RETHINKING HETEROGENEOUS FEDERATED LEARNING: LOW-RANK CONTINUOUS FEATURE REPRESENTATIONS FOR KNOWLEDGE SHARING

Anonymous authors
Paper under double-blind review

ABSTRACT

Real-world federated learning deployments are often model-heterogeneous, rendering parameter averaging ineffective. Existing methods mainly adopt two mechanisms: one introduces homogeneous proxy models for knowledge aggregation and distillation, but incurs high computation and communication costs; the other shares class prototypes, which are lightweight but compress intra-class structure and are prone to feature collapse and semantic mismatch. Therefore, existing methods exhibit a clear expressiveness–efficiency gap. We propose FedCF, which leverages lightweight yet expressive low-rank continuous feature fields to support heterogeneous collaboration, preventing feature collapse while maintaining low overhead. The server aligns and fuses heterogeneous local fields in a canonical function space to form a queryable global field, providing clients with rich feature supervision to mitigate collapse. Local training further adopts feature–logit dual distillation to jointly align representations and calibrate decision boundaries, without proxy models or public data, keeping the overhead low. Across heterogeneous benchmarks, FedCF achieves a better accuracy–efficiency trade-off.

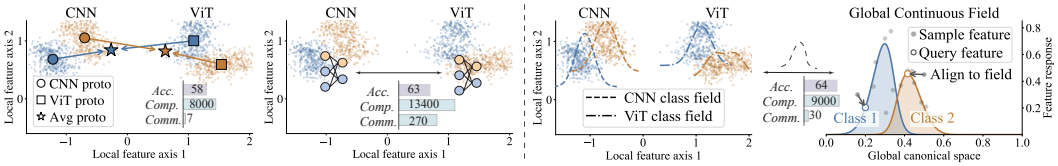
1 INTRODUCTION

Federated learning (FL) supports privacy-preserving collaborative training (McMahan et al., 2017), and has been widely used in applications such as medical imaging (Kuo et al., 2025). However, in practice, clients differ in data distributions, computational capacity, and hardware configurations, leading to heterogeneity in both local data and models (Ye et al., 2023; Huang et al., 2024). While personalized FL can alleviate data heterogeneity (Li et al., 2021; Yu et al., 2025; Tran & Hoang, 2026), parameter aggregation is no longer meaningful under model heterogeneity. Therefore, the key problem shifts to *what kind of knowledge representation can serve as a shared carrier for effective collaboration across heterogeneous clients?*

To tackle model heterogeneity, heterogeneous federated learning (HtFL) has emerged. While submodel schemes (Wu et al., 2024; Ilhan et al., 2023) train compatible subnetworks under a shared backbone, they are difficult to extend to settings with highly diverse architectures. Accordingly, existing HtFL studies mainly develop cross-architecture communicable knowledge representations along two directions. (1) Proxy models: clients or the server rely on proxy data to train a homogeneous auxiliary model and distill its supervision back to heterogeneous clients (Wu et al., 2022; Yi et al., 2024; Zhang et al., 2025; Yi et al., 2026). (2) Prototype sharing: clients share class-wise prototypes or prototype-derived representations, which are aggregated or transformed into global knowledge (Tan et al., 2022; Zhang et al., 2024; Zhou et al., 2025; Meng et al., 2026; Yao et al., 2026).

Both carriers face fundamental bottlenecks. Proxy-model methods require additional homogeneous modules and proxy data, increasing computation and communication (Figure 1b). Prototype methods are lightweight but rely on discrete class-level representations that lose rich intra-class structure (Figure 1a); under severe heterogeneity, alignment or aggregation across heterogeneous feature spaces can cause feature collapse and semantic mismatch. Therefore, existing collaboration carriers exhibit an expressiveness–efficiency gap: discrete prototypes are lightweight but structurally coarse, whereas proxy models provide richer supervision at substantially higher cost.

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.



(a) Prototype-based method. (b) Proxy model-based method. (c) FedCF.

Figure 1: Knowledge representations in heterogeneous federated learning (Blue: Class 1, Orange: Class 2). (a) Prototypes compress each class into one vector, prone to collapse under mismatch. (b) Proxy models absorb knowledge from all data, but with high communication and computation cost. (c) FedCF shares lightweight yet expressive low-rank continuous feature fields, and learns a queryable global field for efficient heterogeneous collaboration.

To bridge this gap, we replace discrete class-level representations with a queryable continuous feature function (Chen et al., 2021; Luo et al., 2024), preserving rich intra-class structure beyond a single prototype (Figure 1c). We further introduce a low-rank parameterization to keep this continuous representation compact and efficient during collaboration. Based on this principle, we propose FedCF, a lightweight continuous-function collaboration framework for HtFL.

Specifically, each client learns a continuous feature field for each class, mapping arbitrary coordinates to expected class features to preserve rich intra-class structure rather than compressing each class into a single prototype; low-rank parameterization further compresses this continuous representation for low communication overhead. To handle heterogeneous feature spaces, lightweight translators map local fields into a shared canonical space, where the server fuses them into global fields without directly averaging incompatible representations. The continuous fields are then queried by coordinate and class to train a lightweight global classifier without proxy data, with the global fields and classifier providing feature- and decision-level shared knowledge, respectively.

Local training adopts feature–logit dual distillation: (i) feature alignment aligns translated local spatial features with global canonical features to facilitate cross-model feature knowledge sharing; and (ii) classifier distillation drives local predictions toward the global classifier's decision boundary to calibrate local boundaries. Thus, FedCF enables efficient collaboration among heterogeneous models in a continuous function space. Our contributions are:

- **Low-rank continuous knowledge carrier.** We propose low-rank continuous feature fields as the shared carrier for HtFL. Continuity recovers rich intra-class structure lost by discrete prototypes, while low-rankness compresses function-valued knowledge, thereby bridging the expressiveness–efficiency gap.
- **Collaboration in continuous function space.** We reformulate infeasible parameter aggregation across heterogeneous models as knowledge fusion in a continuous function space, and feed back global knowledge via feature–logit dual distillation, enabling efficient and lightweight cross-architecture collaboration.
- **Theoretical and empirical support.** Theoretically, we reveal that low-rank continuous fields can alleviate the structural information loss induced by single-point prototypes; empirically, FedCF achieves a better accuracy–efficiency trade-off across diverse data- and model-heterogeneous settings.

2 Method

2.1 Problem Statement and Motivation

We consider a heterogeneous collaborative learning system with N clients. Client i owns a private dataset $D_i = \{(x, y)\}$ drawn from P_i, and all clients share the label space $\mathcal{Y} = \{1, \ldots, C\}$. The local model is $f_i(x; \theta_i) = h_i\big(g_i(x; \theta_i^g); \theta_i^h\big)$, where g_i is the backbone, h_i is the classifier head, and $\theta_i = (\theta_i^g, \theta_i^h)$. Due to architectural heterogeneity, the parameters $\{\theta_i\}$ cannot be directly aggregated, and clients instead collaborate through a cross-architecture knowledge representation $\mathcal{K}$. The training

objective is

$$\min_{\{\theta_i\}_{i=1}^N} \frac{1}{N}\sum_{i=1}^{N} L_i(D_i, \theta_i, \mathcal{K}), \tag{1}$$

where L_i is the local loss of client i. Existing HtFL methods instantiate $\mathcal{K}$ through class prototypes or proxy-model supervision. However, discrete prototypes may break down during aggregation, while auxiliary-model logit distillation incurs additional overhead.

To overcome these limits, a natural idea is to enable clients to exchange all sample features for collaborative learning, but the resulting communication is prohibitive. Therefore, we need a knowledge representation K that balances expressiveness and communication efficiency. To this end, we replace discrete-point representations with a continuous feature function S to preserve rich intra-class feature structure lost by single-point prototypes, and further compress this continuous knowledge through low-rank parameterization. Intuitively, S answers what feature vector the model should produce at any spatial location for class c.

FedCF comprises three stages: (1) fitting local continuous feature fields; (2) learning global continuous fields and a classifier in the canonical space; and (3) feature–logit dual distillation to clients.

2.2 FedCF Framework

2.2.1 Local Continuous Representation

For client i, we cast the feature outputs of different models as $\Phi_i(x) = g_i(x; \theta_i^g) \in \mathbb{R}^{H_i \times W_i \times d_i}$(either the final CNN feature map or ViT patch tokens reshaped into an $H_i \times W_i$ grid), where H_i, W_i are the spatial dimensions of the feature map and d_i is the feature dimension. Thus, $\Phi_i(x)$ can be uniformly viewed as a feature tensor defined on a discrete spatial grid. Based on this structure, each grid location can be normalized to the continuous coordinate domain $[0,1]^2$, and we define class-conditional continuous feature fields to represent and share local knowledge.

Definition 2.1 (Local continuous feature field)**.** For each client–class pair (i, c), a local continuous feature field is a function $S_{i,c}^* : [0,1]^2 \to \mathbb{R}^{d_i}$, defined on the normalized spatial coordinate domain $[0,1]^2$. For any $(u, v) \in [0,1]^2$, the vector $S_{i,c}^*(u, v)$ represents the target feature of client i for class c at relative location.

