Tan et al., 2022) shares class prototypes, and pFedMMA (Ghiasvand et al., 2026) trains shared and private multimodal adapters. FedMC (Ma et al., 2026) supports heterogeneous local training through feature calibration. These methods optimize client models using local data, whereas FedGeo generates modulation for a new client with fixed shared parameters.

Meta-learning-based personalization. Meta-learning can learn initializations for local adaptation (Jiang et al., 2019; Fallah et al., 2020) or mappings from client representations to parameters. The latter use hypernetworks (Ha et al., 2017) as parameter generators. pFedHN (Shamsian et al., 2021) generates full models from learned embeddings and adapts embeddings for new clients. MH-pFedHN (Zhang et al., 2026) extends generation to heterogeneous client architectures through shared hypernetwork heads. PeFLL (Scott et al., 2024) jointly learns an embedding network and a hypernetwork for fine-tuning-free inference, with future-client generalization guarantees. HyperFedZero (Zhou et al., 2026) uses distribution-aware embeddings and chunk-wise generation for non-participating clients. FedAGHN (Song et al., 2025) generates collaboration weights for personalized aggregation. FedGeo generates lightweight feature-wise linear modulation (FiLM) parameters for a shared backbone; geometric stability connects its explicit label-prior representation to bounded parameter sensitivity.

3 Problem Setup

Personalized federated learning. Consider K clients indexed by $[K] = \{1, \dots, K\}$. Client k holds local data $\mathcal{D}_k$, with training/support subset $\mathcal{D}_k^{\text{train}} = \{(x_i^{(k)}, y_i^{(k)})\}_{i=1}^{n_k}$ drawn from P_k and disjoint test subset $\mathcal{D}_k^{\text{test}}$, where $x_i^{(k)} \in \mathcal{X} \subseteq \mathbb{R}^{D_{in}}$, $y_i^{(k)} \in \mathcal{Y} = \{1, \dots, C\}$, and D_{in} and C are the input dimension and number of classes. Let $\mathcal{K}_{\text{train}} \subseteq [K]$ denote the training clients,

$n = \sum_{k \in \mathcal{K}_{\text{train}}} n_k$, and $p_k = n_k/n$. A model $f_\theta : \mathcal{X} \to \mathbb{R}^C$ has parameters $\theta \in \mathbb{R}^{D_{model}}$ and nonnegative loss $\ell : \mathbb{R}^C \times \mathcal{Y} \to \mathbb{R}_+$. Standard FL minimizes

$$\min_\theta F(\theta) = \sum_{k \in \mathcal{K}_{\text{train}}} p_k \mathcal{L}_k(\theta), \qquad \mathcal{L}_k(\theta) = \mathbb{E}_{(x,y)\sim P_k}[\ell(f_\theta(x), y)]. \tag{1}$$

FedAvg (McMahan et al., 2017) broadcasts $\theta^{(r)}$ to sampled clients $\mathcal{S}^{(r)} \subseteq \mathcal{K}_{\text{train}}$ in round r, obtains their locally updated parameters $\theta_k^{(r+1)}$, and aggregates

$$\theta^{(r+1)} = \sum_{k \in \mathcal{S}^{(r)}} a_k^{(r)} \theta_k^{(r+1)}, \qquad a_k^{(r)} = \frac{n_k}{\sum_{j \in \mathcal{S}^{(r)}} n_j}. \tag{2}$$

Under label skew, $P_k(Y)$ varies across clients (Zhao et al., 2018); pure label shift additionally assumes shared class-conditionals $P_k(X \mid Y)$ (Lipton et al., 2018). PFL instead learns client-specific parameters θ_k to minimize $\mathcal{L}_k(\theta_k)$.

Meta-learning-based personalization. Let $z_k \in \mathcal{Z}$ be a client representation and $h_\psi : \mathcal{Z} \to \mathbb{R}^{D_{gen}}$ a generator with parameters ψ, where D_{gen} is the generated parameter dimension. Full-model generators, as in pFedHN (Shamsian et al., 2021) and PeFLL (Scott et al., 2024), set $\theta_k = h_\psi(z_k)$ and $D_{gen} = D_{model}$. FedGeo encodes a class-prior descriptor $d_k \in \Delta^{C-1} = \{d \in \mathbb{R}_+^C : \sum_c d[c] = 1\}$ as $z_k = E_\phi(d_k)$ and generates modulation vectors $m_k = h_\psi(z_k)$ for a shared backbone. Here ϕ parameterizes the encoder, $f_\theta(x; m_k)$ denotes the modulated prediction, and $\Theta = (\theta, \phi, \psi)$ is trained jointly through

$$\min_\Theta \mathcal{F}(\Theta) = \sum_{k \in \mathcal{K}_{\text{train}}} p_k \mathcal{L}_k(\Theta), \qquad \mathcal{L}_k(\Theta) = \mathbb{E}_{(x,y)\sim P_k}[\ell(f_\theta(x; h_\psi(E_\phi(d_k))), y)]\,. \tag{3}$$

Local optimization uses the empirical loss $\widehat{\mathcal{L}}_{k,\mathcal{B}}$ averaged over mini-batch $\mathcal{B} \subseteq \mathcal{D}_k^{\text{train}}$. At deployment, an unseen client $u \notin \mathcal{K}_{\text{train}}$ uses local support labels solely to estimate its prior; support and test sets are disjoint. Fine-tuning-free personalization evaluates E_ϕ, h_ψ, and f_θ with fixed Θ, without a backward pass or optimizer step.

4 METHOD

FedGeo achieves geometric stability through two components: (i) a Geometry-Preserving Encoder that constructs explicit distribution-aware client representations; and (ii) a Spectrally Constrained Affine Generator that produces stable modulation parameters to dynamically adapt the shared backbone network.

As illustrated in Figure 2, the workflow proceeds as follows, where each client k first computes its prior label distribution d_k from the local dataset. This distribution is explicitly encoded into a client representation $z_k = E_\phi(d_k)$ via the encoder E_ϕ. Based on z_k, the affine generator h_ψ dynamically synthesizes layer-wise modulation parameters $\{\gamma_k, \beta_k\} = h_\psi(z_k)$ to modulate the shared backbone f_θ via FiLM (Perez et al., 2018). During training, the modulated model produces predictions for local samples, and the loss is backpropagated to jointly optimize ϕ, ψ, and θ. During inference, both seen clients and unseen clients can directly obtain personalized predictions using priors estimated from local labeled support, without local fine-tuning. The complete training and inference procedure is provided in Appendix D. We next elaborate on the two core components, followed by theoretical analysis.

4.1 GEOMETRY-PRESERVING ENCODER

Client representations condition the parameter generator. Meta-learning-based methods use learned embeddings, as in pFedHN (Shamsian et al., 2021), or latent vectors from data batches, as in PeFLL (Scott et al., 2024). These constructions do not enforce distance preservation between class-prior vectors. FedGeo's Geometry-Preserving Encoder E_ϕ represents the client prior while preserving its geometric structure.

Label-distribution information is used for generation in FedGen (Zhu et al., 2021), calibration in FedLC (Zhang et al., 2022) and CalFAT (Chen et al., 2022), and local learning or aggregation in

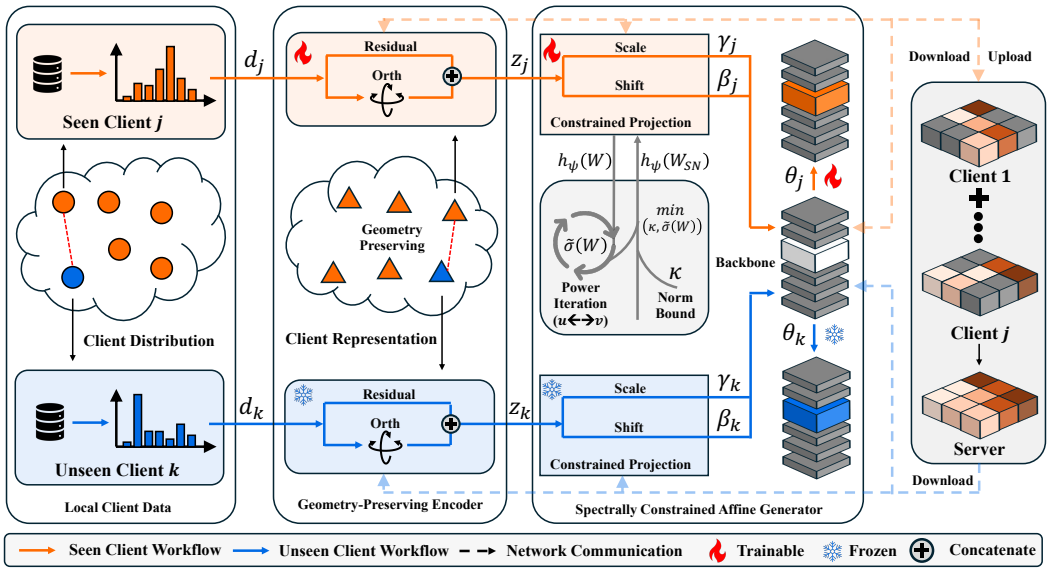


Figure 2: Overview of the FedGeo framework. The client's prior label distribution is encoded via a Geometry-Preserving Encoder to maintain distributional relationships. A Spectrally Constrained Affine Generator then produces FiLM parameters to modulate the shared backbone.

FedSLD (Luo & Wu, 2022) and FedDisco (Ye et al., 2023). In FedGeo, client k computes its prior $d_k \in \mathbb{R}^C$ from class frequencies in $\mathcal{D}_k^{train}$:

$$d_k[c] = \frac{1}{|\mathcal{D}_k^{train}|} \sum_{(x,y)\in\mathcal{D}_k^{train}} \mathbb{I}(y=c), \quad c \in \{1,\dots,C\}. \tag{4}$$

Here $\mathbb{I}$ is the indicator function. FedGeo retains the FedAvg-style training pipeline and does not transmit raw data or features. The prior remains local and is reused across mini-batches.

To obtain a learnable representation of this prior without distorting distributional relationships, we introduce a full-column-rank learnable weight matrix $M \in \mathbb{R}^{D_{orth}\times C}$ in the encoder E_ϕ, where $D_{orth} \geq C$. In each forward pass, we perform QR decomposition on M:

$$M = QR, \quad Q \in \mathbb{R}^{D_{orth}\times C}, \quad Q^\top Q = I_C, \tag{5}$$

where Q is an orthonormal basis and $R \in \mathbb{R}^{C\times C}$ is upper triangular. The final client representation is constructed as:

$$z_k = E_\phi(d_k) = [d_k \,\|\, Qd_k] \in \mathbb{R}^{C+D_{orth}}, \tag{6}$$

where $\|$ denotes vector concatenation. Gradients propagate through QR, and Q is recomputed from the current M at each local update.

This design guarantees scaled isometry: distances are preserved up to a fixed constant factor. For any two clients i and j, orthonormality gives

$$\begin{aligned} \|z_i - z_j\|_2^2 &= \|d_i - d_j\|_2^2 + \|Q(d_i - d_j)\|_2^2 \\ &= \|d_i - d_j\|_2^2 + (d_i - d_j)^\top Q^\top Q(d_i - d_j) = 2\|d_i - d_j\|_2^2. \end{aligned} \tag{7}$$

Taking the square root yields $\|z_i - z_j\|_2 = \sqrt{2}\|d_i - d_j\|_2$. The distance between client representations is therefore strictly proportional to the distance between their underlying label distributions. This geometric consistency ensures that the affine generator receives structured conditions where relative distances between clients are maintained.

4.2 Spectrally Constrained Affine Generator

Let $\mathcal{A} = \{\tau_1, \dots, \tau_{|\mathcal{A}|}\}$ denote the set of adaptable layers in the backbone network. We adopt FiLM as the personalization mechanism. For each adaptable layer $\tau \in \mathcal{A}$ with D_τ channels in the backbone

f_θ, the intermediate feature map $x^{(\tau)}$ is modulated via:

$$x^{(\tau)}_{mod} = (1 + \gamma^{(\tau)}_k) \odot x^{(\tau)} + \beta^{(\tau)}_k, \tag{8}$$

where $\gamma^{(\tau)}_k$ and $\beta^{(\tau)}_k$ are scale and shift parameters, and $\odot$ denotes element-wise multiplication.