Assumption 2.2 (Local field as conditional expectation)**.** For each client–class (i, c) and each grid index $h \in \{0, \dots, H_i - 1\}$, $w \in \{0, \dots, W_i - 1\}$ with normalized coordinate $u = (h + 0.5)/H_i$, $v = (w + 0.5)/W_i$, the ideal field in Definition 2.1 satisfies

$$\left\| S_{i,c}^*(u, v) - \mathbb{E}\big[\Phi_i(X)_{h,w,:} \mid Y = c\big] \right\|_2^2 \;\le\; \varepsilon_1,$$

where (X, Y) is drawn from P_i with $Y = c$, and $\varepsilon_1 \ge 0$ is a constant.

Under Assumption 2.2, at a normalized coordinate (u, v), the value $S_{i,c}^*(u, v)$ gives the expected feature vector when client i recognizes class c at that spatial location. For a labeled sample $(x, y = c)$, its feature map $\Phi_i(x)$ and the feature vector at position (h, w), namely $\Phi_i(x)_{h,w,:}$, can be viewed as a noisy observation of $S_{i,c}^*(u, v)$ at the corresponding normalized coordinate (u, v). Directly storing $S_{i,c}^*$ over $[0,1]^2$ is clearly impractical, as it would require maintaining a dense set of feature values. To obtain a lightweight approximation of $S_{i,c}^*$, inspired by Luo et al. (2024), we approximate each continuous class feature field using a low-rank tensor function factorization.

Concretely, for client i, we introduce two one-dimensional coordinate factor networks $\phi_u : [0,1] \to \mathbb{R}^{r_u}$, $\phi_v : [0,1] \to \mathbb{R}^{r_v}$, which map the vertical coordinate u and horizontal coordinate v into low-dimensional factor spaces. Meanwhile, for each class c we learn a core tensor $G_{i,c} \in \mathbb{R}^{r_u \times r_v \times d_i}$. We then define the local continuous class function as

$$S_{i,c}(u, v; \theta_i^S) = G_{i,c} \times_1 \phi_u(u) \times_2 \phi_v(v) \in \mathbb{R}^{d_i}, \tag{2}$$

where $\times_1$ and $\times_2$ denote tensor–vector products along the first and second modes, respectively. θ_i^S comprises the network weights of ϕ_u, ϕ_v and the core tensors $\{G_{i,c}\}$. With small ranks r_u, r_v, each class field adds only a core of size $r_u \times r_v \times d_i$, yielding a compact, rank-controlled representation of class-conditional spatial structure.

Since $S^*_{i,c}$ is unobservable, we approximate it by fitting Si, c to noisy samples at discrete coordinates, casting field fitting as a regression problem. We seek parameters θ^S_i such that $S_{i,c}(u, v; \theta^S_i)$ predicts the feature pattern at the corresponding coordinates. For each (x, y), we compute the feature map $\Phi_i(x)$ through g_i and map each grid index (h, w) to its normalized coordinate (u, v). In principle, all triples $\big((u, v), \Phi_i(x)h, w, :, y\big)$ on the feature map can be treated as supervised samples, where (u, v) is the input, $\Phi_i(x)h, w, :$ is the observed target vector, and y determines which $S_{i,y}$ is invoked.

To reduce computation and keep the training signals balanced, we uniformly sample a small subset $\Omega(x)$ from the $H_i \times W_i$ spatial locations and form the regression loss only on these sampled points. Collecting all samples yields the regression set $D^s_i = \big\{((u, v), z, y) : (x, y) \in D_i, (h, w) \in \Omega(x), z = \Phi_i(x)_{h,w,:}\big\}$. Define the regression loss as

$$L^s_i(\theta^S_i) = \frac{1}{|D^s_i|} \sum_{((u,v),\, z,\, y)\in D^s_i} \big\|S_{i,y}(u, v; \theta^S_i) - z\big\|^2_2. \tag{3}$$

Under Assumption 2.2, the squared-loss minimizer corresponds to the conditional expectation feature field, providing a principled way to estimate $S^*_{i,c}$ from discrete features. In practice, we freeze g_i after local training and update only θ^S_i. From local feature maps $\{\Phi_i(x)\}$, we sample (u, v) and use $\Phi_i(x)_{h,w,:}$ as targets. For each sampled point, the current $S_{i,y}$ produces a predicted feature vector, from which we compute L^s_i and update θ^S_i. After a few iterations, the resulting $S_{i,c}(u, v; \theta^S_i)$ can be regarded as a low-rank continuous approximation of $S^*_{i,c}$ over $[0, 1]^2$, capable of providing a continuous interpolation of class-conditional mean features over the normalized coordinate domain. Crucially, each $S_i = \{S_{i,c}\}_{c\in\mathcal{Y}}$ compactly summarizes the spatially varying class-conditional mean features of client i, providing rich knowledge for subsequent client collaboration.

2.2.2 Global Continuous Representation

The local fields $S_{i,c}$ convert discrete features into continuous knowledge on $[0, 1]^2$, mitigating geometric discrepancies across architectures, but still reside in client-specific feature spaces. We therefore introduce a canonical global space to fuse class knowledge across clients.

Definition 2.3 (Global continuous feature field)**.** For each class $c \in \mathcal{Y}$, a shared canonical coordinate domain $[0, 1]^2$ is fixed across all clients. A global continuous feature field for class c is a continuous function $S^*_c : [0, 1]^2 \to \mathbb{R}^d$, which assigns to every canonical coordinate $(s, t) \in [0, 1]^2$ a class-c feature vector $S^*_c(s, t)$ in the shared canonical feature space $\mathbb{R}^d$.

Assumption 2.4 (Local–global alignment with bounded error)**.** For each client i and class c, there exist a coordinate map $\psi_i : [0, 1]^2 \to [0, 1]^2$ and a feature translator $A_i : \mathbb{R}^{d_i} \to \mathbb{R}^d$ such that for any $(u, v) \in [0, 1]^2$ and $(s, t) = \psi_i(u, v)$, the global field in Definition 2.3 satisfies

$$\big\|S^*_c(s, t) - A_i(S^*_{i,c}(u, v))\big\|^2_2 \le \varepsilon_2, \quad \varepsilon_2 \ge 0.$$

Intuitively, after coordinate and feature mapping, each $S^*_{i,c}$ is transferred to the shared space and becomes close to the same global field S^*_c. We approximate S^*_c using the same lightweight low-rank tensor-function representation. In the canonical space, we introduce two one-dimensional coordinate factor networks $\phi_s : [0, 1] \to \mathbb{R}^{r_s}, \phi_t : [0, 1] \to \mathbb{R}^{r_t}$, shared across all classes, and for each class c we learn a global core tensor $G_c \in \mathbb{R}^{r_s\times r_t\times d}$. Define the global continuous function as

$$S_c(s, t; \theta^G) = G_c \times_1 \phi_s(s) \times_2 \phi_t(t) \in \mathbb{R}^d, \tag{4}$$

where θ^G collects the parameters of ϕ_s, ϕ_t and the cores $\{G_c\}$. We parameterize ψ_i and A_i with lightweight MLPs and ℓ_2-normalize the translated features. Given $S_{i,c}$, its translated feature in the canonical space is $z^{\text{feat}}_{i,c}(s, t) = A_i(S_{i,c}(u, v)) \in \mathbb{R}^d$, $(s, t) = \psi_i(u, v)$. To reduce parameter overhead, clients in the same backbone family may share (A_i, ψ_i). For modeling, we still treat each client as having its own translator.

The fields S_c and translators $\{\psi_i, A_i\}$ are jointly learned by querying local fields without accessing raw images. For each client i and class c, we uniformly sample $\Omega^c_i \subset [0, 1]^2$, where each $(u, v) \in \Omega^c_i$ yields a tuple $\big((s, t), z^{\text{feat}}_{i,c}, c\big)$. All tuples form the regression dataset $D^g = \Big\{((s, t), z^{\text{feat}}_{i,c}, c) : i \in$

$\{1, \ldots, N\},\ c \in \mathcal{Y}\Big\}$. We fit $\{S_c\}$ by minimizing the least-squares objective

$$L^g(\theta^G) = \frac{1}{|D^g|} \sum_{((s,t),\, z_{i,c}^{\text{feat}},\, c) \in D^g} \big\| S_c(s,t;\theta^G) - z_{i,c}^{\text{feat}} \big\|_2^2. \tag{5}$$

In practice, we use the same objective (5) to optimize both the global fields and the translators, and adopt an alternating scheme: (i) fixing $\{\psi_i, A_i\}$ and updating θ^G to fit the current translated local fields; (ii) fixing θ^G and updating $\{\psi_i, A_i\}$ based on D^g. The global fields and translators are optimized under a single objective with alternating updates, improving stability and yielding a global continuous representation that aggregates class semantics across heterogeneous clients.

2.2.3 GLOBAL CLASSIFIER

Continuous feature fields address what feature knowledge to share, but do not explicitly capture inter-class decision boundaries. To further integrate decision-level knowledge, we train a lightweight global classifier in the canonical feature space. To train the classifier, we discard the coordinate information in D^g and keep only the feature–label pairs, yielding the canonical classification dataset $D^{\text{cls}} = \{(z_{i,c}^{\text{feat}}, c) : ((s,t), z_{i,c}^{\text{feat}}, c) \in D^g\}$.