The critical challenge lies in synthesizing these parameters from z_k with stability guarantees. A straightforward approach employs an S-layer multi-layer perceptron (MLP) with 1-Lipschitz activations, such as the rectified linear unit (ReLU), for each adaptable layer τ. Writing $W^{(\tau)}_s$ for its layer weights, its Lipschitz upper bound is their norm product. For any two clients i and j:

$$\|h^{(\tau)}_\psi(z_i) - h^{(\tau)}_\psi(z_j)\|_2 \leq \left(\prod_{s=1}^{S} \|W^{(\tau)}_s\|_2 \right) \|z_i - z_j\|_2. \tag{9}$$

Without constraints on the layer norms, this product can grow with depth S, permitting minor distributional differences to produce large changes in personalization parameters. To ensure that geometric consistency translates into controlled sensitivity in parameter space, we propose a Spectrally Constrained Affine Generator, $h^{(\tau)}_\psi(z_k) = W^{(\tau)} z_k + b^{(\tau)}$, with explicit spectral control. We impose spectral truncation with threshold κ on each $W^{(\tau)}$ to control operator norms (Gouk et al., 2021). Unlike unit-norm normalization, truncation allows a flexible threshold. Using power iteration (Miyato et al., 2018), we maintain unit vectors $u^{(\tau)} \in \mathbb{R}^{2D_\tau}$ and $v^{(\tau)} \in \mathbb{R}^{C+D_{orth}}$, updated once per forward pass:

$$\tilde{v}^{(\tau)} \leftarrow \frac{(W^{(\tau)})^\top u^{(\tau)}}{\|(W^{(\tau)})^\top u^{(\tau)}\|_2}, \quad \tilde{u}^{(\tau)} \leftarrow \frac{W^{(\tau)} \tilde{v}^{(\tau)}}{\|W^{(\tau)} \tilde{v}^{(\tau)}\|_2}. \tag{10}$$

The spectral norm estimate $\tilde{\sigma}(W^{(\tau)}) = (\tilde{u}^{(\tau)})^\top W^{(\tau)} \tilde{v}^{(\tau)}$ triggers rescaling if exceeding κ:

$$W^{(\tau)}_{SN} = \min\left(1, \frac{\kappa}{\tilde{\sigma}(W^{(\tau)})}\right) W^{(\tau)}. \tag{11}$$

Exact norm rescaling enforces $\|W^{(\tau)}_{SN}\|_2 \leq \kappa$. With finite-iteration estimates, the actual bound is $\kappa_{\text{eff}} = \max_{\tau \in \mathcal{A}} \|W^{(\tau)}_{SN}\|_2$; the theoretical statements below use the condition $\kappa_{\text{eff}} \leq \kappa$. For a client with representation $z_k \in \mathbb{R}^{C+D_{orth}}$, we generate FiLM parameters for each $\tau \in \mathcal{A}$:

$$\begin{bmatrix} \gamma^{(\tau)}_k \\ \beta^{(\tau)}_k \end{bmatrix} = W^{(\tau)}_{SN} z_k + b^{(\tau)}, \tag{12}$$

where $W^{(\tau)}_{SN} \in \mathbb{R}^{2D_\tau \times (C+D_{orth})}$, $b^{(\tau)} \in \mathbb{R}^{2D_\tau}$, giving generated dimension $D_{gen} = 2\sum_{\tau \in \mathcal{A}} D_\tau$. The output is split into scale and shift parameters applied via Eq. 8. This design respects the distributional geometry in z_k while maintaining bounded sensitivity for stable personalization.

Under $\kappa_{\text{eff}} \leq \kappa$, Eq. 7 gives

$$\|h_\psi(E_\phi(d_i)) - h_\psi(E_\phi(d_j))\|_2 \leq \sqrt{2|\mathcal{A}|}\, \kappa \|d_i - d_j\|_2. \tag{13}$$

Together, Eqs. 7 and 13 formalize geometric stability: scaled distance preservation in representation space and bounded sensitivity in modulation space. The affine bias cancels; concatenation over layers contributes $\sqrt{|\mathcal{A}|}$. For finite-iteration rescaling, the bound holds with κ_{eff} in place of κ.

4.3 Theoretical Analysis

We bound the generalization of FedGeo on unseen clients. After R communication rounds, the learned parameters $\Theta^{(R)} = (\theta^{(R)}, \phi^{(R)}, \psi^{(R)})$ support direct personalization without fine-tuning. Write $m^{(R)}(d) = h_{\psi^{(R)}}(E_{\phi^{(R)}}(d))$. For an unseen client u, the prior d_u is encoded as $z^{(R)}_u = E_{\phi^{(R)}}(d_u)$, and $m^{(R)}(d_u)$ supplies the modulation for prediction. In the analysis, $d_j = P_j(Y)$ is the population prior and $\mathcal{L}^{(R)}_j$ denotes the expected loss of client j with its generated modulation. We use the following assumptions.

Assumption 4.1 (Loss Lipschitz continuity)**.** For each label y, the loss function $\ell(\hat{y}, y)$ is L_ℓ-Lipschitz:

$$|\ell(\hat{y}_1, y) - \ell(\hat{y}_2, y)| \leq L_\ell \|\hat{y}_1 - \hat{y}_2\|_2. \tag{14}$$

Table 1: Performance comparison of FedGeo and baselines on CIFAR-10/100 with LeNet-5 under pathological and practical label skew. Accuracies (%) cover seen and unseen clients.

	Pathological				Practical ($\alpha = 0.1$)			
	CIFAR-10		CIFAR-100		CIFAR-10		CIFAR-100	
Method	Seen	Unseen	Seen	Unseen	Seen	Unseen	Seen	Unseen
FedAvg	56.91±1.3*	57.74±1.0	28.69±0.5*	28.21±2.2*	56.68±1.1*	62.16±1.8*	30.29±0.6*	28.66±1.1
FedProto	84.77±0.8	N/A	44.93±0.3*	N/A	80.45±2.9	N/A	37.06±1.0*	N/A
Ditto	88.00±1.2	N/A	53.66±2.5	N/A	83.02±2.0	N/A	45.22±2.2	N/A
FedBN	88.35±0.9	38.62±3.7*	48.57±0.9*	11.91±1.4*	83.76±1.8	45.97±4.1*	36.88±1.0*	10.52±1.0*
FloCo	87.48±1.4	79.57±7.4	44.33±2.3	27.16±1.7*	82.12±2.3	75.05±4.5	36.65±0.8*	25.96±1.3
kNN-Per	72.56±1.1*	71.45±1.8	37.94±0.2*	37.20±2.5	68.80±0.7	70.18±1.8	36.86±0.7*	34.85±1.1
pFedHN	85.95±0.2	24.51±6.8*	47.34±1.5*	3.16±0.1*	79.74±1.5	26.80±3.9*	35.57±2.3	3.58±1.4*
FedAGHN	85.18±1.2	11.39±4.4*	55.18±0.9*	1.59±0.8*	83.65±2.0	18.43±9.3	49.43±0.7	2.03±1.0*
PeFLL	87.88±1.0	75.41±4.3	51.30±0.8*	13.05±1.6*	82.10±2.5	69.94±5.4	39.35±1.2*	10.21±0.7*
FedGeo	**89.48**±0.8	**86.78**±3.7	**63.13**±0.7	**51.73**±2.4	**84.13**±1.7	**81.74**±2.4	**50.77**±1.1	**38.62**±2.1

Assumption 4.2 (Backbone Lipschitz continuity)**.** For fixed backbone parameters and each input x, the output is L_f-Lipschitz on the relevant set of concatenated FiLM vectors:

$$\|f_\theta(x; m_1) - f_\theta(x; m_2)\|_2 \le L_f \|m_1 - m_2\|_2. \tag{15}$$

Assumption 4.3 (Bounded class-conditional loss)**.** For each client j, class c, and conditioning prior d, the class-conditional expected loss is bounded by $B > 0$:

$$0 \le g_{j,c}^{(R)}(d) := \mathbb{E}_{x \sim P_j(X|Y=c)}[\ell(f_{\theta^{(R)}}(x; m^{(R)}(d)), c)] \le B. \tag{16}$$

The client family has class-conditional distributions for all classes considered in the comparison.

Theorem 4.4 (Generalization bound for unseen clients)**.** *For an unseen client u with prior d_u, let $k \in \mathcal{K}_{\text{train}}$ be the nearest training client and $\delta_{\min} = \|d_u - d_k\|_2$. Under Assumptions 4.1–4.3 and $\|W_{SN}^{(\tau)}\|_2 \le \kappa$ for every $\tau \in \mathcal{A}$, the expected risk satisfies*

$$\mathcal{L}_u^{(R)} \le \mathcal{L}_k^{(R)} + \left(B\sqrt{C} + \sqrt{2|\mathcal{A}|} L_\ell L_f \kappa\right) \delta_{\min} + \varepsilon_{\text{shift}}^{(R)}(u, k), \tag{17}$$

where $\mathcal{L}_j^{(R)} = \sum_c d_j[c] g_{j,c}^{(R)}(d_j)$ and

$$\varepsilon_{\text{shift}}^{(R)}(u, k) = \sum_c d_u[c] \left| g_{u,c}^{(R)}(d_k) - g_{k,c}^{(R)}(d_k) \right|. \tag{18}$$

The bound separates label mismatch, $B\sqrt{C}\delta_{\min}$, parameter sensitivity, $\sqrt{2|\mathcal{A}|} L_\ell L_f \kappa \delta_{\min}$, and conditional feature error, $\varepsilon_{\text{shift}}^{(R)}(u, k)$. The factor $\sqrt{|\mathcal{A}|}$ accounts for modulated layers. Under pure label shift, the conditional error is zero and transfer depends on the nearest training-prior distance. Appendix A gives the proof and empirical-prior analysis.

Convergence for unseen clients. Let $\mathcal{L}_j^* = \inf_{\theta,\phi,\psi} \mathcal{L}_j(\theta, \phi, \psi)$ over the parameter family under consideration.

Assumption 4.5 (Training risk and optimal-risk regularity)**.** For each training client k, client pair (i, j), and round $R \ge 1$, constants $C_{fed}, L_{opt} \ge 0$ satisfy

$$\mathbb{E}[\mathcal{L}_k^{(R)}] - \mathcal{L}_k^* \le C_{fed}/\sqrt{R}, \tag{19}$$

$$|\mathcal{L}_i^* - \mathcal{L}_j^*| \le L_{opt} \|d_i - d_j\|_2. \tag{20}$$

The expectation is over training randomness.

Table 2: Seen and unseen client accuracy (%) across four backbones on pathological CIFAR-100. OOM denotes out of memory.

	LeNet-5		ShuffleNetV2		ResNet-8		ResNet-18	
Method	Seen	Unseen	Seen	Unseen	Seen	Unseen	Seen	Unseen
FedAvg	28.69±0.5*	28.21±2.2*	18.86±0.3*	16.79±0.8*	35.49±0.7*	33.64±1.2*	27.23±1.4*	26.94±2.5
FedProto	44.93±0.3*	N/A	42.35±0.2*	N/A	50.83±0.5*	N/A	<u>46.10</u>±1.1	N/A
Ditto	53.66±2.5	N/A	14.65±0.4*	N/A	43.70±1.3*	N/A	44.54±2.8	N/A
FedBN	48.57±0.9*	11.91±1.4*	18.05±0.6*	15.97±0.9*	<u>62.56</u>±1.1	3.99±1.5*	39.07±2.1*	3.38±2.2*
FloCo	44.33±2.3	27.16±1.7*	37.64±0.5*	22.68±1.1*	49.85±1.5*	29.18±2.4*	42.25±2.4	28.50±3.1
kNN-Per	37.94±0.2*	<u>37.20</u>±2.5	29.73±0.2*	<u>25.66</u>±0.7*	49.19±0.6*	<u>47.25</u>±1.8	33.18±1.2*	<u>28.77</u>±2.9
pFedHN	47.34±1.5*	3.16±0.1*	40.93±1.1	2.70±0.8*	39.52±1.1*	2.77±0.1*	OOM	OOM
FedAGHN	<u>55.18</u>±0.9*	1.59±0.8*	<u>45.74</u>±1.2	2.03±0.9*	46.99±1.3*	2.64±1.1*	OOM	OOM
PeFLL	51.30±0.8*	13.05±1.6*	36.29±0.7*	5.07±1.2*	39.41±1.4*	2.43±1.0*	OOM	OOM
FedGeo	**63.13**±0.7	**51.73**±2.4	**52.91**±0.4	**39.20**±0.9	**69.75**±0.9	**57.07**±1.9	**57.03**±1.8	**42.51**±3.3

Theorem 4.6 (Conditional convergence transfer)**.** *Under Theorem 4.4 with constants uniform over training rounds and Assumption 4.5,*

$$\mathbb{E}[\mathcal{L}_u^{(R)}] - \mathcal{L}_u^* \leq \frac{C_{fed}}{\sqrt{R}} + C_{gen}\delta_{\min} + \mathbb{E}[\varepsilon_{\text{shift}}^{(R)}(u,k)], \tag{21}$$

where $C_{gen} = B\sqrt{C} + \sqrt{2|\mathcal{A}|}L_\ell L_f \kappa + L_{opt}$.

The bound transfers the $R^{-1/2}$ training-client rate with a residual from prior distance and conditional feature variation, reducing to $C_{gen}\delta_{\min}$ under pure label shift. The proof and effective-norm extension are in Appendix B.

5 Experiments

Following Related Work, we compare (i) fine-tuning-based personalization: Ditto, FedBN, FedProto, and FloCo (Grinwald et al., 2024), with FedAvg and kNN-Per (Marfoq et al., 2022) as reference baselines; and (ii) meta-learning-based generation: pFedHN, PeFLL, and FedAGHN.

5.1 Experimental Setup

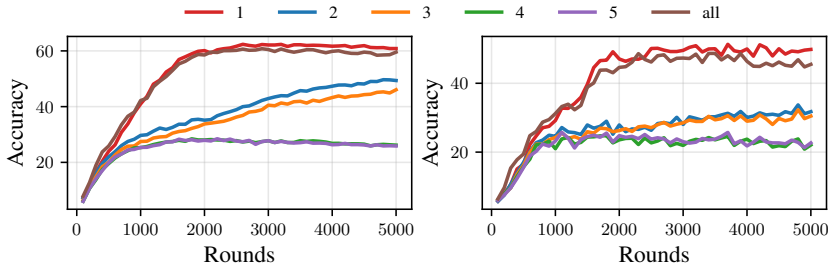
Data and client splits. We partition CIFAR-10/100 (Krizhevsky, 2009) among 100 clients: 90 seen and 10 held out. Each client's data are split 75%/25% into training/support and test sets; test labels are excluded from personalization. Only the support subset is used to estimate an unseen client's prior. Pathological splits assign 2 and 10 classes per client on CIFAR-10 and CIFAR-100, respectively. Practical splits use a Dirichlet distribution with $\alpha = 0.1$ (Li et al., 2021a).

Models and optimization. We use FL-bench (Tan & Wang, n.d.) and modified LeNet-5 (LeCun et al., 1998). All methods share partitions and train for 5,000 rounds with 20% client sampling, one local epoch, batch size 32, and stochastic gradient descent (SGD) at learning rate 0.01. FedGeo uses $D_{orth} = C$, $\kappa = 5$, and final-layer modulation; baselines use FL-bench defaults. Seeds are 42, 43, and 44. Unseen clients use direct inference without gradient updates. Implementation details are in Appendices E–G; adaptation with the same support set is in Appendix H.5.

Tables 1–5 report three-run means ± SD, with SD in smaller type. Bold/underline: best/second-best means. N/A: no native unseen-client model. *: Holm-adjusted $p < 0.05$ against the best; see Appendix I. Ablation and hyperparameter experiments have no significance tests.

5.2 Main Results

FedGeo attains the highest mean accuracy in all eight CIFAR comparisons in Table 1. Seen-client gains over the strongest baseline are 1.13% and 7.95% on pathological CIFAR-10/100, and 0.37%



(a) Seen Client (b) Unseen Client

Figure 3: Effect of modulation layers on CIFAR-100 pathological with LeNet-5.

and 1.34% on the practical splits. Unseen-client gains are larger: 7.21% and 14.53% on pathological splits, and 6.69% and 3.77% on practical splits.

On CIFAR-10, seen-to-unseen gaps are 2.70% and 2.39%, versus 12.47% and 12.16% for PeFLL. These smaller gaps indicate better transfer to unseen clients under both splits.

5.3 Generalization across Backbones

FedGeo achieves the highest mean seen and unseen accuracy with all four backbones in Table 2. Unseen gains over the strongest baseline are 14.53%, 13.54%, 9.82%, and 13.74% for LeNet-5, ShuffleNetV2 (Ma et al., 2018), ResNet-8, and ResNet-18 (He et al., 2016). ResNet-8 reaches 69.75% seen and 57.07% unseen accuracy. ResNet-18 reaches 57.03% and 42.51%, supporting FedGeo's use across feature extractors.

5.4 Ablation Studies

Table 3 compares MLPs and random Fourier features (RFF) (Rahimi & Recht, 2007) under the same backbone, partition, and schedule. An MLP encoder lowers unseen accuracy from 51.73% to 36.15%, a 15.58% drop after removing explicit geometry preservation. An MLP generator yields 42.58%, a 9.15% drop, supporting sensitivity control after encoding.

RFF reaches 44.47%; replacing both stages yields 26.13%, a 25.60% drop. The full model also leads in seen accuracy; visualizations are in Appendix H.4.

Table 3: Encoder/generator comparisons on CIFAR-100 pathological (%).

Encoder	Generator	Seen	Unseen
E_ϕ	h_ψ	**63.13**±0.7	**51.73**±2.4
E_ϕ	MLP	60.52±1.3	42.58±2.8
RFF	h_ψ	58.76±0.9	44.47±3.0
MLP	h_ψ	52.84±2.1	36.15±3.6
MLP	MLP	28.46±1.4	26.13±1.9

5.5 Sensitivity Analysis and Efficiency

Modulation layers. Figure 3 numbers layers backwards from the output. Final-layer and all-layer modulation give the strongest seen and unseen curves; earlier-layer-only modulation reaches lower accuracy. Their similar performance supports the final-layer default, which reduces generated parameters while conditioning the mapping from shared features to class scores.

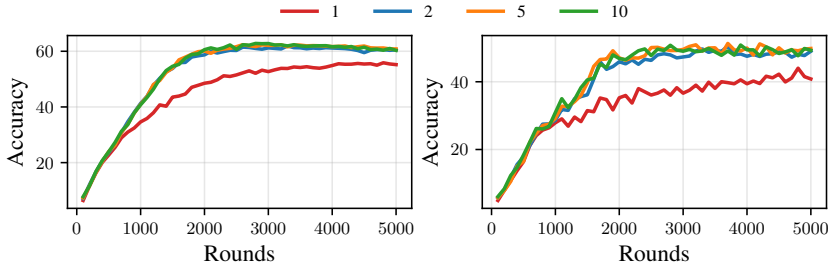
Encoder dimension. Table 4 varies D_{orth} from 100 to 1024. Seen accuracy spans 63.13–63.86%; unseen accuracy spans 51.73–53.52%, peaking at 256 with a 1.79% gain over the default dimension. Gains are not monotonic.

Table 4: Encoder dimension on CIFAR-100 pathological (%).

D_{orth}	Seen	Unseen
100	63.13±0.7	51.73±2.4
128	63.81±0.2	52.62±2.0
256	63.81±0.7	**53.52**±2.8
512	**63.86**±0.7	52.72±2.9
1024	<u>63.83</u>±0.4	<u>53.20</u>±3.2

The encoder parameter count $D_{orth} \times C$ and generator input size $C + D_{orth}$ grow with dimension, while the FiLM output size and factor-$\sqrt{2}$ isometry remain fixed. Thus $D_{orth} = C$ retains the same geometric guarantee at the smallest tested parameter cost.

Spectral threshold. Figure 4 shows similar final accuracy for $\kappa \in \{2, 5, 10\}$, while $\kappa = 1$ gives lower accuracy. Eq. 13 bounds one-layer sensitivity by $\sqrt{2}\kappa$ under the norm condition. The weaker $\kappa = 1$ curves suggest that overly strict control limits client-specific variation, supporting $\kappa = 5$ as a balanced default.



(a) Seen Client (b) Unseen Client

Figure 4: Sensitivity to κ on CIFAR-100 pathological.

Training efficiency. Figure 5 shows training times close to FedAvg. QR, spectral estimation, and FiLM generation operate on the client descriptor and selected channels; all examples in a mini-batch share the generated vectors. For fixed local data, each batch reuses cached class frequencies, recomputes QR, and performs one warm-started power iteration. Deployment generates the FiLM vectors once per client and reuses them across test examples without gradient updates. C10/C100 denote CIFAR-10/100, and Path/Prac denote the two splits. Further complexity details are provided in Appendix C.

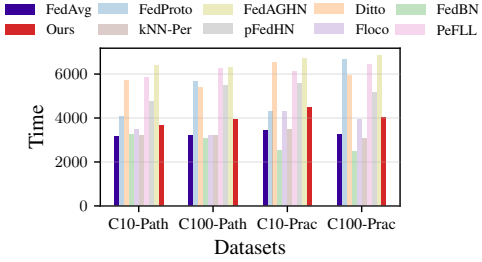


Figure 5: CIFAR-10/100 training time.

5.6 Evaluation under Natural Feature and Label Heterogeneity

FEMNIST (Caldas et al., 2018) has 3,597 naturally partitioned writers; 360 are held out. We use a 0.01 participation rate and three initializations on the same split. FedGeo achieves 89.98% seen and 88.49% unseen accuracy in Table 5. Its seen-client mean accuracy ranks second to PeFLL (91.49%). On unseen clients, FedGeo significantly outperforms pFedHN, FedAGHN, and PeFLL, with all Holm-adjusted $p < 0.05$ in Appendix I; PeFLL achieves 40.31%. FedGeo's unseen mean accuracy is the highest among the compared methods, 2.11% above FloCo; gains over FedAvg are 7.35% seen and 5.07% unseen. FedGeo's 1.49% seen-to-unseen gap, versus PeFLL's 51.18% gap, supports transfer to new writers. Both seen and unseen clients use local label priors with the fixed shared model for direct personalization without gradient updates, reusing the generated modulation across test samples.

Table 5: FEMNIST accuracy (%).

Method	Seen	Unseen
FedAvg	82.63±0.5*	83.42±0.8
FedProto	38.87±0.3*	N/A
Ditto	83.15±1.2	N/A
FedBN	87.88±0.9	54.64±2.1*
FloCo	86.07±1.1	<u>86.38</u>±1.4
kNN-Per	84.95±0.2	85.61±0.5
pFedHN	85.73±1.5	23.11±0.4*
FedAGHN	10.80±0.7*	0.70±0.2*
PeFLL	**91.49**±0.8	40.31±1.6*
FedGeo	<u>89.98</u>±0.4	**88.49**±0.6

6 Conclusion

FedGeo implements geometric stability through distance-preserving label-prior encoding and spectrally constrained FiLM generation. Both components are trained jointly with a shared backbone. Our generalization and conditional convergence bounds connect unseen-client risk to distributional proximity and training-client progress under the stated assumptions. Experiments demonstrate strong performance across label-skew benchmarks, backbones, and naturally partitioned writers. A new client estimates its prior from local support labels and generates modulation once; subsequent predictions reuse these parameters without local gradient updates. Future work will combine label priors and feature-level descriptors for direct personalization under broader feature shift.

AI USE STATEMENT

Generative AI tools were used solely for grammar checking and language polishing of the manuscript. They were not used in the design or conduct of experiments, analysis of results, or any other part of the research.

ETHICS STATEMENT

This work studies federated image classification using CIFAR-10/100 and FEMNIST. Raw examples remain on clients in the described protocol. Responsible deployment should assess privacy at the protocol level and protect model updates and client statistics. Application-specific evaluation should examine subgroup performance, data governance, and the consequences of prediction errors in the intended setting.

REPRODUCIBILITY STATEMENT

Section 4 specifies the encoder, generator, and theoretical assumptions. Appendices A and B provide the proofs. Algorithm 1 in Appendix D specifies training and direct inference; Appendices E–G give the reported optimization settings, data splits, and backbone specifications. Appendix H provides supplementary results. The implementation uses FL-bench (Tan & Wang, n.d.).

REFERENCES

Durmus Alp Emre Acar, Yue Zhao, Ramon Matas, Matthew Mattina, Paul Whatmough, and Venkatesh Saligrama. Federated learning based on dynamic regularization. In *Proc. ICLR*, 2021.

Sebastian Caldas, Sai Meher Karthik Duddu, Peter Wu, Tian Li, Jakub Konečný, H. Brendan McMahan, Virginia Smith, and Ameet Talwalkar. LEAF: A benchmark for federated settings. *arXiv preprint arXiv:1812.01097*, 2018.

Chen Chen, Yuchen Liu, Xingjun Ma, and Lingjuan Lyu. CalFAT: Calibrated federated adversarial training with label skewness. In *Proc. NeurIPS*, volume 35, pp. 3569–3581, 2022.

Liam Collins, Hamed Hassani, Aryan Mokhtari, and Sanjay Shakkottai. Exploiting shared representations for personalized federated learning. In *Proc. ICML*, volume 139 of *Proceedings of Machine Learning Research (PMLR)*, pp. 2089–2099. PMLR, 2021.

Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach. In *Proc. NeurIPS*, volume 33, pp. 3557–3568, 2020.

Dario Fenoglio, Mohan Li, Pietro Barbiero, Nicholas D. Lane, Marc Langheinrich, and Martin Gjoreski. FLUX: Efficient descriptor-driven clustered federated learning under arbitrary distribution shifts. In *Proc. NeurIPS*, volume 38, pp. 71229–71292, 2025.

Saeed Ghadimi and Guanghui Lan. Stochastic first- and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization*, 23(4):2341–2368, 2013.

Sajjad Ghiasvand, Mahnoosh Alizadeh, and Ramtin Pedarsani. pFedMMA: Personalized federated fine-tuning with multi-modal adapter for vision-language models. In *Proc. ICLR*, pp. 76213–76240, 2026.

Henry Gouk, Eibe Frank, Bernhard Pfahringer, and Michael J. Cree. Regularisation of neural networks by enforcing Lipschitz continuity. *Machine Learning*, 110(2):393–416, 2021.

Dennis Grinwald, Philipp Wiesner, and Shinichi Nakajima. Federated learning over connected modes. In *Proc. NeurIPS*, volume 37, pp. 85179–85202, 2024.

David Ha, Andrew M. Dai, and Quoc V. Le. HyperNetworks. In *Proc. ICLR*, 2017.

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

Sture Holm. A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics*, 6:65–70, 1979.