Since $S_{i,c}$ is a continuous function, extra labeled features can be queried at negligible cost by evaluating it on resampled coordinates, without raw or proxy data, reducing communication and preserving privacy. We train a shared shallow-MLP classifier $g_{\text{glob}}(z_{i,c}^{\text{feat}}; \theta^M) : \mathbb{R}^d \to \mathbb{R}^C$ on D^{cls}:

$$L_{\text{cls}}(\theta^M) = \frac{1}{|D^{\text{cls}}|} \sum_{(z_{i,c}^{\text{feat}}, c) \in D^{\text{cls}}} \ell_{\text{ce}}\big(g_{\text{glob}}(z_{i,c}^{\text{feat}}; \theta^M),\, c\big), \tag{6}$$

where ℓ_{ce} denotes the multi-class cross-entropy loss. In this stage, we only update the classifier parameters to learn a decision boundary in the shared space.

2.2.4 LOCAL TRAINING WITH CONTINUOUS FIELDS

The server broadcasts $\{S_c\}$, $\{\psi_i, A_i\}$, and g_{glob} to all clients. To reduce communication cost, client i only receives the core tensors of the global fields for the classes present in its local data. Each client then absorbs this knowledge locally at both feature and logit levels.

Global feature alignment. For a labeled sample $(x, y = c)$ on client i, let $\Phi_i(x)$ be its feature map. For each sampled grid index $(h, w) \in \Omega(x)$ with normalized coordinates (u, v), the corresponding backbone feature is $z_i = \Phi_i(x)_{h,w,:} \in \mathbb{R}^{d_i}$. The translator maps the backbone feature and its coordinate into the canonical space $z_{i,c}^{\text{local}}(u,v) = A_i\big(z_i\big), (s,t) = \psi_i(u,v)$. The global continuous field provides the corresponding canonical teacher feature $z_c^{\text{global}}(s,t) = S_c(s,t) \in \mathbb{R}^d$. The local backbone is regularized so that, after translation by A_i and ψ_i, its spatial features align with the global field at the same canonical coordinates. Concretely, define

$$\ell_{\text{feat}} = \frac{1}{|\Omega(x)|} \sum_{(h,w) \in \Omega(x)} \big\| z_{i,c}^{\text{local}}(u,v) - z_c^{\text{global}}(s,t) \big\|_2^2, \tag{7}$$

which measures the mean-squared discrepancy between translated local and canonical global features at sampled spatial locations. Client i's expected feature-alignment loss is $L_i^{\text{feat}}(\theta_i) = \mathbb{E}_{(x,y)\sim D_i} \ell_{\text{feat}}$.

Global classifier distillation. While L_i^{feat} aligns intermediate features, we also distill the decision boundary encoded by the global classifier g_{glob}. For the same sample $(x, y = c)$ and its sampled locations $(h, w) \in \Omega(x)$, the $S_{i,c}$ is queried as $\hat{z}_{i,c} = S_{i,c}(u, v; \theta_i^S) \in \mathbb{R}^{d_i}$, which is projected into the shared space as $z_{i,c}^{\text{feat}} = A_i(\hat{z}_{i,c})$, and fed into the global classifier $q_{i,c} = g_{\text{glob}}\big(z_{i,c}^{\text{feat}}; \theta^M\big) \in \mathbb{R}^C$.

With temperature $\rho > 0$, the teacher distribution is $p_{i,c} = \text{softmax}\big(q_{i,c}/\rho\big)$. The student's image-level logits $q_i^s(x) = f_i(x; \theta_i) \in \mathbb{R}^C$ and induce $p_i^s(x) = \text{softmax}\big(q_i^s(x)/\rho\big)$, shared across sampled locations of x. We average the KL divergence between these distributions over sampled coordinates:

$$\ell_{\text{dist}} = \frac{1}{|\Omega(x)|} \sum_{(h,w) \in \Omega(x)} \text{KL}\big(p_{i,c} \,\|\, p_i^s\big). \tag{8}$$

The expected distillation loss on client i is $L_i^{\mathrm{dist}}(\theta_i) = \mathbb{E}_{(x,y)\sim D_i}\ell_{\mathrm{dist}}$. Thus, each image-level prediction on client i is required to agree at canonical spatial locations, with the class scores that the global continuous teacher would assign to its continuous features, thereby obtaining guidance from the global classifier.

Overall local objective and optimization. The local training objective on client i is

$$L_i(\theta_i) = L_i^{\mathrm{ce}}(\theta_i) + \lambda_{\mathrm{feat}}\, L_i^{\mathrm{feat}}(\theta_i^g) + \lambda_{\mathrm{dist}}\, L_i^{\mathrm{dist}}(\theta_i), \tag{9}$$

where $L_i^{\mathrm{ce}}(\theta_i) = \frac{1}{|D_i|}\sum_{(x,y)\in D_i}\ell_{\mathrm{ce}}\big(f_i(x;\theta_i), y\big)$ is the standard cross-entropy loss, and $\lambda_{\mathrm{feat}}, \lambda_{\mathrm{dist}} \geq 0$ control the strength of feature alignment and distillation. During local training on client i, the parameters of $\{S_c\}$, (ψ_i, A_i), and g_{glob} are frozen, and only the backbone and classifier parameters θ_i are updated.

As local models are updated, their feature distributions change, so the local continuous fields $\{S_{i,c}\}$ are periodically refreshed. After local training, θ_i is frozen, coordinate–feature pairs are resampled from the cached feature grids, and θ_i^S is updated by minimizing the regression loss in Eq. (3). Alternating backbone updates and field re-fitting keep $S_{i,c}$ aligned with the current backbone and provide up-to-date continuous representations for subsequent global updates. Overall, this procedure enables each local model to absorb global knowledge at both the feature and decision levels without exchanging raw samples or training proxy models.

2.3 Theoretical Analysis

Theorem 2.5 (Sampling-induced low-rank property of local fields)**.** *For a low-rank continuous field $S_{i,c}(\cdot,\cdot)$ with ranks (r_u, r_v) and core $G_{i,c} \in \mathbb{R}^{r_u\times r_v\times d_i}$, any coordinate sets $\{u_\alpha\}_{\alpha=1}^{n_u} \subset [0,1]$ and $\{v_\beta\}_{\beta=1}^{n_v} \subset [0,1]$, define the sampled tensor $\mathbf{S}_{i,c}^{n_u\times n_v} \in \mathbb{R}^{n_u\times n_v\times d_i}$ by $\mathbf{S}_{i,c}^{n_u\times n_v}[\alpha,\beta,:] = S_{i,c}(u_\alpha, v_\beta)$. Then $\mathbf{S}_{i,c}^{n_u\times n_v}$ admits the Tucker factorization $\mathbf{S}_{i,c}^{n_u\times n_v} = G_{i,c}\times_1 \mathbf{F}_u \times_2 \mathbf{F}_v$, where $\mathbf{F}_u \in \mathbb{R}^{n_u\times r_u}$ and $\mathbf{F}_v \in \mathbb{R}^{n_v\times r_v}$ are given by $\mathbf{F}_u[\alpha,:] = \phi_u(u_\alpha)^\top$ and $\mathbf{F}_v[\beta,:] = \phi_v(v_\beta)^\top$. Consequently, for any sampling resolution $n_u \times n_v$, the sampled tensor $\mathbf{S}_{i,c}^{n_u\times n_v}$ has spatial Tucker ranks at most r_u and r_v. This yields a compression guarantee: dense sampling preserves low-rank structure, enabling communication-efficient sharing of knowledge representation.*

Corollary 2.6 (Prototypes as a rank-1 special case)**.** *Under the local continuous function in Eq. (2), if $r_u = r_v = 1$ and $\phi_u(\cdot) \equiv 1$, $\phi_v(\cdot) \equiv 1$ on $[0,1]$, then the field degenerates to a spatially constant vector $S_{i,c}(u,v) \equiv \mu_{i,c} \in \mathbb{R}^{d_i}$ with $\mu_{i,c} := G_{i,c}[1,1,:]$. Therefore, any prototype-based method that represents each class by a single vector can be viewed as instantiating our field with spatial rank* 1.

Lemma 2.7 (Irreducible approximation error of prototypes)**.** *Let $\epsilon_{\mathrm{proto}}^2$ be the best prototype approximation error to $S_{i,c}^*$: $\epsilon_{\mathrm{proto}}^2 := \min_{\mu_{i,c}} \mathbb{E}\big\|S_{i,c}^*(u,v) - \mu_{i,c}\big\|_2^2$. Then $\epsilon_{\mathrm{proto}}^2 = \mathrm{Var}\big(S_{i,c}^*(u,v)\big), \epsilon_r^2 \leq \epsilon_{\mathrm{proto}}^2$, where ϵ_r^2 is the optimal approximation error of our field model and prototypes correspond to its constant-field degeneration.*

Theorem 2.8 (Local convergence bound)**.** *Let $L_i^*(\theta)$ denote the ideal local objective induced by the population canonical field S^*, and let $L_i(\theta)$ denote the realizable local objective induced by the learned teacher, fixed within each round. Client i performs E stochastic-gradient updates per round for T rounds with stepsize η, producing $\{\theta_\tau\}_{\tau=0}^{TE-1}$, and uses $K := |\Omega(x)|$ queried points per image in the pointwise losses. Assume L_i^* is M-smooth, the stochastic gradients are unbiased with bounded variances, and $\|\nabla L_i^*(\theta) - \nabla L_i(\theta)\|_2 \leq \kappa\,\epsilon_r$. If $\eta \leq \frac{1}{2M}$, then*

$$\frac{1}{TE}\sum_{\tau=0}^{TE-1}\mathbb{E}\big\|\nabla L_i^*(\theta_\tau)\big\|_2^2 \leq \frac{2\Delta}{\eta TE} + \frac{M\eta}{K}\,\Sigma^2 + M\eta\,\Sigma_{\mathrm{mb}}^2 + \kappa^2\,\epsilon_r^2, \tag{10}$$

where $\Delta := \mathbb{E}[L_i^*(\theta_0)] - \inf_\theta L_i^*(\theta)$, Σ^2 bounds the variance of the pointwise stochastic gradient estimator, and Σ_{mb}^2 bounds the mini-batch sampling noise.