Guan Huang and Tao Shu. Federated oriented learning: A practical one-shot personalized federated learning framework. In *Proc. ICML*, volume 267 of *Proceedings of Machine Learning Research (PMLR)*, pp. 25762–25790. PMLR, 2025.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proc. ICML*, volume 37 of *Proceedings of Machine Learning Research (PMLR)*, pp. 448–456. PMLR, 2015.

Yihan Jiang, Jakub Konečný, Keith Rush, and Sreeram Kannan. Improving federated learning personalization via model agnostic meta learning. *arXiv preprint arXiv:1909.12488*, 2019.

Kaili Jin, Li Xu, Xiaoding Wang, Sun-Yuan Hsieh, Jie Wu, and Limei Lin. FedCPD: Personalized federated learning with prototype-enhanced representation and memory distillation. In *Proc. IJCAI*, volume 6, pp. 5498–5507, 2025.

Peter Kairouz, H. Brendan McMahan, Brendan Avent, Aurélien Bellet, et al. Advances and open problems in federated learning. *Foundations and Trends in Machine Learning*, 14(1–2):1–210, 2021.

Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. SCAFFOLD: Stochastic controlled averaging for federated learning. In *Proc. ICML*, volume 119 of *Proceedings of Machine Learning Research (PMLR)*, pp. 5132–5143. PMLR, 2020.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Mohan Li, Dario Fenoglio, Martin Gjoreski, and Marc Langheinrich. Federated learning with profile mapping under distribution shifts and drifts. In *Proc. ICLR*, pp. 72119–72170, 2026.

Qinbin Li, Bingsheng He, and Dawn Song. Model-contrastive federated learning. In *Proc. CVPR*, pp. 10713–10722, 2021a.

Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. In *Proc. MLSys*, volume 2, pp. 429–450, 2020a.

Tian Li, Shengyuan Hu, Ahmad Beirami, and Virginia Smith. Ditto: Fair and robust federated learning through personalization. In *Proc. ICML*, volume 139 of *Proceedings of Machine Learning Research (PMLR)*, pp. 6357–6368. PMLR, 2021b.

Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. On the convergence of FedAvg on non-IID data. In *Proc. ICLR*, 2020b.

Xiaoxiao Li, Meirui Jiang, Xiaofei Zhang, Michael Kamp, and Qi Dou. FedBN: Federated learning on non-IID features via local batch normalization. In *Proc. ICLR*, 2021c.

Zachary Lipton, Yu-Xiang Wang, and Alexander Smola. Detecting and correcting for label shift with black box predictors. In *Proc. ICML*, volume 80 of *Proceedings of Machine Learning Research (PMLR)*, pp. 3122–3130. PMLR, 2018.

Zili Lu, Heng Pan, Yueyue Dai, Xueming Si, and Yan Zhang. Federated learning with non-IID data: A survey. *IEEE Internet of Things Journal*, 11(11):19188–19209, 2024.

Jun Luo and Shandong Wu. FedSLD: Federated learning with shared label distribution for medical image classification. In *Proc. ISBI*, pp. 1–5, 2022.

Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. ShuffleNet V2: Practical guidelines for efficient CNN architecture design. In *Proc. ECCV*, pp. 116–131, 2018.

Yanbiao Ma, Wei Dai, Gaoyang Jiang, Wanyi Chen, Chenyue Zhou, Yiwei Zhang, Fei Luo, Junhao Wang, and Andi Zhang. FedMC: Federated manifold calibration. In *Proc. ICLR*, pp. 60055–60075, 2026.

Yishay Mansour, Mehryar Mohri, Jae Ro, and Ananda Theertha Suresh. Three approaches for personalization with applications to federated learning. *arXiv preprint arXiv:2002.10619*, 2020.

Othmane Marfoq, Giovanni Neglia, Richard Vidal, and Laetitia Kameni. Personalized federated learning through local memorization. In *Proc. ICML*, volume 162 of *Proceedings of Machine Learning Research (PMLR)*, pp. 15070–15092. PMLR, 2022.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Proc. AISTATS*, volume 54 of *Proceedings of Machine Learning Research (PMLR)*, pp. 1273–1282. PMLR, 2017.

Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *Proc. ICLR*, 2018.

Dinh C. Nguyen, Ming Ding, Pubudu N. Pathirana, Aruna Seneviratne, Jun Li, and H. Vincent Poor. Federated learning for Internet of Things: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, 23(3):1622–1658, 2021.

Ziru Niu, Hai Dong, and A. K. Qin. FedSPU: Personalized federated learning for resource-constrained devices with stochastic parameter update. In *Proc. AAAI*, volume 39, pp. 19721–19729, 2025.

Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron Courville. FiLM: Visual reasoning with a general conditioning layer. In *Proc. AAAI*, volume 32, 2018.

Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Proc. NeurIPS*, volume 20, 2007.

Jonathan Scott, Hossein Zakerinia, and Christoph H. Lampert. PeFLL: Personalized federated learning by learning to learn. In *Proc. ICLR*, pp. 32406–32436, 2024.

Aviv Shamsian, Aviv Navon, Ethan Fetaya, and Gal Chechik. Personalized federated learning using hypernetworks. In *Proc. ICML*, volume 139 of *Proceedings of Machine Learning Research (PMLR)*, pp. 9489–9502. PMLR, 2021.

Jiarui Song, Yunheng Shen, Chengbin Hou, Pengyu Wang, Jinbao Wang, Ke Tang, and Hairong Lv. FedAGHN: Personalized federated learning with attentive graph hypernetworks. *Knowledge-Based Systems*, 329:114355, 2025.

Alysa Ziying Tan, Han Yu, Lizhen Cui, and Qiang Yang. Towards personalized federated learning. *IEEE Transactions on Neural Networks and Learning Systems*, 34(12):9587–9603, 2023.

Jiahao Tan and Xinpeng Wang. FL-bench: A federated learning benchmark for solving image classification tasks. Software repository, n.d. URL `https://github.com/KarhouTam/FL-bench`. Accessed September 25, 2026.

Yue Tan, Guodong Long, Lu Liu, Tianyi Zhou, Qinghua Lu, Jing Jiang, and Chengqi Zhang. FedProto: Federated prototype learning across heterogeneous clients. In *Proc. AAAI*, volume 36, pp. 8432–8440, 2022.

Dipanwita Thakur, Antonella Guzzo, Giancarlo Fortino, and Francesco Piccialli. Green federated learning: A new era of green aware AI. *ACM Computing Surveys*, 57(8):1–36, 2025.

B. L. Welch. The generalization of 'Student's' problem when several different population variances are involved. *Biometrika*, 34(1–2):28–35, 1947. doi: 10.1093/biomet/34.1-2.28.

Jian Xu, Xinyi Tong, and Shao-Lun Huang. Personalized federated learning with feature alignment and classifier collaboration. In *Proc. ICLR*, 2023.

Tian Yang, Xinhui Yu, Martin J. McKeown, and Z. Jane Wang. When federated learning meets medical image analysis: A systematic review with challenges and solutions. *APSIPA Transactions on Signal and Information Processing*, 13(1):1–55, 2024.

Rui Ye, Mingkai Xu, Jianyu Wang, Chenxin Xu, Siheng Chen, and Yanfeng Wang. FedDisco: Federated learning with discrepancy-aware collaboration. In *Proc. ICML*, volume 202 of *Proceedings of Machine Learning Research (PMLR)*, pp. 39879–39902. PMLR, 2023.

Xin Yu, Zelin He, Ying Sun, Lingzhou Xue, and Runze Li. Understanding the statistical accuracy-communication trade-off in personalized federated learning with minimax guarantees. In *Proc. ICML*, volume 267 of *Proceedings of Machine Learning Research (PMLR)*, pp. 73009–73050. PMLR, 2025.

Chen Zhang, Husheng Li, Xiang Liu, Linshan Jiang, and Danxin Wang. Revisiting hypernetwork in model heterogeneous personalized federated learning. In *Proc. IJCAI*, pp. 5288–5296, 2026. doi: 10.24963/ijcai.2026/589.

Jie Zhang, Zhiqi Li, Bo Li, Jianghe Xu, Shuang Wu, Shouhong Ding, and Chao Wu. Federated learning with label distribution skew via logits calibration. In *Proc. ICML*, volume 162 of *Proceedings of Machine Learning Research (PMLR)*, pp. 26311–26329. PMLR, 2022.

Yue Zhao, Meng Li, Liangzhen Lai, Naveen Suda, Damon Civin, and Vikas Chandra. Federated learning with non-IID data. *arXiv preprint arXiv:1806.00582*, 2018.

Yuhao Zhou, Jindi Lv, Yuxin Tian, Dan Si, Qing Ye, and Jiancheng Lv. Deploying models to non-participating clients in federated learning without fine-tuning: A hypernetwork-based approach. In *Proc. ICLR*, pp. 63600–63621, 2026.

Zhuangdi Zhu, Junyuan Hong, and Jiayu Zhou. Data-free knowledge distillation for heterogeneous federated learning. In *Proc. ICML*, volume 139 of *Proceedings of Machine Learning Research (PMLR)*, pp. 12878–12889. PMLR, 2021.

A Proof of the Generalization Bound

For fixed learned parameters satisfying $\|W_{SN}^{(\tau)}\|_2 \le \kappa$ for every $\tau \in \mathcal{A}$, write $m(d) = h_\psi(E_\phi(d))$ for the concatenated FiLM vectors and define

$$\mathcal{R}_j(d) = \mathbb{E}_{(x,y)\sim P_j}[\ell(f_\theta(x; m(d)), y)]. \tag{22}$$

Thus $\mathcal{L}_j = \mathcal{R}_j(d_j)$ when $d_j = P_j(Y)$. Assumptions 4.1–4.3 in Section 4.3 apply uniformly over the compared clients and conditioning priors.

Proof of Theorem 4.4.

Proof. Suppress the round superscript for clarity. For each adaptable layer,

$$\|m^{(\tau)}(d_u) - m^{(\tau)}(d_k)\|_2 = \|W_{SN}^{(\tau)}(E_\phi(d_u) - E_\phi(d_k))\|_2 \le \sqrt{2}\,\kappa\|d_u - d_k\|_2. \tag{23}$$

The concatenated modulation vector contains the outputs of all adaptable layers. Its squared distance is therefore

$$\|m(d_u) - m(d_k)\|_2^2 = \sum_{\tau\in\mathcal{A}} \|m^{(\tau)}(d_u) - m^{(\tau)}(d_k)\|_2^2 \le 2|\mathcal{A}|\kappa^2\|d_u - d_k\|_2^2. \tag{24}$$

Taking square roots establishes Eq. 13, with $L_m = \sqrt{2|\mathcal{A}|}\,\kappa$. For a fixed data distribution P_j, changing only the conditioning prior gives

$$\begin{aligned} |\mathcal{R}_j(d) - \mathcal{R}_j(d')| &\le \mathbb{E}_{(x,y)\sim P_j}\big|\ell(f_\theta(x; m(d)), y) - \ell(f_\theta(x; m(d')), y)\big| \\ &\le L_\ell L_f \|m(d) - m(d')\|_2 \\ &\le L_\ell L_f L_m \|d - d'\|_2. \end{aligned} \tag{25}$$

The first comparison below changes the conditioning prior on client u; the second keeps the conditioning prior fixed and changes the client data distribution:

$$\begin{aligned} \mathcal{L}_u - \mathcal{L}_k &= \mathcal{R}_u(d_u) - \mathcal{R}_u(d_k) + \mathcal{R}_u(d_k) - \mathcal{R}_k(d_k) \\ &\le L_\ell L_f L_m \|d_u - d_k\|_2 + |\mathcal{R}_u(d_k) - \mathcal{R}_k(d_k)|. \end{aligned} \tag{26}$$

To bound the second term, expand both risks by class. Adding and subtracting $\sum_c d_u[c] g_{k,c}(d_k)$ yields the exact decomposition

$$\mathcal{R}_u(d_k) - \mathcal{R}_k(d_k) = \sum_c (d_u[c] - d_k[c]) g_{k,c}(d_k) + \sum_c d_u[c]\big(g_{u,c}(d_k) - g_{k,c}(d_k)\big). \tag{27}$$

The first sum accounts for changing class proportions, and the second for changing the class-conditionals while using the same generated parameters. Assumption 4.3 and the triangle inequality give

$$\begin{aligned} |\mathcal{R}_u(d_k) - \mathcal{R}_k(d_k)| &= \left|\sum_c d_u[c] g_{u,c}(d_k) - \sum_c d_k[c] g_{k,c}(d_k)\right| \\ &\le B\|d_u - d_k\|_1 + \sum_c d_u[c]\,|g_{u,c}(d_k) - g_{k,c}(d_k)|. \end{aligned} \tag{28}$$

Define

$$\varepsilon_{\text{shift}}(u,k) = \sum_c d_u[c]\,|g_{u,c}(d_k) - g_{k,c}(d_k)|. \tag{29}$$

Using $\|v\|_1 \le \sqrt{C}\|v\|_2$ and $\delta_{\min} = \|d_u - d_k\|_2$ in Eq. 26 yields

$$\mathcal{L}_u \le \mathcal{L}_k + (B\sqrt{C} + L_\ell L_f L_m)\delta_{\min} + \varepsilon_{\text{shift}}(u,k). \tag{30}$$

Restoring the round superscript proves the claim. Under pure label shift, the compared class-conditionals agree, so $\varepsilon_{\text{shift}}(u,k) = 0$. □

Estimated priors. The implementation conditions on empirical frequencies $\hat{d}_j$. Let $e_j = \|\hat{d}_j - d_j\|_2$. The same stability argument gives

$$|\mathcal{R}_j(\hat{d}_j) - \mathcal{R}_j(d_j)| \le L_\ell L_f L_m e_j. \tag{31}$$

Applying this inequality to both clients in Theorem 4.4 gives

$$\mathcal{R}_u(\hat{d}_u) \le \mathcal{R}_k(\hat{d}_k) + (B\sqrt{C} + L_\ell L_f L_m)\delta_{\min} + \varepsilon_{\text{shift}}(u,k) + L_\ell L_f L_m(e_u + e_k). \tag{32}$$

For $m \ge 1$ i.i.d. labeled support examples with prior d, the empirical histogram has covariance $(\operatorname{diag}(d) - dd^\top)/m$. Its trace gives

$$\mathbb{E}\|\hat{d} - d\|_2^2 = \frac{1 - \|d\|_2^2}{m}, \qquad \mathbb{E}\|\hat{d} - d\|_2 \le \sqrt{\frac{1 - \|d\|_2^2}{m}} \le m^{-1/2}. \tag{33}$$

Jensen's inequality gives the first inequality. Each class frequency averages Bernoulli indicators with variance $d[c](1 - d[c])/m$; summing the diagonal variances gives the mean-square error. At fixed support size, a single-class prior has zero histogram error, while less concentrated priors have larger variance.

For the same fixed trained mapping, empirical priors add conditioning error $L_\ell L_f L_m(e_u + e_k)$. The bound thus separates training-client coverage, generator sensitivity, and prior-estimation error.

Practical Spectral Control. Each training forward pass updates the retained unit vectors once by Eq. 10, estimates $\widetilde{\sigma}(W^{(\tau)}) = (\widetilde{u}^{(\tau)})^\top W^{(\tau)} \widetilde{v}^{(\tau)}$, and rescales by Eq. 11. The updates require nonzero normalization denominators. A zero matrix returns the zero map; a nonzero matrix requires nondegenerate initialization or a restart if a denominator vanishes.

The sensitivity of a fixed implemented generator is determined by its effective operator norm,

$$\kappa_{\text{eff}} = \max_{\tau \in \mathcal{A}} \|W_{SN}^{(\tau)}\|_2 = \max_{\tau \in \mathcal{A}} \min\left(1, \frac{\kappa}{\widetilde{\sigma}(W^{(\tau)})}\right) \|W^{(\tau)}\|_2. \tag{34}$$

For fixed trained weights, the parameter-sensitivity and pointwise risk bounds use κ_{eff}. Exact spectral-norm rescaling gives $\kappa_{\text{eff}} \le \kappa$; finite-iteration rescaling is characterized by the effective norm above.

B Conditional Convergence Transfer

We prove Theorem 4.6 under Assumption 4.5. The nearest client k is selected by population-prior distance, with ties resolved independently of training randomness. Prior analyses study strongly convex FedAvg objectives (Li et al., 2020b) and nonconvex stationarity (Ghadimi & Lan, 2013; Karimireddy et al., 2020); the stated per-client excess-risk premise specifies the optimization rate transferred here.

Proof. Insert the nearest training client's current and optimal risks:

$$\mathbb{E}[\mathcal{L}_u^{(R)}] - \mathcal{L}_u^* = \mathbb{E}[\mathcal{L}_u^{(R)} - \mathcal{L}_k^{(R)}] + \mathbb{E}[\mathcal{L}_k^{(R)}] - \mathcal{L}_k^* + (\mathcal{L}_k^* - \mathcal{L}_u^*). \tag{35}$$

Theorem 4.4 bounds the first term by

$$\mathbb{E}[\mathcal{L}_u^{(R)} - \mathcal{L}_k^{(R)}] \le \left(B\sqrt{C} + \sqrt{2|\mathcal{A}|} L_\ell L_f \kappa\right) \delta_{\min} + \mathbb{E}[\varepsilon_{\text{shift}}^{(R)}(u,k)]. \tag{36}$$

The distance $\delta_{\min}$ is fixed by the population priors. The deterministic constants can therefore be taken outside the expectation. The two conditions in Assumption 4.5 bound the remaining terms:

$$\mathbb{E}[\mathcal{L}_k^{(R)}] - \mathcal{L}_k^* \le C_{fed}/\sqrt{R}, \tag{37}$$

$$\mathcal{L}_k^* - \mathcal{L}_u^* \le |\mathcal{L}_k^* - \mathcal{L}_u^*| \le L_{opt}\delta_{\min}. \tag{38}$$

Substituting all three bounds into Eq. 35 and collecting the coefficients of $\delta_{\min}$ gives $C_{gen} = B\sqrt{C} + \sqrt{2|\mathcal{A}|} L_\ell L_f \kappa + L_{opt}$, which proves the claim.

□

Effective-norm form. If the implemented generator has random effective norm $\kappa_{\text{eff}}^{(R)}$, the same decomposition yields

$$\begin{aligned}\mathbb{E}[\mathcal{L}_u^{(R)}] - \mathcal{L}_u^* \le \frac{C_{fed}}{\sqrt{R}} + (B\sqrt{C} + L_{opt})\delta_{\min} \\ + \sqrt{2|\mathcal{A}|}L_\ell L_f\, \delta_{\min}\, \mathbb{E}[\kappa_{\text{eff}}^{(R)}] + \mathbb{E}[\varepsilon_{\text{shift}}^{(R)}(u,k)].\end{aligned} \tag{39}$$

Here the remaining constants are deterministic and uniform over training randomness. A uniform deterministic bound $\kappa_{\text{eff}}^{(R)} \le \kappa$ recovers the stated theorem.

Under the uniform norm condition in Theorem 4.6, the transferred bound has a decaying $R^{-1/2}$ term and a distribution-dependent residual. In particular,

$$\limsup_{R\to\infty} \mathbb{E}[\mathcal{L}_u^{(R)}] \le \mathcal{L}_u^* + C_{gen}\delta_{\min} + \limsup_{R\to\infty} \mathbb{E}[\varepsilon_{\text{shift}}^{(R)}(u,k)]. \tag{40}$$

The limit superior characterizes asymptotic risk through an upper envelope of the training trajectory. If $\delta_{\min} = 0$ and the conditional error vanishes, the finite-round bound reduces to the assumed training-client rate. For a nonzero prior distance, the bound explicitly retains the corresponding transfer term. This analysis complements personalization bounds (Mansour et al., 2020), gradient-based adaptation (Jiang et al., 2019; Fallah et al., 2020), and future-client generalization in PeFLL (Scott et al., 2024) by exposing the role of prior-space distance and generator sensitivity.

C Complexity Analysis

Let $n_k = |\mathcal{D}_k^{\text{train}}|$, n_u be the number of labeled support examples, $n_u^{test} = |\mathcal{D}_u^{\text{test}}|$, and b be the batch size. Then $N_b = E\lceil n_k/b \rceil$ is the number of local mini-batches. Write F_θ and B_θ for per-example backbone forward and backward costs, including FiLM application. Define

$$P_G = \sum_{\tau\in\mathcal{A}} 2D_\tau(C + D_{orth}), \qquad |\Gamma| = D_{orth}C + P_G + \sum_{\tau\in\mathcal{A}} 2D_\tau. \tag{41}$$

The trainable parameter count comprises the encoder matrix, generator weights, and generator biases.

Local training. Class counting costs $O(n_k + C)$ and can be cached for a fixed local dataset. Each batch recomputes a reduced QR decomposition at cost $O(D_{orth}C^2)$, forms the encoded prior at cost $O(D_{orth}C)$, and performs power iteration and affine generation at cost $O((E_{power}+1)P_G)$. The same client prior is shared by the examples in a batch. Including differentiation through these operations gives the per-client, per-round cost

$$T_{\text{FedGeo}} = O(n_k + C) + O\big(En_k(F_\theta + B_\theta)\big) + O\big(N_b\left[D_{orth}C^2 + (E_{power}+1)P_G\right]\big). \tag{42}$$

We use $E_{power} = 1$. Cached class counts incur the first term only once across rounds. The QR bound assumes full-column-rank M and $D_{orth} \ge C$, including differentiation. Encoding and generation occur once per mini-batch; the backbone cost has the FedAvg form, with FiLM included in F_θ and B_θ.

Component-wise batch cost. Reduced QR and its backward pass cost $O(D_{orth}C^2)$, and forming Qd_k costs $O(D_{orth}C)$. Each power iteration multiplies by the generator matrix and its transpose; affine generation multiplies by the encoded prior. Summing over adaptable layers gives

$$T_{\text{batch,aux}} = O(D_{orth}C^2) + O(D_{orth}C) + O(E_{power}P_G) + O(P_G). \tag{43}$$

Since $C \ge 1$, QR covers the $O(D_{orth}C)$ term. Backpropagation accumulates batch gradients for the shared modulation vectors before propagating them through the generator and encoder, giving N_b times the auxiliary batch cost.

Parameter and output dimensions. Writing $D_{gen} = 2\sum_{\tau\in\mathcal{A}} D_\tau$, the same parameter count can be expressed as

$$P_G = D_{gen}(C + D_{orth}), \qquad |\Gamma| = D_{orth}C + D_{gen}(C + D_{orth} + 1). \tag{44}$$

The first term counts encoder parameters; the remainder counts generator weights and biases. More adaptable channels increase D_{gen}, while increasing D_{orth} enlarges the encoder and generator input without changing the FiLM output size.

Table 6: Reported complexity values for different backbone architectures on CIFAR-100 with $D_{orth} = 100$.

Backbone	$\lvert\theta\rvert$	$\mathcal{F}_\theta$	$\lvert\Gamma\rvert$	$\mathcal{F}_\Gamma$
LeNet-5	7.7×10^4	4.2×10^6	3.4×10^4	3.5×10^4
ResNet-8	9.0×10^4	4.1×10^7	2.3×10^4	2.4×10^4
ShuffleNetV2	1.4×10^6	1.5×10^8	3.0×10^4	3.1×10^4
ResNet-18	1.1×10^7	5.6×10^8	3.2×10^4	3.3×10^4
Hypernetwork overhead ratios:				
LeNet-5	$\lvert\Gamma\rvert/\lvert\theta\rvert = 0.44$		$\mathcal{F}_\Gamma/\mathcal{F}_\theta = 8.3 \times 10^{-3}$	
ResNet-8	$\lvert\Gamma\rvert/\lvert\theta\rvert = 0.26$		$\mathcal{F}_\Gamma/\mathcal{F}_\theta = 5.9 \times 10^{-4}$	
ShuffleNetV2	$\lvert\Gamma\rvert/\lvert\theta\rvert = 2.1 \times 10^{-2}$		$\mathcal{F}_\Gamma/\mathcal{F}_\theta = 2.1 \times 10^{-4}$	
ResNet-18	$\lvert\Gamma\rvert/\lvert\theta\rvert = 2.9 \times 10^{-3}$		$\mathcal{F}_\Gamma/\mathcal{F}_\theta = 5.9 \times 10^{-5}$	

Inference. With the trained shared parameters fixed, QR and spectral rescaling can be performed once and cached. A new client counts n_u support labels, encodes its prior, and generates its FiLM vectors. Its total setup-plus-prediction cost is

$$I_{\text{FedGeo}} = O(n_u + C + D_{orth}C + P_G) + O(n_u^{test} F_\theta). \tag{45}$$

If model-dependent preprocessing is performed on the device, add $O(D_{orth}C^2 + E_{power}P_G)$. No backward pass, optimizer state, or local gradient update is needed during this deployment procedure.

Setup and amortization. Clients share the frozen model's QR map and effective generator weights. Each client counts support labels and generates modulation once, reusing it for all n_u^{test} predictions. For $n_u^{test} > 0$, the amortized per-example cost is

$$\frac{I_{\text{FedGeo}}}{n_u^{test}} = O(F_\theta) + O\left(\frac{n_u + C + D_{orth}C + P_G}{n_u^{test}}\right), \tag{46}$$

plus model preprocessing if performed locally. The setup cost per prediction decreases as more test examples use the same prior and shared model.

Comparison with adaptation and full-model generation. For E_{adapt} local epochs on n_u support examples, adaptation followed by prediction has total cost

$$I_{\text{adapt}} = O(E_{adapt} n_u (F_\theta + B_\theta)) + O(n_u^{test} F_\theta). \tag{47}$$

A data-to-model hypernetwork such as PeFLL incurs descriptor-extraction cost $T_{desc}(n_u)$ and parameter-generation cost T_{hyp} before prediction:

$$I_{\text{full-gen}} = T_{desc}(n_u) + T_{hyp} + O(n_u^{test} F_\theta). \tag{48}$$

The setup costs depend on the descriptor network, generated output, and support-processing procedure. Full-model generation outputs the backbone parameters, while FedGeo outputs D_{gen} FiLM values. These expressions separate client-specific setup, local adaptation, and repeated prediction costs.