Remark 2.9. Eq. (10) is a non-convex stationarity bound for L_i^*, decomposed into optimization, variance, and non-vanishing terms. Compared with prototype supervision, FedCF can provide a tighter representation term, while multi-point querying further reduces the variance of its pointwise supervision. **(i) Richer supervision class.** Prototype-based supervision is a constant-field degeneration. Its best achievable *constant-field* approximation error satisfies $\epsilon_{\mathrm{proto}}^2 = \mathrm{Var}_{(u,v)}(S_{i,c}^*)$, which

Table 1: Accuracy (%), computation cost (10^4 GFLOPs/round), and communication cost (MB/round) under different data partitions. *Costs* use Pat; *Aux. source* denotes the data source used by the auxiliary model: *Local* (client data), *Synth* (synthetic data), or *None* (no extra data).

Method Group	Aux. source	Datasets	CIFAR-10				CIFAR-100				Tiny-ImageNet			
		Metrics	*Acc.*↑		*Cost*↓		*Acc.*↑		*Cost*↓		*Acc.*↑		*Cost*↓	
		Settings	*Pat*	$Dir_{0.1}$	*Comp.*	*Comm.*	*Pat*	$Dir_{0.1}$	*Comp.*	*Comm.*	*Pat*	$Dir_{0.1}$	*Comp.*	*Comm.*
Proxy model	*Local*	FedKD	86.46	84.43	8.80	260.11	62.68	58.12	141.34	265.36	44.23	36.62	405.61	271.26
	Local	FedSCE	85.22	84.52	8.80	260.11	63.62	57.25	141.34	265.36	43.52	36.13	405.61	271.26
	Local	pFedMoE	84.25	82.37	8.52	153.59	59.11	54.25	141.05	153.59	40.63	33.08	404.25	153.59
	Synth	FedGen	83.99	82.12	8.25	46.99	58.76	54.35	140.78	62.84	40.64	32.85	401.79	80.44
	Synth	FedZGE	82.73	79.41	9.93	993.52	58.60	55.10	169.46	1025.16	41.39	33.35	443.58	4030.31
Prototype	*None*	FedProto	84.18	81.56	8.25	0.76	58.92	54.52	140.78	6.74	39.45	33.11	401.78	13.48
	None	FedTGP	83.71	82.39	8.25	0.76	59.37	55.63	140.78	6.74	40.36	33.83	401.78	13.48
	None	FedSA	83.89	82.69	8.25	0.76	57.81	53.14	140.78	6.74	40.74	32.74	401.78	13.48
	None	FedRE	84.74	82.79	9.17	0.65	59.49	54.83	156.42	5.94	40.30	33.05	446.42	11.82
Ours	***None***	FedCF	**86.58**	**84.63**	8.36	7.43	**64.02**	**59.30**	140.92	33.10	**45.10**	**37.54**	402.10	65.18

induces a non-vanishing floor $\kappa^2\epsilon^2_{\text{proto}}$ whenever $S^*_{i,c}$ is non-constant. In contrast, our low-rank field class strictly contains the constant-field class, implying $\epsilon_r^2 \le \epsilon^2_{\text{proto}}$, with strict inequality when the field captures nonconstant variation that cannot be represented by a constant field, and thus yields a no-larger representation term in Eq. (10).

(ii) Multi-point queried supervision. Our pointwise losses average supervision over $K = |\Omega(x)|$ queried locations per image, leading to the explicit variance reduction factor $1/K$ in $\frac{M\eta}{K}\Sigma^2$. Prototype methods provide only a single class vector, hence they cannot benefit from the same multi-point querying effect for learning spatially varying semantics. Full proofs of the theoretical results are deferred to Appendix E.

3 Experiments

3.1 Experimental Setup

Datasets. We evaluate FedCF on CIFAR-10, CIFAR-100 and Tiny-ImageNet under two common partition protocols. In the pathological setting, following FedAvg (McMahan et al., 2017), each client is assigned an imbalanced subset with a fixed number of classes (3/15/30 for CIFAR-10/CIFAR-100/Tiny-ImageNet, respectively). In the practical setting, following Li et al. (2021), we generate non-IID splits via a Dirichlet distribution $\text{Dir}(\alpha)$ with $\alpha = 0.1$ by default.

Model heterogeneity. To assess performance under model heterogeneity, we customize heterogeneous model pools for different tasks. LeNet/AlexNet/ResNet for CIFAR-10; AlexNet/ResNet/ViT for CIFAR-100; and MobileNet/ResNet/ViT for Tiny-ImageNet, which better fit each task while inducing severe heterogeneity.

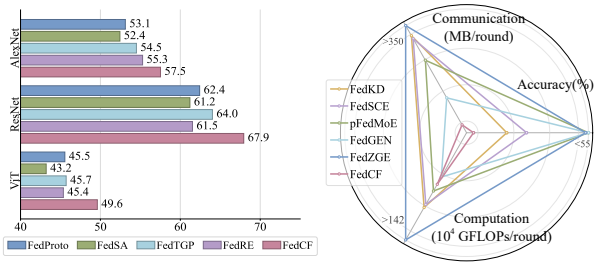


Figure 2: CIFAR-100 accuracy and cost. Left: accuracy by backbone. Right: accuracy–efficiency trade-off; outward indicates worse performance.

Baselines. We compare FedCF with prototype-based baselines (FedProto (Tan et al., 2022), FedTGP (Zhang et al., 2024), FedSA (Zhou et al., 2025), FedRE (Yao et al., 2026)) and proxy-model baselines using local data (FedKD (Wu et al., 2022), FedSCE (Zhang et al., 2025), pFedMoE (Yi et al., 2026)) or synthetic data (FedGen (Zhu et al., 2021), FedZGE (Ma et al., 2025)).

Implementation details. We use 30 clients with full participation; each client runs 3 local epochs with Adam ($\eta = 0.0003$), and the number of global rounds varies across tasks. Both local and global field fitting use Adam ($\eta_{\text{field}} = 0.01$) for 2–3 rounds. We report the average test accuracy over all clients evaluated on their local test sets. Additional details are provided in Appendix G.

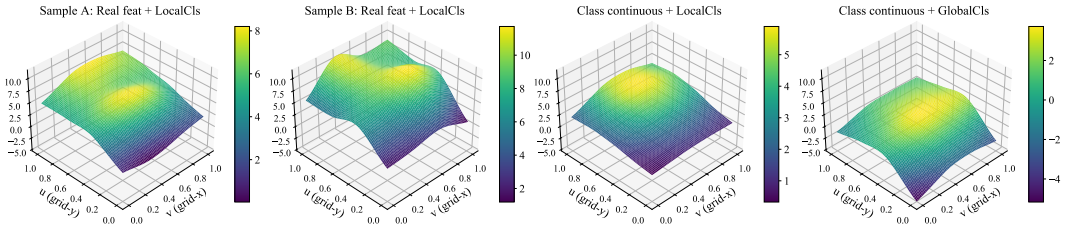


Figure 3: 3D surfaces of class-direction projections. u, v denote normalized coordinates corresponding to 2D locations on the original feature map. The surface value represents the projection response of features along the class-c weight direction of the local/global classifier. Samples A and B belong to the same class, while class continuous denotes the local class-specific continuous field.

3.2 Comparison with Baselines

Overall accuracy. Table 1 shows that prototype-based methods become less competitive on harder tasks, consistent with the limited expressiveness of discrete prototypes. FedRE further entangles local prototypes to train a global classifier, but its shared knowledge remains highly compressed. In contrast, FedCF preserves richer intra-class structures through continuous feature fields and consistently achieves the best accuracy, with gains of up to 4.53 percentage points over the strongest prototype baseline. Figure 2 (left) further shows that the gains are consistent across heterogeneous backbones, with FedCF achieving the largest improvement for each architecture group. FedGen/FedZGE yield limited gains, as the server-side generator struggles to match diverse local distributions under severe non-IID partitions.

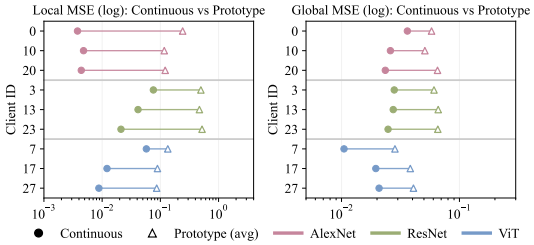


Figure 4: Reconstruction MSE (log) of continuous fields vs. mean prototypes in local and global canonical spaces.