Backbone and Generator Measurements. Table 6 lists the reported backbone and generator parameter and operation counts for CIFAR-100, together with their overhead ratios.

D Training and Direct Personalization Algorithm

Algorithm 1 details Section 4. The trainable state $\Theta = (\theta, \phi, \psi)$ comprises the backbone, encoder matrix M, and affine generator $\{W^{(\tau)}, b^{(\tau)}\}_{\tau \in \mathcal{A}}$. Auxiliary spectral-estimation vectors warm-start normalization as specified in Appendix A.

Algorithm 1 FedGeo: complete federated training and direct personalization

Require: Training clients $\mathcal{K}_{\text{train}}$ and data $\{\mathcal{D}_k^{\text{train}}\}$; rounds R; local epochs E; learning rate η; threshold κ; adaptable layers $\mathcal{A}$; unit spectral-estimation vectors; unseen-client support $\mathcal{D}_u^{\text{train}}$ and test inputs

Initialize shared parameters $\Theta^{(0)} = (\theta^{(0)}, \phi^{(0)}, \psi^{(0)})$, with encoder matrix M of full column rank.
for communication round $r = 0, \dots, R-1$ **do**
  Sample participating clients $\mathcal{S}^{(r)} \subseteq \mathcal{K}_{\text{train}}$.
  Broadcast $\Theta^{(r)}$ to every selected client.
  for each selected client $k \in \mathcal{S}^{(r)}$ **do**
    Initialize the local parameters $\Theta_k \leftarrow \Theta^{(r)}$.
    Count classes in $\mathcal{D}_k^{\text{train}}$ and compute the prior d_k by Eq. 4.
    for local epoch $e = 1, \dots, E$ **do**
      for mini-batch $\mathcal{B} \subseteq \mathcal{D}_k^{\text{train}}$ **do**
        Compute the current reduced QR decomposition $M_k = Q_k R_k$.
        Encode the client: $z_k \leftarrow [d_k \,\|\, Q_k d_k]$.
        for each adaptable layer $\tau \in \mathcal{A}$ **do**
          $\widetilde{v}_k^{(\tau)} \leftarrow (W_k^{(\tau)})^\top u_k^{(\tau)} / \|(W_k^{(\tau)})^\top u_k^{(\tau)}\|_2$.
          $\widetilde{u}_k^{(\tau)} \leftarrow W_k^{(\tau)} \widetilde{v}_k^{(\tau)} / \|W_k^{(\tau)} \widetilde{v}_k^{(\tau)}\|_2$.
          Estimate $\widetilde{\sigma}_k^{(\tau)} \leftarrow (\widetilde{u}_k^{(\tau)})^\top W_k^{(\tau)} \widetilde{v}_k^{(\tau)}$.
          Set $W_{SN,k}^{(\tau)} \leftarrow \min(1, \kappa / \widetilde{\sigma}_k^{(\tau)}) W_k^{(\tau)}$.
          Generate $[\gamma_k^{(\tau)}; \beta_k^{(\tau)}] \leftarrow W_{SN,k}^{(\tau)} z_k + b_k^{(\tau)}$.
          Retain $(u_k^{(\tau)}, v_k^{(\tau)}) \leftarrow (\widetilde{u}_k^{(\tau)}, \widetilde{v}_k^{(\tau)})$ for the next estimate.
        end for
        Form $m_k = \text{concat}_{\tau \in \mathcal{A}} [\gamma_k^{(\tau)}; \beta_k^{(\tau)}]$.
        Predict $\hat{y}_k(x) = f_{\theta_k}(x; m_k)$ for the batch, applying FiLM via Eq. 8.
        Compute $\widehat{\mathcal{L}}_{k,\mathcal{B}} = |\mathcal{B}|^{-1} \sum_{(x,y) \in \mathcal{B}} \ell(\hat{y}_k(x), y)$.
        Backpropagate $g_k \leftarrow \nabla_{\Theta_k} \widehat{\mathcal{L}}_{k,\mathcal{B}}$ through all trainable components, including QR.
        Update the local model jointly: $\Theta_k \leftarrow \Theta_k - \eta g_k$.
      end for
    end for
    Return the model update $\Delta_k \leftarrow \Theta_k - \Theta^{(r)}$ to the server.
  end for
  Set aggregation weights $a_k^{(r)} \leftarrow n_k / \sum_{j \in \mathcal{S}^{(r)}} n_j$.
  Aggregate $\Theta^{(r+1)} \leftarrow \Theta^{(r)} + \sum_{k \in \mathcal{S}^{(r)}} a_k^{(r)} \Delta_k$.
end for
Direct personalization: freeze the trained shared parameters $\Theta^{(R)}$.
Compute or retrieve the fixed $Q^{(R)}$ and $\{W_{SN}^{(\tau,R)}\}$ using Eqs. 5, 10, and 11.
Estimate the unseen client's prior d_u from labeled support $\mathcal{D}_u^{\text{train}}$ by Eq. 4.
Encode $z_u \leftarrow [d_u \,\|\, Q^{(R)} d_u]$.
for each adaptable layer $\tau \in \mathcal{A}$ **do**
  Generate $[\gamma_u^{(\tau)}; \beta_u^{(\tau)}] \leftarrow W_{SN}^{(\tau,R)} z_u + b^{(\tau,R)}$.
end for
Form $m_u = \text{concat}_{\tau \in \mathcal{A}} [\gamma_u^{(\tau)}; \beta_u^{(\tau)}]$ once.
for each test input x from $\mathcal{D}_u^{\text{test}}$ **do**
  Predict $f_{\theta^{(R)}}(x; m_u)$, reusing m_u without a backward pass or parameter update.
end for

Ensure: Shared parameters $\Theta^{(R)}$ and direct personalized predictions.

D.1 Federated Training Workflow

The training process consists of R communication rounds. At the beginning of round r, the server broadcasts $\Theta^{(r)}$ to a sampled subset $\mathcal{S}^{(r)} \subseteq \mathcal{K}_{\text{train}}$. Each selected client initializes its local copy Θ_k

from this shared state and performs E local epochs. The prior is computed from local training labels; raw samples and their class-frequency vector remain on the client.

Phase 1: geometry-preserving encoding. Client k counts the labels in $\mathcal{D}_k^{\text{train}}$ and normalizes these counts to form d_k using Eq. 4. The prior is cached across batches and rounds for a fixed local dataset. Each mini-batch recomputes the reduced QR decomposition of the current encoder weight M_k and constructs $z_k = [d_k \,\|\, Q_k d_k]$. Gradients propagate through QR to update M_k jointly with the backbone and generator.

Phase 2: spectrally constrained affine generation. For each adaptable layer, one power iteration updates the unit vectors and spectral-norm estimate, followed by Eq. 11. The affine map generates scale and shift vectors shared across the mini-batch. The backbone applies $x_{\text{mod}}^{(\tau)} = (1+\gamma_k^{(\tau)}) \odot x^{(\tau)} + \beta_k^{(\tau)}$, broadcasting channel-wise vectors over convolutional spatial positions. Spectral estimation follows the normalization-domain conventions in Appendix A.

Joint local optimization. With the generated modulation denoted by $m_k = h_{\psi_k}(E_{\phi_k}(d_k))$, the batch objective is

$$\widehat{\mathcal{L}}_{k,\mathcal{B}}(\Theta_k) = \frac{1}{|\mathcal{B}|} \sum_{(x,y)\in\mathcal{B}} \ell(f_{\theta_k}(x; m_k), y)\,. \tag{49}$$

A backward pass updates all three trainable components jointly. If s indexes local mini-batch steps, the update is

$$\Theta_{k,s+1} = \Theta_{k,s} - \eta \nabla_{\Theta_{k,s}} \widehat{\mathcal{L}}_{k,\mathcal{B}}(\Theta_{k,s}). \tag{50}$$

The updated $\Theta_{k,s+1}$ generates the modulation shared by the next mini-batch.

Server aggregation. After E local epochs, client k returns $\Delta_k = \Theta_k - \Theta^{(r)}$. The server uses the selected clients' training-set sizes to form

$$a_k^{(r)} = \frac{n_k}{\sum_{j\in\mathcal{S}^{(r)}} n_j}, \qquad \Theta^{(r+1)} = \Theta^{(r)} + \sum_{k\in\mathcal{S}^{(r)}} a_k^{(r)} \Delta_k. \tag{51}$$

Because the weights sum to one, this is equivalent to the sample-weighted average $\Theta^{(r+1)} = \sum_{k\in\mathcal{S}^{(r)}} a_k^{(r)} \Theta_k$. The aggregated parameters initialize the next round.

D.2 Direct Inference for an Unseen Client

After training, $\Theta^{(R)}$ remains fixed. For the standard labeled-support protocol, the unseen client computes d_u from its nonempty support set $\mathcal{D}_u^{\text{train}}$, encodes it as $z_u = E_{\phi^{(R)}}(d_u)$, and generates

$$m_u = h_{\psi^{(R)}}(z_u) = \text{concat}_{\tau\in\mathcal{A}} \left(W_{SN}^{(\tau,R)} z_u + b^{(\tau,R)} \right). \tag{52}$$

The fixed QR map and rescaled weights can be precomputed. Each client generates modulation once and reuses it in $f_{\theta^{(R)}}(x; m_u)$, with no test labels, backward pass, or optimizer update. Seen clients use the same procedure. Appendix H.5 describes the uniform-prior setting.

E Implementation Details

E.1 Hardware and Software Environment

Experiments use Python 3.12, PyTorch 2.8.0, and FL-bench (Tan & Wang, n.d.) on an NVIDIA GeForce RTX 5090 with 32GB memory.

E.2 Training Hyperparameters

Table 7 gives the common optimization settings. Experiment-specific client sampling and sensitivity settings are stated with the corresponding results.

Table 7: Training hyperparameters for all methods.

Parameter	Value
Total communication rounds (R)	5,000
Client sampling rate	20%
Local epochs per round (E)	1
Batch size (B)	32
Optimizer	SGD
Learning rate (η)	0.01
Momentum	0.0
Weight decay	0.0
Evaluation frequency	Every 100 rounds
Random seeds	{42, 43, 44}

Table 8: CIFAR dataset statistics before client partitioning.

Attribute	CIFAR-10	CIFAR-100
Number of Classes (C)	10	100
Input Resolution	$32 \times 32 \times 3$	$32 \times 32 \times 3$
Global Training Samples	50,000	50,000
Global Test Samples	10,000	10,000
Samples per Class	6,000	600

Table 9: Comparison of model parameters on CIFAR-10 and CIFAR-100 datasets.

Model	CIFAR-10	CIFAR-100
LeNet-5	69,700	77,350
ResNet-8	83,892	89,742
ShuffleNetV2	1,356,104	1,448,354
ResNet-18	11,227,812	11,273,982

E.3 FedGeo Configuration

The encoder uses $M \in \mathbb{R}^{D_{orth} \times C}$ with $D_{orth} = C$, giving representation dimension $2C$: $z_k \in \mathbb{R}^{20}$ for CIFAR-10 and $z_k \in \mathbb{R}^{200}$ for CIFAR-100.

Unless varied in the corresponding sensitivity experiment, the spectral threshold is $\kappa = 5.0$ and $\mathcal{A}$ contains only the final classification layer of each backbone.

The generator h_ψ, encoder E_ϕ, and backbone f_θ are jointly optimized using the SGD configuration in Table 7.

E.4 Baseline Method Configurations

For all baseline methods including FedAvg (McMahan et al., 2017), Ditto (Li et al., 2021b), FedBN (Li et al., 2021c), pFedHN (Shamsian et al., 2021), kNN-Per (Marfoq et al., 2022), FedProto (Tan et al., 2022), PeFLL (Scott et al., 2024), FloCo (Grinwald et al., 2024), and FedAGHN (Song et al., 2025), we directly adopt the default hyperparameter configurations provided in the FL-bench repository (Tan & Wang, n.d.). These implementations use their method-specific defaults under the common data partition and evaluation protocol.

F Dataset Specifications

CIFAR-10 and CIFAR-100 (Krizhevsky, 2009) each contain 60,000 color images, with 10 and 100 classes, respectively. Table 8 gives the global statistics before client partitioning.

G Model Architectures

In this section, we provide detailed specifications for the backbone architectures utilized in our experiments: LeNet-5, ResNet-8, ResNet-18, and ShuffleNetV2. We conduct analyses on the CIFAR family datasets. For all models, the input tensors are fixed at $3 \times 32 \times 32$, and the final output dimension is denoted as C_d (e.g., $C_d = 10$ for CIFAR-10, $C_d = 100$ for CIFAR-100).

Table 10: Architecture of the modified LeNet-5.

Layer	Output Shape	Configuration
Input	$3 \times 32 \times 32$	-
Conv1	$6 \times 28 \times 28$	5×5, 6, stride 1, no pad
BN1 + ReLU	$6 \times 28 \times 28$	BatchNorm2d, ReLU
MaxPool1	$6 \times 14 \times 14$	2×2, stride 2
Conv2	$16 \times 10 \times 10$	5×5, 16, stride 1, no pad
BN2 + ReLU	$16 \times 10 \times 10$	BatchNorm2d, ReLU
MaxPool2	$16 \times 5 \times 5$	2×2, stride 2
Flatten	400	Vectorize
FC1	120	Linear($400 \to 120$), ReLU
FC2	84	Linear($120 \to 84$), ReLU
Classifier	C_d	Linear($84 \to C_d$)

Table 11: Architecture of ResNet-8.

Stage	Output Shape	Structure
Input	$3 \times 32 \times 32$	-
Initial Conv	$16 \times 32 \times 32$	3×3 Conv, BN, ReLU
Layer 1	$16 \times 32 \times 32$	$\begin{bmatrix} 3 \times 3, 16 \\ 3 \times 3, 16 \end{bmatrix} \times 1$ (Basic Block)
Layer 2	$32 \times 16 \times 16$	$\begin{bmatrix} 3 \times 3, 32 \\ 3 \times 3, 32 \end{bmatrix} \times 1$ (Stride 2)
Layer 3	$64 \times 8 \times 8$	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 1$ (Stride 2)
AvgPool	$64 \times 1 \times 1$	Global Average Pooling
Classifier	C_d	Linear($64 \to C_d$)

G.1 Complexity Overview

Table 9 summarizes parameter counts for the four backbones on CIFAR-10 and CIFAR-100.

G.2 LeNet-5

We adopt a modified version of the LeNet-5 (LeCun et al., 1998) architecture. To improve training stability, we integrate Batch Normalization (BN) layers (Ioffe & Szegedy, 2015) following each convolutional layer. The specific layer-wise configuration and tensor shapes are detailed in Table 10.

G.3 ResNet-8

ResNet-8 (He et al., 2016) is a compact residual network with a 3×3 input convolution and three Basic Block stages with 16, 32, and 64 channels. Table 11 gives the layer configuration.

G.4 ResNet-18

We utilize the standard ResNet-18 (He et al., 2016) architecture provided by the `torchvision` library. The network begins with a 7×7 convolution (stride 2) followed by a 3×3 max pooling layer (stride 2). Following the stem layers, the features are processed through four stages of residual blocks. The detailed structure applied to the input tensors is provided in Table 12.

G.5 ShuffleNetV2 (1.0x)

We employ the ShuffleNetV2 (1.0x) (Ma et al., 2018). This model uses channel splitting, depthwise and pointwise convolutions, and channel shuffling. The network consists of a convolutional stem

Table 12: Architecture of ResNet-18.

Stage	Output Shape	Structure
Input	$3 \times 32 \times 32$	-
Stem (Conv1)	$64 \times 16 \times 16$	$7 \times 7, 64$, stride 2, pad 3
Stem (Pool)	$64 \times 8 \times 8$	3×3 MaxPool, stride 2, pad 1
Layer 1	$64 \times 8 \times 8$	Basic Block $\times 2$
Layer 2	$128 \times 4 \times 4$	Basic Block $\times 2$ (Stride 2)
Layer 3	$256 \times 2 \times 2$	Basic Block $\times 2$ (Stride 2)
Layer 4	$512 \times 1 \times 1$	Basic Block $\times 2$ (Stride 2)
AvgPool	$512 \times 1 \times 1$	Global Average Pooling
Classifier	C_d	Linear($512 \to C_d$)

Table 13: Architecture of ShuffleNetV2 (1.0x).

Stage	Output Shape	Structure
Input	$3 \times 32 \times 32$	-
Conv1	$24 \times 16 \times 16$	$3 \times 3, 24$, stride 2
MaxPool	$24 \times 8 \times 8$	3×3, stride 2
Stage 2	$116 \times 4 \times 4$	ShuffleUnit $\times 4$ (Stride 2 start)
Stage 3	$232 \times 2 \times 2$	ShuffleUnit $\times 8$ (Stride 2 start)
Stage 4	$464 \times 1 \times 1$	ShuffleUnit $\times 4$ (Stride 2 start)
Conv5	$1024 \times 1 \times 1$	$1 \times 1, 1024$
AvgPool	$1024 \times 1 \times 1$	Global Average Pooling
Classifier	C_d	Linear($1024 \to C_d$)

followed by three stages of stacked Shuffle Units. The detailed architecture configuration is shown in Table 13.

H Additional Experiments

We provide sensitivity analyses and supplementary deployment experiments.

H.1 Prior Perturbations

Protocol and results. We add scaled noise to d_k and project the perturbed vector onto the probability simplex. The trained model remains fixed throughout this evaluation. Table 14 reports point accuracies for each noise scale. At scale 0.0010, seen and unseen accuracy fall to 60.49% and 47.33%, drops of 2.64% and 4.40% from the unperturbed prior; at intermediate scales, unseen accuracy ranges from 49.09% to 50.30%. The larger unseen loss highlights the importance of accurate local priors for direct personalization. The simplex projection maintains nonnegative, normalized class proportions.

Table 14: Prior perturbations on CIFAR-100 pathological (%).

Noise scale	Seen	Unseen
0	63.13	51.73
0.0001	63.32	49.90
0.0003	62.90	50.30
0.0005	61.97	49.09
0.0010	60.49	47.33

H.2 Client Participation Rate

Figure 6 varies the fraction of clients sampled per round on pathological CIFAR-100. Higher participation reaches high accuracy in fewer communication rounds. The final curves cluster around 60% on seen clients and 50% on unseen clients. Similar final levels with fewer rounds suggest that greater client participation mainly accelerates optimization by supplying more client information per round.

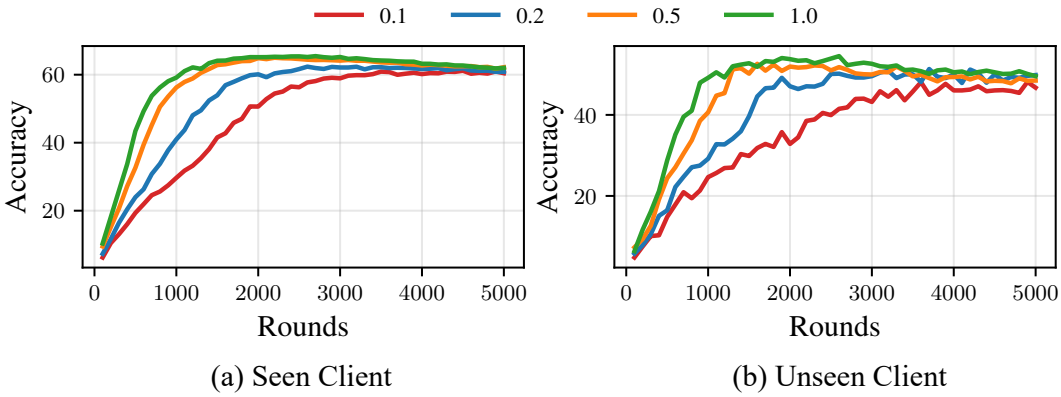


Figure 6: Client participation on CIFAR-100 pathological.

H.3 Number of Clients

Figure 7 varies the total number of clients from 50 to 500 while keeping the global dataset fixed. Seen accuracy is lower at 500 clients than at 50 clients, decreasing from about 62% to about 50%. The strongest unseen-client curve occurs near 200 clients and is lower at 500 clients. The pattern is consistent with a trade-off between broader coverage of client priors and fewer examples per client; this experiment varies both factors together.

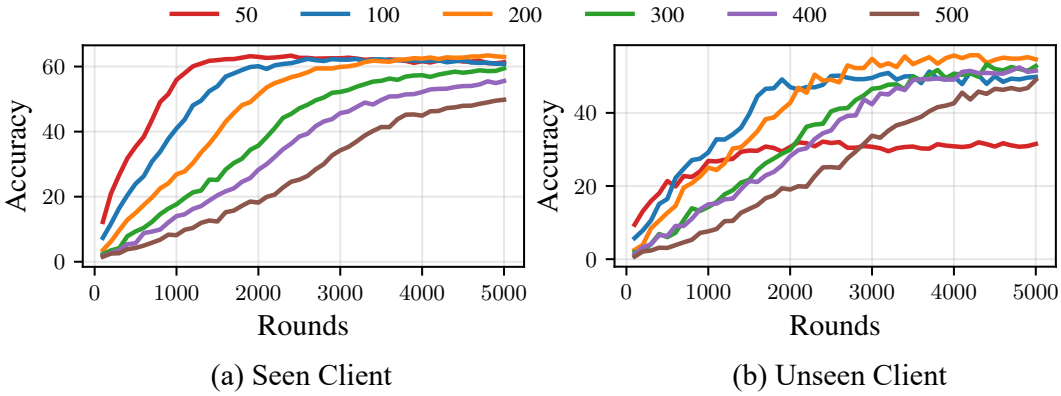


Figure 7: Impact of total client number. Test accuracy on CIFAR-100 with varying total clients K.

H.4 Visualization of Geometric Preservation

We sample 40 clients from the pathological CIFAR-100 partition and project the priors, encoded representations, and generated FiLM vectors to two dimensions using principal component analysis (PCA). Clients are color-coded by distributional similarity and linked to their nearest neighbors. Figure 8 illustrates the geometry-preserving encoding: Eq. 7 establishes exact distance preservation up to the fixed factor $\sqrt{2}$ in the full representation space. The generator applies an affine transformation whose amplification of representation distances is bounded by its effective spectral norm.

Figure 9 applies the same projection and neighbor-linking protocol to an MLP encoder–generator. Its projected configurations differ more across stages. These visualizations provide a qualitative comparison of the representations produced by the two component designs.

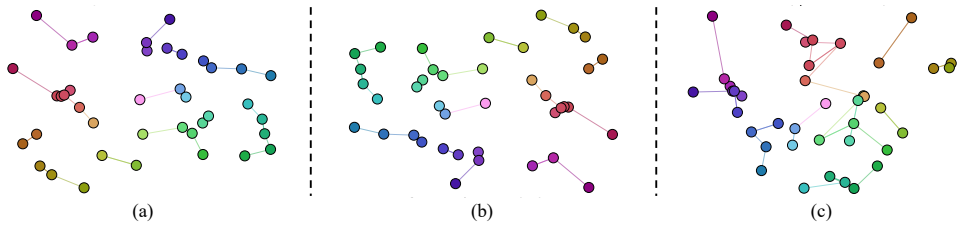


Figure 8: Visualization of geometric preservation in FedGeo. (a) Raw label distributions d_k of 40 sampled clients. (b) Encoded representations $z_k = E_\phi(d_k)$ after the Geometry-Preserving Encoder. (c) Generated FiLM parameters $\{\gamma_k, \beta_k\} = h_\psi(z_k)$ from the Spectrally Constrained Affine Generator. Each client is connected to its nearest neighbor, and color-coded by distributional similarity. All visualizations use PCA for 2D projection.

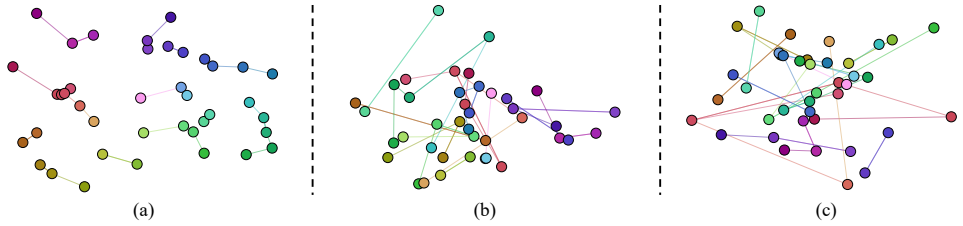


Figure 9: PCA visualization of MLP-based components. (a) Raw label distributions d_k of 40 sampled clients. (b) Encoded representations from the MLP encoder. (c) Generated FiLM parameters from the MLP generator. Each client is connected to its nearest neighbor, and color-coded by distributional similarity. All visualizations use PCA for 2D projection.

H.5 Labeled Support and Optional Local Adaptation

Table 15 compares six methods using the same labeled support for direct inference and one epoch of full-model fine-tuning. FedGeo uses support labels to estimate its prior; fine-tuning updates all model layers.
FedGeo's direct accuracy, 51.24%, exceeds the highest adapted baseline, pFedHN at 45.97%, by 5.27%. One local epoch of full-model fine-tuning raises FedGeo to 53.96% (+2.72%). This additional gain shows that local optimization can refine the model already personalized through its prior.

Table 15: Supplementary unseen-client accuracy (%) with and without one epoch of full-model fine-tuning. Values are point estimates from this supplementary evaluation.

Method	No fine-tuning	One local epoch
FedProto	N/A	6.03
Ditto	N/A	5.96
pFedHN	2.10	45.97
FedAGHN	1.08	4.87
PeFLL	9.48	40.28
FedGeo	**51.24**	**53.96**

Uniform-prior fallback. In a supplementary pathological CIFAR evaluation, a uniform prior with no labeled support yields 54.13% and 28.64% unseen accuracy on CIFAR-10 and CIFAR-100; full support yields 84.59% and 50.85%. The corresponding gains from labeled support are 30.46% and 22.21%, highlighting the value of client-distribution information for generation with the same encoder and generator.