Communication cost. FedKD/FedSCE perform well because a shared proxy model is repeatedly aggregated, approximating training on the union of client data. However, this incurs large communication overhead, with average cost about 7.5× that of FedCF across all datasets. pFedMoE reduces this cost by sharing only a proxy feature extractor, but global knowledge is mainly transferred at the feature level. In contrast, FedCF shares low-rank continuous feature fields. Continuity preserves richer intra-class structure than discrete prototypes, while low-rank parameterization enables efficient transmission in a compact form. Figure 2 (right) further shows that, compared with all proxy-model baselines, FedCF achieves a more favorable accuracy–efficiency trade-off.

3.3 Advantages of Continuous Feature Fields

Richer spatial structure. For a fixed client and class, we visualize four 3D response surfaces over the normalized coordinate grid. As shown in Figure 3, the two sample-feature surfaces differ markedly in shape, reflecting sample-level noise and local variations. In contrast, the surface induced by the local continuous field is smoother with a stable, consistent structure that a single class-mean prototype cannot capture. Moreover, the continuous-field response exhibits a spatial pattern compatible with the global classifier's class-c direction, suggesting that the learned field preserves discriminative class-specific structure useful for cross-client collaboration.

Superior feature reconstruction. We quantify how well continuous fields fit real features compared to mean prototypes using reconstruction error (MSE). For each client i, we compute the average MSE of continuous fields and class-mean prototypes on sampled real features, both in the local feature space and global canonical space, shown in Figure 4. Continuous fields achieve lower MSE for clients in both spaces, indicating that $S_{i,c}$ and S_c better fit class-conditional features; the larger gap in the canonical space further suggests that S_c alleviates the prototype bottleneck under severe heterogeneity, where a single-point prototype cannot match diverse feature geometries.

Cross-architecture effectiveness. We further use Grad-CAM to assess whether the shared semantics are consistent across heterogeneous models. As shown in Figure 5, prototype-CAMs are often more diffuse and highlight inconsistent regions across architectures, indicating semantic mismatch when using a single vector prototype under heterogeneous feature geometries. In contrast, continuous-CAMs consistently focus on similar semantic parts, indicating that the global field S_c serves as a shared semantic template and supports more accurate knowledge transfer.

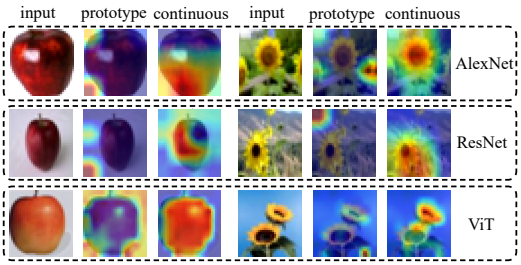


Figure 5: Prototype vs. continuous-field Grad-CAM on CIFAR-100.

3.4 Ablation Study

Ablation. Figure 6(a) shows that the full objective achieves the best accuracy. L_{feat} alone lacks decision-level constraints and cannot mitigate classifier drift, while L_{dist} can align the global decision boundary but its distillation signal is constrained by heterogeneous representation mismatch and is not fully absorbed. Combining them forms a closed loop from representation alignment to decision calibration, yielding stable gains. The rank=1 variant uses constant coordinate factors and thus degenerates the field to a class-wise mean vector, dropping by about 4.8 percentage points on average, suggesting improvements come from richer intra-class structure rather than extra regularization. For optimizing the global field and translators, FedCF adopts alternating updates. As shown in Figure 6(b), compared with jointly optimizing both, alternating updates achieve higher accuracy and lower Align MSE, indicating better optimization stability.

Low-rankness. Figure 6(c) examines reconstruction MSE as a function of the local field rank r. For each backbone, the log-MSE of continuous fields aggregated over all clients is compared with the mean-prototype baseline. Continuous fields already outperform prototypes at small r and quickly saturate as r increases. AlexNet/ResNet are near-optimal around r=6, whereas the more complex ViT benefits from a larger r. This suggests that compact low-rank fields are sufficient to capture class-conditional spatial structure, while a single mean vector is limited in expressiveness.

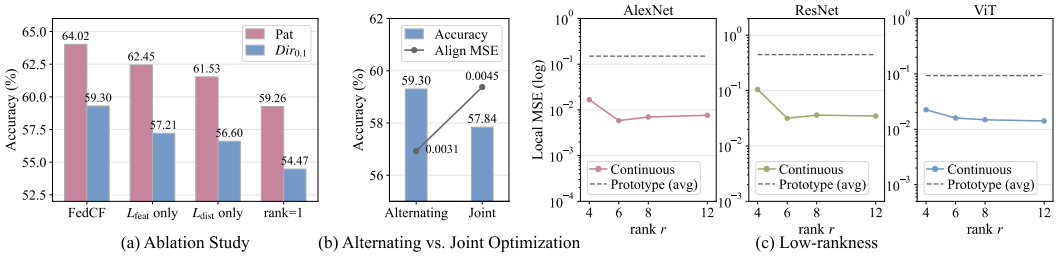


Figure 6: Ablation study. (a) Component ablation under Pat and Dir(0.1). (b) Comparison of alternating and joint optimization in terms of accuracy and Align MSE. (c) Effect of field rank r on local reconstruction MSE across different backbones, compared with mean prototypes.

4 Conclusion

This paper proposes FedCF, using lightweight yet expressive low-rank continuous feature fields as the collaboration carrier for HtFL, enabling knowledge collaboration among heterogeneous clients in function space. Continuous functions preserve rich intra-class structure lost by discrete prototypes, while low-rank parameterization enables compact, efficient knowledge transmission. FedCF transfers feature- and decision-level knowledge through global continuous fields and a global classifier, enabling cross-architecture collaboration without proxy models or public data. Across diverse heterogeneous settings, FedCF consistently outperforms prototype-based methods and achieves the best accuracy–efficiency trade-off. These results further show that low-rank continuous representations alleviate the expressiveness–efficiency gap in heterogeneous collaboration, enabling efficient cross-architecture sharing while preserving rich knowledge.

AI USE STATEMENT

In this work, we used generative AI to polish the manuscript, retrieve relevant literature, and identify errors in the presentation of the paper. AI was not used to formulate the core research methodology, design or conduct experiments, generate experimental data or results, implement the core method, or perform theoretical analysis. No other uses requiring disclosure are applicable to this work. All AI-assisted content was independently verified by the authors, who take full responsibility for the final research design, implementation, experimental results, theoretical analysis, and content of the paper.

REPRODUCIBILITY STATEMENT

Detailed information on the method and experiments is provided in Appendices B–D and G, including the complete FedCF algorithm. All theoretical assumptions and proofs are provided in Appendix E, and the computation and communication analyses are provided in Appendix F.

REFERENCES

Shamim Ahmed, M. Shamim Kaiser, Sudipto Chaki, Saad Aloteibi, and Mohammad Ali Moni. Federated learning model with dynamic scoring-based client selection for diabetes diagnosis. *Knowledge-Based Systems*, 320:113662, 2025.

Huancheng Chen and Haris Vikalo. Heterogeneity-guided client sampling: Towards fast and efficient non-iid federated learning. In *Proceedings of the Advances in Neural Information Processing Systems*, volume 37, pp. 65525–65561, 2024.

Yinbo Chen, Sifei Liu, and Xiaolong Wang. Learning continuous image representation with local implicit image function. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8628–8638, 2021.

Zhengyun Cheng, Ruizhe Zhang, Guanwen Zhang, Yi Xu, Xiangyang Ji, and Wei Zhou. Low-rank tensor recovery via variational schatten-p quasi-norm and jacobian regularization. *arXiv preprint arXiv:2506.22134*, 2025.

Enmao Diao, Jie Ding, and Vahid Tarokh. HeteroFL: Computation and communication efficient federated learning for heterogeneous clients. In *Proceedings of the International Conference on Learning Representations*, pp. 1–24, 2021.

Fucheng Guo, Zeyu Luan, Qing Li, Dan Zhao, and Yong Jiang. FedORGP: Guiding heterogeneous federated learning with orthogonality regularization on global prototypes. *arXiv preprint arXiv:2502.16119*, 2025.

Lachlan Holden, Feras Dayoub, David Harvey, and Tat-Jun Chin. Federated neural radiance fields. *arXiv preprint arXiv:2305.01163*, 2023.

Samuel Horváth, Stefanos Laskaridis, Mario Almeida, Ilias Leontiadis, Stylianos Venieris, and Nicholas Lane. FjORD: Fair and accurate federated learning under heterogeneous targets with ordered dropout. In *Proceedings of the Advances in Neural Information Processing Systems*, volume 34, pp. 12876–12889, 2021.

Wenke Huang, Mang Ye, Zekun Shi, Guancheng Wan, He Li, Bo Du, and Qiang Yang. Federated learning for generalization, robustness, fairness: A survey and benchmark. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(12):9387–9406, 2024.

Zhouyuan Huo, Bin Gu, Qian Yang, and Heng Huang. Decoupled parallel backpropagation with convergence guarantee. In *Proceedings of the International Conference on Machine Learning*, volume 80, pp. 2098–2106, 2018.

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

Tsung-Ting Kuo, Rodney A Gabriel, Jejo Koola, Robert T Schooley, and Lucila Ohno-Machado. Distributed cross-learning for equitable federated models-privacy-preserving prediction on data from five california hospitals. *Nature Communications*, 16(1):1371, 2025.

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

Xin-Chun Li, Yi-Chu Xu, Shaoming Song, Bingshuai Li, Yinchuan Li, Yunfeng Shao, and De-Chuan Zhan. Federated learning with position-aware neurons. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10072–10081, 2022.