I Significance Analysis from Summary Statistics

I.1 Test Specification

We analyze the rounded three-run summaries in Tables 1, 2, and 5 under their reported partitions and direct-inference protocol. Each observation is one run's accuracy for the specified client group. The

Table 16: Unadjusted FEMNIST p-values under the reported SD convention, conversion from divisor n to $n-1$, and last-digit rounding.

Group	Baseline	Reported SD	Converted SD	Rounding range
Seen	PeFLL	0.0628	0.0987	[0.0511, 0.0764]
Unseen	FloCo	0.1051	0.1547	[0.0952, 0.1155]
Unseen	PeFLL	<0.0001	0.0001	$[4.71, 10.9] \times 10^{-5}$

reference model assumes independent, approximately normal run-level accuracies and interprets the reported deviations as sample SDs.

For FedGeo and baseline B, let $\bar{a}_F, \bar{a}_B$ denote mean accuracies, s_F, s_B their SDs, and $n_F = n_B = 3$. We use the two-sided Welch test (Welch, 1947) for $H_0 : \mu_F = \mu_B$, allowing unequal variances. With $v_F = s_F^2/n_F$, $v_B = s_B^2/n_B$, and $\Delta = \bar{a}_F - \bar{a}_B$, the statistic and degrees of freedom are

$$t_W = \frac{\Delta}{\sqrt{v_F + v_B}}, \qquad \nu = \frac{(v_F + v_B)^2}{v_F^2/(n_F - 1) + v_B^2/(n_B - 1)}. \tag{53}$$

Writing F_{t_ν} for the Student-t cumulative distribution function, the two-sided p-value and pointwise 95% CI are

$$p = 2\big[1 - F_{t_\nu}(|t_W|)\big], \qquad \mathrm{CI}_{95\%} = \Delta \pm t_{\nu,0.975}\sqrt{v_F + v_B}. \tag{54}$$

Positive Δ favors FedGeo; differences and intervals are in %.

The $m = 130$ unique tests comprise 64 CIFAR main-result, 42 additional-backbone, 16 FEMNIST FedGeo–baseline, and eight additional FEMNIST Seen comparisons with PeFLL. Repeated LeNet-5 results are counted once; N/A and `OOM` entries are excluded. Holm's correction (Holm, 1979) at level 0.05 uses the ordered values $p_{(1)} \le \cdots \le p_{(m)}$:

$$p_{\mathrm{H},(i)} = \min\left\{1, \max_{1 \le j \le i}(m - j + 1)p_{(j)}\right\}. \tag{55}$$

Stars in Tables 1, 2, and 5 mark $p_{\mathrm{H}} < 0.05$ against the column best. FEMNIST uses PeFLL as the Seen reference and FedGeo as the Unseen reference. CIs are pointwise, without multiplicity adjustment. Non-rejection of the mean-difference null does not establish equivalence.

I.2 FEMNIST Comparisons

FedGeo's seen mean is 1.51% below PeFLL, with $p = 0.0628$, $p_{\mathrm{H}} = 1.0000$, and a 95% CI of $[-3.17, 0.15]$%. On unseen clients, its gains over pFedHN, FedAGHN, and PeFLL are 65.38%, 87.79%, and 48.18%; adjusted p-values are 9.92×10^{-6}, 2.54×10^{-4}, and 0.0065, respectively. The corresponding CIs are $[64.15, 66.61]$%, $[86.46, 89.12]$%, and $[44.70, 51.66]$%. The 2.11% gain over FloCo has $p = 0.1051$, $p_{\mathrm{H}} = 1.0000$, and a CI of $[-0.87, 5.09]$%. All comparisons appear in Table 17.

I.3 Sensitivity to Summary Conventions

Table 16 evaluates two features of the reported summaries. If the deviations use divisor n rather than $n-1$, their sample SDs are $s\sqrt{n/(n-1)}$. For rounding, each mean varies by half a unit of its last displayed digit and each SD by the corresponding half-unit interval. The p-value ranges use a 201×201 grid over the two deviation intervals and the extreme absolute mean differences. These are numerical sensitivity ranges, not CIs.

The seen-client comparison with PeFLL remains above 0.05 under either SD convention and over the rounding range. The unseen comparisons with pFedHN, FedAGHN, and PeFLL remain below 0.05 after recomputing all 130 tests and Holm adjustments under the divisor-n convention; their adjusted values are 2.01×10^{-5}, 4.17×10^{-4}, and 0.0111, respectively.

A paired model additionally depends on the within-pair correlation ρ, which the marginal summaries do not specify. Its difference variance and test statistic are

$$s_\Delta^2 = s_F^2 + s_B^2 - 2\rho s_F s_B, \qquad t_{\mathrm{paired}} = \frac{\Delta\sqrt{3}}{s_\Delta}, \quad \nu_{\mathrm{paired}} = 2. \tag{56}$$

For $\rho \in [-1, 1]$, the hypothetical two-sided paired p-value ranges are $[0.0226, 0.1611]$ for seen PeFLL, $[0.0447, 0.2092]$ for unseen FloCo, and $[0.000144, 0.000694]$ for unseen PeFLL. These ranges vary the covariance model; the reported significance decisions use the independent-sample Welch model in Appendix I.1.

I.4 Complete Comparison Results

Table 17 reports all comparisons by setting and reference method. A dash denotes N/A or `OOM`; p-values below 10^{-4} are shown as < 0.0001.

Table 17: Complete summary-statistic significance results. Δ denotes reference minus baseline accuracy in %; the reference is FedGeo except in the final PeFLL block. p is unadjusted and p_{H} is Holm-adjusted over all 130 comparisons. Bold adjusted values satisfy $p_{\mathrm{H}} < 0.05$.

	Seen			Unseen		
Baseline	Δ	p	p_{H}	Δ	p	p_{H}
CIFAR-10 / Pathological / LeNet-5						
FedAvg	+32.57	<0.0001	**0.0020**	+29.04	0.0033	0.1346
FedProto	+4.71	0.0020	0.0942	—	—	—
Ditto	+1.48	0.1606	1.0000	—	—	—
FedBN	+1.13	0.1804	1.0000	+48.16	<0.0001	**0.0077**
FloCo	+2.00	0.1158	1.0000	+7.21	0.2301	1.0000
kNN-Per	+16.92	<0.0001	**0.0051**	+15.33	0.0084	0.2350
pFedHN	+3.53	0.0127	0.3169	+62.27	0.0007	**0.0400**
FedAGHN	+4.30	0.0096	0.2538	+75.39	<0.0001	**0.0029**
PeFLL	+1.60	0.0997	1.0000	+11.37	0.0265	0.5822
CIFAR-100 / Pathological / LeNet-5						
FedAvg	+34.44	<0.0001	**0.0001**	+23.52	0.0002	**0.0177**
FedProto	+18.20	<0.0001	**0.0063**	—	—	—
Ditto	+9.47	0.0167	0.3917	—	—	—
FedBN	+14.56	<0.0001	**0.0039**	+39.82	<0.0001	**0.0075**
FloCo	+18.80	0.0027	0.1181	+24.57	0.0003	**0.0178**
kNN-Per	+25.19	<0.0001	**0.0077**	+14.53	0.0019	0.0942
pFedHN	+15.79	0.0007	**0.0400**	+48.57	0.0008	**0.0447**
FedAGHN	+7.95	0.0004	**0.0244**	+50.14	0.0002	**0.0177**
PeFLL	+11.83	<0.0001	**0.0047**	+38.68	<0.0001	**0.0055**
CIFAR-10 / Practical / LeNet-5						
FedAvg	+27.45	<0.0001	**0.0060**	+19.58	0.0005	**0.0324**
FedProto	+3.68	0.1477	1.0000	—	—	—
Ditto	+1.11	0.5055	1.0000	—	—	—
FedBN	+0.37	0.8086	1.0000	+35.77	0.0007	**0.0400**
FloCo	+2.01	0.2957	1.0000	+6.69	0.1062	1.0000
kNN-Per	+15.33	0.0013	0.0680	+11.56	0.0034	0.1346
pFedHN	+4.39	0.0291	0.6120	+54.94	0.0001	**0.0097**
FedAGHN	+0.48	0.7677	1.0000	+63.31	0.0048	0.1715
PeFLL	+2.03	0.3175	1.0000	+11.80	0.0462	0.7860
CIFAR-100 / Practical / LeNet-5						
FedAvg	+20.48	<0.0001	**0.0067**	+9.96	0.0052	0.1733
FedProto	+13.71	<0.0001	**0.0078**	—	—	—
Ditto	+5.55	0.0308	0.6168	—	—	—
FedBN	+13.89	<0.0001	**0.0077**	+28.10	0.0003	**0.0215**
FloCo	+14.12	0.0001	**0.0086**	+12.66	0.0020	0.0942
kNN-Per	+13.91	0.0002	**0.0121**	+3.77	0.0699	1.0000
pFedHN	+15.20	0.0023	0.1055	+35.04	<0.0001	**0.0050**
FedAGHN	+1.34	0.1624	1.0000	+36.59	0.0002	**0.0116**

Continued on the next page.

Table 17 (continued).

	Seen			Unseen		
Baseline	Δ	p	p_{H}	Δ	p	p_{H}
CIFAR-100 / Practical / LeNet-5 (continued)						
PeFLL	+11.42	0.0003	**0.0187**	+28.41	0.0007	**0.0403**
CIFAR-100 / Pathological / ShuffleNetV2						
FedAvg	+34.05	<0.0001	**<0.0001**	+22.41	<0.0001	**0.0007**
FedProto	+10.56	<0.0001	**0.0038**	—	—	—
Ditto	+38.26	<0.0001	**<0.0001**	—	—	—
FedBN	+34.86	<0.0001	**<0.0001**	+23.23	<0.0001	**0.0007**
FloCo	+15.27	<0.0001	**0.0004**	+16.52	<0.0001	**0.0047**
kNN-Per	+23.18	<0.0001	**0.0004**	+13.54	<0.0001	**0.0049**
pFedHN	+11.98	0.0010	0.0569	+36.50	<0.0001	**0.0001**
FedAGHN	+7.17	0.0051	0.1733	+37.17	<0.0001	**0.0001**
PeFLL	+16.62	<0.0001	**0.0031**	+34.13	<0.0001	**0.0006**
CIFAR-100 / Pathological / ResNet-8						
FedAvg	+34.26	<0.0001	**0.0002**	+23.43	0.0002	**0.0129**
FedProto	+18.92	<0.0001	**0.0047**	—	—	—
Ditto	+26.05	<0.0001	**0.0026**	—	—	—
FedBN	+7.19	0.0011	0.0589	+53.08	<0.0001	**0.0006**
FloCo	+19.90	0.0002	**0.0122**	+27.89	0.0001	**0.0103**
kNN-Per	+20.56	<0.0001	**0.0020**	+9.82	0.0029	0.1256
pFedHN	+30.23	<0.0001	**0.0006**	+54.30	0.0004	**0.0253**
FedAGHN	+22.76	<0.0001	**0.0040**	+54.43	<0.0001	**0.0017**
PeFLL	+30.34	<0.0001	**0.0026**	+54.64	<0.0001	**0.0025**
CIFAR-100 / Pathological / ResNet-18						
FedAvg	+29.80	<0.0001	**0.0037**	+15.57	0.0037	0.1352
FedProto	+10.93	0.0020	0.0942	—	—	—
Ditto	+12.49	0.0049	0.1726	—	—	—
FedBN	+17.96	0.0004	**0.0254**	+39.13	0.0002	**0.0129**
FloCo	+14.78	0.0014	0.0718	+14.01	0.0059	0.1832
kNN-Per	+23.85	0.0001	**0.0095**	+13.74	0.0059	0.1832
FEMNIST / Natural writers						
FedAvg	+7.35	<0.0001	**0.0050**	+5.07	0.0013	0.0675
FedProto	+51.11	<0.0001	**<0.0001**	—	—	—
Ditto	+6.83	0.0057	0.1832	—	—	—
FedBN	+2.10	0.0396	0.7125	+33.85	0.0006	**0.0365**
FloCo	+3.91	0.0163	0.3917	+2.11	0.1051	1.0000
kNN-Per	+5.03	0.0003	**0.0220**	+2.88	0.0034	0.1346
pFedHN	+4.25	0.0320	0.6168	+65.38	<0.0001	**<0.0001**
FedAGHN	+79.18	<0.0001	**<0.0001**	+87.79	<0.0001	**0.0003**
PeFLL	-1.51	0.0628	1.0000	+48.18	<0.0001	**0.0065**
FEMNIST / Natural writers / PeFLL reference (Seen)						
FedAvg	+8.86	0.0003	**0.0180**	—	—	—
FedProto	+52.62	<0.0001	**0.0011**	—	—	—
Ditto	+8.34	0.0011	0.0587	—	—	—
FedBN	+3.61	0.0068	0.1971	—	—	—
FloCo	+5.42	0.0032	0.1346	—	—	—
kNN-Per	+6.54	0.0032	0.1346	—	—	—
pFedHN	+5.76	0.0094	0.2538	—	—	—
FedAGHN	+80.69	<0.0001	**<0.0001**	—	—	—