Yisi Luo, Xile Zhao, Zhemin Li, Michael K. Ng, and Deyu Meng. Low-rank tensor function representation for multi-dimensional data recovery. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46(5):3351–3369, 2024.

Xinge Ma, Jin Wang, and Xuejie Zhang. Data-free black-box federated learning via zeroth-order gradient estimation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(18):19314–19322, 2025.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, pp. 1273–1282, 2017.

Chuizheng Meng, Jianke Yang, Hao Niu, Guillaume Habault, Roberto Legaspi, Shinya Wada, Chihiro Ono, and Yan Liu. Sample-level prototypical federated learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 48(2):1133–1144, 2026.

Zhuang Qi, Lei Meng, Zhaochuan Li, Han Hu, and Xiangxu Meng. Cross-silo feature space alignment for federated learning on clients with imbalanced data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 19986–19994, 2025.

Zixuan Qin, Liu Yang, Qilong Wang, Yahong Han, and Qinghua Hu. Reliable and interpretable personalized federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20422–20431, 2023.

Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. *Proceedings of the Advances in Neural Information Processing Systems*, 33:7462–7473, 2020.

Chuanneng Sun, Tingcong Jiang, and Dario Pompili. Heterogeneous federated learning via generative model-aided knowledge distillation in the edge. *IEEE Internet of Things Journal*, 12(5):5589–5599, 2025.

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

Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *Proceedings of the Advances in Neural Information Processing Systems*, 33:7537–7547, 2020.

Mahendra Singh Thapa and Rui Li. Harnessing heterogeneous statistical strength for personalized federated learning via hierarchical bayesian inference. In *Proceedings of the International Conference on Machine Learning*, pp. 59285–59303, 2025.

Hai Anh Tran and Nam-Thang Hoang. Towards efficient and adaptive traffic classification: A knowledge distillation-based personalized federated learning framework. *IEEE Transactions on Network and Service Management*, 23:594–604, 2026.

Sai Karthikeya Vemuri, Tim Büchner, and Joachim Denzler. F-INR: Functional tensor decomposition for implicit neural representations. In *Proceedings of the Winter Conference on Applications of Computer Vision*, pp. 6557–6568, 2026.

Hongyi Wang, Mikhail Yurochkin, Yuekai Sun, Dimitris Papailiopoulos, and Yasaman Khazaeni. Federated learning with matched averaging. In *Proceedings of the International Conference on Learning Representations*, pp. 1–16, 2020.

Jiaqi Wang, Chenxu Zhao, Lingjuan Lyu, Quanzeng You, Mengdi Huai, and Fenglong Ma. Bridging model heterogeneity in federated learning via uncertainty-based asymmetrical reciprocity learning. In *Proceedings of the International Conference on Machine Learning*, volume 235, pp. 52290–52308, 2024.

Chuhan Wu, Fangzhao Wu, Lingjuan Lyu, Yongfeng Huang, and Xing Xie. Communication-efficient federated learning via knowledge distillation. *Nature Communications*, 13(1):2032, 2022.

Feijie Wu, Xingchen Wang, Yaqing Wang, Tianci Liu, Lu Su, and Jing Gao. FIARSE: Model-heterogeneous federated learning via importance-aware submodel extraction. In *Proceedings of the Advances in Neural Information Processing Systems*, volume 37, pp. 115615–115651, 2024.

Yuan Yao, Lixu Wang, Jiaqi Wu, Jin Song, Simin Chen, Zehua Wang, Zijian Tian, Wei Chen, Huixia Li, and Xiaoxiao Li. Fedre: A representation entanglement framework for model-heterogeneous federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 39466–39475, 2026.

Mang Ye, Xiuwen Fang, Bo Du, Pong C. Yuen, and Dacheng Tao. Heterogeneous federated learning: State-of-the-art and research challenges. *ACM Computing Surveys*, 56(3):1–44, 2023.

Liping Yi, Han Yu, Chao Ren, Gang Wang, Xiaoguang Liu, and Xiaoxiao Li. Federated model heterogeneous matryoshka representation learning. In *Proceedings of the Advances in Neural Information Processing Systems*, volume 37, pp. 66431–66454, 2024.

Liping Yi, Han Yu, Gang Wang, Xiaoguang Liu, and Qinghua Hu. pfedmoe: Data-level personalization with mixture of experts in model-heterogeneous personalized federated learning. *IEEE Transactions on Knowledge and Data Engineering*, 2026.

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 *Proceedings of the International Conference on Machine Learning*, pp. 73009–73050, 2025.

Junhyeog Yun, Minui Hong, and Gunhee Kim. FedMeNF: Privacy-preserving federated meta-learning for neural fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 2161–2171, 2025.

Jianqing Zhang, Yang Liu, Yang Hua, and Jian Cao. FedTGP: Trainable global prototypes with adaptive-margin-enhanced contrastive learning for data and model heterogeneity in federated learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 16768–16776, 2024.

Xiangtao Zhang, Sheng Li, Ao Li, Yipeng Liu, Fan Zhang, Ce Zhu, and Le Zhang. Subspace constraint and contribution estimation for heterogeneous federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20632–20642, 2025.

Hao Zheng, Zhigang Hu, Liu Yang, Meiguang Zheng, Aikun Xu, and Boyu Wang. FedCALM: Conflict-aware layer-wise mitigation for selective aggregation in deeper personalized federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15444–15453, 2025.

Yanbing Zhou, Xiangmou Qu, Chenlong You, Jiyang Zhou, Jingyue Tang, Xin Zheng, Chunmao Cai, and Yingbo Wu. FedSA: A unified representation learning via semantic anchors for prototype-based federated learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 23009–23017, 2025.

Zhuangdi Zhu, Junyuan Hong, and Jiayu Zhou. Data-free knowledge distillation for heterogeneous federated learning. In *Proceedings of the International Conference on Machine Learning*, pp. 12878–12889, 2021.

A Related Work

A.1 Heterogeneous Federated Learning

Federated learning has made substantial progress in handling data heterogeneity (Ahmed et al., 2025; Chen & Vikalo, 2024), while personalized FL further improves local adaptation (Qin et al., 2023; Zheng et al., 2025; Thapa & Li, 2025). However, most conventional FL methods assume homogeneous client models, making parameter aggregation infeasible when architectures differ substantially. Sub-model schemes aggregate compatible parameters under a shared backbone (Diao et al., 2021; Wu et al., 2024; Horváth et al., 2021), while alignment methods match model units before aggregation (Li et al., 2022; Wang et al., 2020). Such approaches become difficult to apply across fundamentally different architectures, motivating HtFL to seek architecture-agnostic knowledge carriers.

Proxy-model-based collaboration. A major line of HtFL introduces homogeneous auxiliary components to bridge heterogeneous client models (Sun et al., 2025; Zhu et al., 2021; Wang et al., 2024). FedKD (Wu et al., 2022) exchanges knowledge through a compact shared proxy model, while FedSCE (Zhang et al., 2025) further constrains auxiliary-model updates in adaptive subspaces to improve stability. FedMRL (Yi et al., 2024) employs a shared homogeneous auxiliary model and representation fusion to facilitate cross-model knowledge transfer. More recently, pFedMoE (Yi et al., 2026) shares a homogeneous proxy feature extractor and combines it with heterogeneous local features through personalized sample-wise gating. Other approaches avoid real proxy data by generating synthetic supervision: FedGen (Zhu et al., 2021) learns a conditional generator, while FedZGE (Ma et al., 2025) updates server-side synthesis through zeroth-order feedback from client predictions. Although flexible across architectures, these methods introduce auxiliary models, synthetic-data generation, or additional knowledge-transfer modules, increasing computation or communication overhead.

Prototype-based collaboration. Prototype-based methods (Qi et al., 2025; Guo et al., 2025) instead exchange compact class-level representations. FedProto (Tan et al., 2022) aggregates local class prototypes as global references, FedTGP (Zhang et al., 2024) learns trainable global prototypes with margin-based contrastive learning, and FedSA (Zhou et al., 2025) introduces semantic anchors to improve representation consistency. FedRE (Yao et al., 2026) further entangles local class prototypes and their label encodings into compact cross-category representations to train a global classifier. These methods are communication-efficient, but their shared knowledge is ultimately constructed from discrete class-level representations. Under severe model heterogeneity, such representations can heavily compress intra-class variation, while aggregation or alignment across incompatible feature geometries may yield ambiguous supervision.

A.2 Continuous Function Modeling

Continuous function modeling represents multidimensional signals as queryable coordinate-to-value functions, enabling fine-grained signal representation. Representative approaches include Fourier feature mappings (Tancik et al., 2020), periodic activations (Sitzmann et al., 2020), and local implicit decoding (Chen et al., 2021). Low-rank continuous modeling further extends classical tensor decompositions to function spaces. LRTFR (Luo et al., 2024) combines one-dimensional factor networks with compact core tensors to represent low-rank tensor functions, while F-INR (Vemuri et al., 2026) and CP-INR (Cheng et al., 2025) introduce controllable low-rank structures for compact representation and efficient recovery.

Continuous representations have also appeared in federated settings, but with different objectives. Federated neural-field methods mainly treat neural fields themselves as the models or reconstruction targets to be collaboratively trained (Yun et al., 2025; Holden et al., 2023), rather than using continuous functions as shared knowledge carriers across heterogeneous architectures. In contrast, inspired by low-rank tensor function modeling, FedCF represents each client's class-conditional feature knowledge as a compact continuous function and aligns and aggregates these functions in a canonical space. Thus, continuous functions serve directly as the collaboration carrier for heterogeneous federated learning.

B LOCAL CONTINUOUS FIELD IMPLEMENTATION

B.1 BACKBONE FEATURE GRID CONSTRUCTION

Each client i maps an input image x to a feature tensor $\Phi_i(x) \in \mathbb{R}^{H_i \times W_i \times d_i}$, which provides discrete supervision signals for fitting the reconstruction objective in the main text. Note that (H_i, W_i, d_i) are architecture-dependent.

CNN backbones. For CNN-based backbones, we take the activation map from the last convolutional stage as $\Phi_i(x)$, yielding $H_i \times W_i$ spatial locations. On CIFAR-style inputs, architectures such as AlexNet and ResNet typically produce $H_i = W_i = 4$, while the channel dimension d_i may differ across backbones. Such channel mismatches are handled by the feature translator to unify representations in the canonical space.

ViT backbones. For ViTs, we reshape the patch-token sequence into an $H_i \times W_i$ grid and permute dimensions to match the tensor form of $\Phi_i(x)$. On CIFAR inputs, with patch size $\mathcal{P}$, we have $H_i = W_i = 32/\mathcal{P}$ (e.g., $\mathcal{P} = 4 \Rightarrow H_i = W_i = 8$).

Why a grid, not a single vector. A core challenge in heterogeneous federated learning is that different backbones encode semantics at different spatial granularities. CNNs emphasize local spatial structure, while ViTs rely on more global token mixing. A single class mean prototype collapses intra-class variations, whereas the feature grid $\Phi_i(x)$ preserves structured and rich within-class information. Our continuous field learns a smooth function over this native grid geometry, enabling transferable knowledge expression without forcing incompatible cross-architecture feature averaging.

Benefits of local fields under heterogeneity. Local continuous fields learn *within-architecture* structure. They fit each client's native representation geometry instead of mixing coordinate systems across heterogeneous backbones. This avoids the typical failure mode of heterogeneous prototype averaging, where semantically non-commensurate features are naively merged, leading to semantic mismatch. Cross-architecture interaction is deferred to the canonical space, where translators and the global field perform alignment, making collaboration more architecture-friendly.

B.2 POINT SAMPLING

For each training example $(x, y) \sim D_i$, instead of using all $H_i W_i$ grid locations, we uniformly sample a subset $\Omega(x) \subseteq \{0, \ldots, H_i - 1\} \times \{0, \ldots, W_i - 1\}$ without replacement with a fixed budget $|\Omega(x)| = K$. For each sampled location $(h, w) \in \Omega(x)$, we (i) map the discrete index to continuous coordinates (u, v), (ii) extract the local feature vector $z = \Phi_i(x)_{h,w,:} \in \mathbb{R}^{d_i}$, and append the tuple $((u, v), z, y)$ to the regression set D_i^s.

Unbiased approximation and efficiency. Let the full-grid reconstruction loss average over all $H_i W_i$ locations. Since $\Omega(x)$ is sampled uniformly without replacement, the sampled average reconstruction error is an *unbiased estimator* of the full-grid objective, i.e., its expectation equals the full-grid loss. Thus, point sampling preserves the training target in expectation while reducing computation. When $H_i = W_i = 4$ (only 16 locations), using the full grid incurs limited overhead. However, for $H_i = W_i = 8$ (64 locations), the cost of full-grid fitting grows noticeably, making the benefit of point sampling substantially more pronounced (and the gap further widens for larger grids).

B.3 LOCAL FIELD FITTING IMPLEMENTATION

When fitting the local continuous field, we freeze the backbone parameters that produce the feature grid $\Phi_i(x)$ and optimize only the field parameters θ_i^S. The key reason is that the regression targets of the field are the backbone features themselves. If the backbone and the field were updated simultaneously, the supervision signal would keep changing, making the regression problem a moving-target and thus unstable. By freezing the backbone, we obtain a relatively stationary supervision signal, which enables learning a more robust field representation that can be reused across communication rounds.

Low-rank parameterization. For each class c, the continuous field $S_{i,c}(u, v)$ is implemented via a low-rank factorization. Intuitively, it uses a small number of 1D coordinate factors (along the u and v axes) together with a compact class core to generate the feature vector at any coordinate (u, v), avoiding an explicit dense lookup table of size $H_i \times W_i \times d_i$. This parameterization provides: (i) continuity and interpolation, producing smooth predictions at unsampled locations; (ii) controllable capacity, where the rank controls the function complexity and mitigates overfitting when local data are limited; and (iii) storage/communication efficiency, because one shared function summarizes spatial features from many local examples, while its parameter size is controlled by the field ranks and does not scale with the number of local samples.

Feature caching and multi-epoch fitting. During local training, we cache the generated feature grids $\Phi_i(x)$. After the local update, the backbone is frozen and the cached features are reused for multi-epoch regression over sampled point sets $\Omega(x)$. This avoids additional backbone forward passes and reduces field-fitting cost.

C Global Continuous Field Implementation

C.1 Global Data Construction

Canonical space and translators. Each client's local feature space $\mathbb{R}^{d_i}$ is mapped to a unified canonical space $\mathbb{R}^d$ via a feature translator A_i, and local coordinates (u, v) are mapped to canonical coordinates (s, t) via a coordinate translator ψ_i. Therefore, queries to any local continuous field can be converted into supervised samples in the canonical space.

Sampling protocol. For each available client–class continuous field, we uniformly sample K_{core} coordinates $(u, v) \sim \text{Unif}([0, 1]^2)$ and construct the canonical coordinate $(s, t) = \psi_i(u, v)$, the canonical feature $z^{\text{feat}}_{i,c} = A_i\big(S_{i,c}(u, v)\big)$, and the label c. We then append the tuple $((s, t), z^{\text{feat}}_{i,c}, c)$ to D^g.

Why this is data-free, heterogeneity-safe, and efficient. Each sample in D^g requires only a single query to the local field $S_{i,c}$ followed by translation, so the server can arbitrarily resample and expand D^g (and D^{cls}) to synthesize as many labeled canonical features as needed at low cost, without public data and without ever accessing raw client examples. Cross-architecture interaction occurs only through the translators (ψ_i, A_i) and the canonical field S_c, rather than directly mixing incompatible representation coordinates, thereby avoiding semantic mismatch and collapse induced by naive prototype/feature averaging. Moreover, the generated data contain only low-dimensional canonical features and labels, which are lightweight to store compared to images or dense feature maps, making the overall pipeline scalable and efficient.

C.2 Training the Global Field and Translators

The server minimizes the regression error on canonical samples, i.e., it fits the global canonical field $S_c(s, t; \theta^G)$ to the canonical features translated from heterogeneous clients.

Alternating updates. We adopt an alternating (block-coordinate) optimization scheme: ***Step-1*** updates the global field parameters θ^G with translators fixed, so that S_c better matches the current canonically-aligned supervision; ***Step-2*** updates the translators (ψ_i, A_i) with the global field fixed, so that each client's local field aligns better to the fixed canonical "semantic template" in the canonical space. Joint end-to-end optimization of θ^G and (ψ_i, A_i) can lead to unstable coupled updates, so we use the above alternating updates: fix the translators to learn the canonical template (update θ^G), then fix the template to refine alignment (update (ψ_i, A_i)). In implementation, θ^G is updated every round with a small number of gradient steps, while the translators are updated once every R_{tr} rounds to further stabilize alignment.

C.3 Advantages of the Global Feature Field and the Global Classifier

Why a global field is better than a global prototype. For client i, the key to absorbing global knowledge is that, given a local feature and its class label, the server can return a teacher representation

for alignment, thereby guiding the update of local representations. If the server only provides a single-point class prototype μ_c, then all samples of class c, regardless of which sub-mode they belong to (e.g., pose or background), are forced to contract toward the same center. This collapses an inherently diverse class-conditional structure into a single point and imposes the same alignment direction on all samples. Samples near the dominant mode may benefit, but those from non-dominant sub-modes are often pulled toward incorrect regions, flattening intra-class structure and becoming a common source of boundary confusion under severe heterogeneity.

The advantage of a global field $S_c(s,t)$ is that its supervision varies over canonical spatial coordinates. It upgrades one anchor per class to a queryable structure per class. Specifically, the client first maps its local coordinate (u,v) to a canonical coordinate (s,t) via ψ_i, and then queries $S_c(s,t)$ to obtain the corresponding teacher representation at that location. Thus, the teacher is no longer a fixed class center but a coordinate-dependent class-conditional target that varies with (s,t). Different canonical positions within the same class can therefore receive different targets, rather than being uniformly aligned to a single global center. In other words, a prototype cannot explicitly represent spatially varying intra-class structure, whereas field supervision provides coordinate-dependent alignment, enabling knowledge sharing while preserving such structure, which is particularly suited for collaboration under severe heterogeneity.

Why the global classifier helps clients. Under severe non-IID data and severe model heterogeneity, each local classifier only observes a biased distribution and thus learns fragmented, mutually inconsistent decision boundaries. We leverage a global classifier to capture cross-client boundary structure, such as inter-class relative relations and confusion directions. We then distill this global decision knowledge in the canonical space back to all clients, working together with global feature alignment to stably correct client-side boundary drift induced by non-IID data. Unlike approaches that require training a heavy proxy model on the server or synthesizing images, our teacher is simply a small global decision module operating in the canonical space, enabling effective cross-architecture knowledge transfer with low communication overhead.

D Overall Algorithm

Algorithm 1 summarizes the full FedCF pipeline, aligned with the method equations in the main text: local field fitting (3), global field training (5), global classifier training (6), and local absorption (7), (8), (9).

E Proofs for Section 2.3

E.1 Proof of Theorem 2.5

Proof. We prove the Tucker factorization and the induced spatial rank bounds, which provide a theoretical guarantee on the communication efficiency of sharing the knowledge representation. Throughout the proof, we fix a client i and a class c.

Expand the field at sampled coordinates. The $S_{i,c}(u,v)$ is parameterized by the low-rank form in Eq. (2), with spatial ranks (r_u, r_v) and core $G_{i,c} \in \mathbb{R}^{r_u \times r_v \times d_i}$. By the definition of the mode-wise product, for any $(u,v) \in [0,1]^2$ and any channel index $k \in \{1,\dots,d_i\}$, we have the entrywise expansion

$$S_{i,c}(u,v)[k] = \sum_{a=1}^{r_u}\sum_{b=1}^{r_v} G_{i,c}[a,b,k]\,\big(\phi_u(u)\big)_a\,\big(\phi_v(v)\big)_b. \tag{11}$$

Fix any coordinate sets $\{u_\alpha\}_{\alpha=1}^{n_u} \subset [0,1]$ and $\{v_\beta\}_{\beta=1}^{n_v} \subset [0,1]$, and define the sampled tensor $\mathbf{S}_{i,c}^{n_u\times n_v} \in \mathbb{R}^{n_u\times n_v\times d_i}$ by $\mathbf{S}_{i,c}^{n_u\times n_v}[\alpha,\beta,:] = S_{i,c}(u_\alpha, v_\beta)$. Substituting $(u,v) = (u_\alpha, v_\beta)$ into Eq. (11) gives, for every (α,β,k),

$$\mathbf{S}_{i,c}^{n_u\times n_v}[\alpha,\beta,k] = \sum_{a=1}^{r_u}\sum_{b=1}^{r_v} G_{i,c}[a,b,k]\,\big(\phi_u(u_\alpha)\big)_a\,\big(\phi_v(v_\beta)\big)_b. \tag{12}$$

Algorithm 1 FedCF

Input: clients $\{1, \dots, N\}$ with local datasets $\{D_i\}_{i=1}^N$; local model $\{f_i(\cdot; \theta_i)\}$; rounds T; local steps E.
Output: client models $\{\theta_i\}_{i=1}^N$; local field parameters $\{\theta_i^S\}_{i=1}^N$; global field parameters θ^G; translators $\{A_i, \psi_i\}_{i=1}^N$; global classifier parameters θ^M.
Initialize:
for each client i **in parallel do**
    Initialize local model parameters θ_i.
    Train θ_i on D_i using L_i^{ce}, then fit the initial local fields by minimizing Eq. (3).
    Upload the initial local field parameters θ_i^S to the server.
end for
Initialize global field parameters θ^G (field form in Eq. (4)).
Initialize translators $\{A_i, \psi_i\}_{i=1}^N$ (or shared within each architecture group, as described in Sec. 2.2.2).
Initialize θ^M, then fit the initial global field and translators and train θ^M.
for round $0, 1, \dots, T-1$ **do**
    Server $\rightarrow$ Clients: broadcast θ^G and θ^M, and provide each client with its translator parameters (A_i, ψ_i).
    Client updates:
    for each client $i \in \{1, \dots, N\}$ **do**
        Update local model parameters θ_i for E steps on D_i by minimizing Eq. (9), which uses feature alignment Eq. (7) and distillation Eq. (8).
        Refit local field parameters θ_i^S by minimizing Eq. (3) using sampled points from backbone grids $\Phi_i(x)$.
        Upload local field parameters θ_i^S to the server.
    end for
    Server updates:
    Construct canonical regression samples for global-field training by querying local fields $\{\theta_i^S\}$ and translating them with $\{A_i, \psi_i\}$.
    Update global field parameters θ^G by minimizing Eq. (5).
    Periodically update translators $\{A_i, \psi_i\}$ with θ^G fixed.
    Train the global classifier θ^M by minimizing Eq. (6) on field-generated canonical features.
end for

Construct the factor matrices and obtain the Tucker form. Define the factor matrices $\mathbf{F}_u \in \mathbb{R}^{n_u \times r_u}$ and $\mathbf{F}_v \in \mathbb{R}^{n_v \times r_v}$ as

$$\mathbf{F}_u[\alpha, :] = \phi_u(u_\alpha)^\top, \qquad \mathbf{F}_v[\beta, :] = \phi_v(v_\beta)^\top,$$

i.e., $\mathbf{F}_u[\alpha, a] = (\phi_u(u_\alpha))_a$ and $\mathbf{F}_v[\beta, b] = (\phi_v(v_\beta))_b$. Then Eq. (12) can be rewritten as

$$\mathbf{S}_{i,c}^{n_u \times n_v}[\alpha, \beta, k] = \sum_{a=1}^{r_u} \sum_{b=1}^{r_v} G_{i,c}[a, b, k]\, \mathbf{F}_u[\alpha, a]\, \mathbf{F}_v[\beta, b]. \tag{13}$$

By definition of the Tucker product, Eq. (13) is exactly the entrywise form of

$$\mathbf{S}_{i,c}^{n_u \times n_v} = G_{i,c} \times_1 \mathbf{F}_u \times_2 \mathbf{F}_v, \tag{14}$$

where $\times_n$ denotes the mode-n tensor–matrix product.

Spatial Tucker-rank upper bounds via unfoldings. We now show that the Tucker ranks of $\mathbf{S}_{i,c}^{n_u \times n_v}$ along the two spatial modes are at most r_u and r_v.

Let $\mathbf{S}_{(1)} \in \mathbb{R}^{n_u \times (n_v d_i)}$ denote the mode-1 unfolding of $\mathbf{S}_{i,c}^{n_u \times n_v}$, obtained by stacking all mode-1 fibers $\mathbf{S}_{i,c}^{n_u \times n_v}[:, \beta, k]$ as columns indexed by (β, k). Concretely, for each $\alpha \in \{1, \dots, n_u\}$ and each column index (β, k),

$$\mathbf{S}_{(1)}[\alpha, (\beta, k)] = \mathbf{S}_{i,c}^{n_u \times n_v}[\alpha, \beta, k]. \tag{15}$$

Similarly, define the mode-2 unfolding $\mathbf{S}_{(2)} \in \mathbb{R}^{n_v \times (n_u d_i)}$ by

$$\mathbf{S}_{(2)}[\beta, (\alpha, k)] = \mathbf{S}_{i,c}^{n_u \times n_v}[\alpha, \beta, k]. \tag{16}$$

We also define the unfoldings of the core $G_{i,c}$ as follows: $G_{(1)} \in \mathbb{R}^{r_u \times (r_v d_i)}$ stacks $G_{i,c}[:, b, k]$ as columns indexed by (b, k), i.e.,

$$G_{(1)}[a, (b, k)] = G_{i,c}[a, b, k], \tag{17}$$

and $G_{(2)} \in \mathbb{R}^{r_v \times (r_u d_i)}$ stacks $G_{i,c}[a, :, k]$ as columns indexed by (a, k), i.e.,

$$G_{(2)}[b, (a, k)] = G_{i,c}[a, b, k]. \tag{18}$$

We first derive the mode-1 unfolding identity. For each α and (β, k), Eq. (13) can be expanded as

$$\begin{aligned}
\mathbf{S}_{(1)}[\alpha, (\beta, k)] &= \mathbf{S}_{i,c}^{n_u \times n_v}[\alpha, \beta, k] \\
&= \sum_{a=1}^{r_u} \sum_{b=1}^{r_v} G_{i,c}[a, b, k]\, \mathbf{F}_u[\alpha, a]\, \mathbf{F}_v[\beta, b] \\
&= \sum_{a=1}^{r_u} \mathbf{F}_u[\alpha, a] \sum_{b=1}^{r_v} G_{i,c}[a, b, k] \mathbf{F}_v[\beta, b].
\end{aligned} \tag{19}$$

Now define the matrix $\mathbf{M} \in \mathbb{R}^{r_v d_i \times n_v d_i}$ by

$$\mathbf{M}[(b, k), (\beta, k')] = \begin{cases} \mathbf{F}_v[\beta, b], & \text{if } k' = k, \\ 0, & \text{otherwise,} \end{cases} \tag{20}$$

where (b, k) indexes rows with $b \in [r_v]$ and $k \in [d_i]$, and (β, k') indexes columns with $\beta \in [n_v]$ and $k' \in [d_i]$. Hence, each (b, β) block of $\mathbf{M}$ equals $\mathbf{F}_v[\beta, b]\mathbf{I}_{d_i}$, and therefore

$$\mathbf{M} = (\mathbf{F}_v \otimes \mathbf{I}_{d_i})^\top.$$

Using Eq. (17) and Eq. (20), we can rewrite the inner sum in Eq. (19) as

$$\sum_{b=1}^{r_v} G_{i,c}[a, b, k] \mathbf{F}_v[\beta, b] = \sum_{b=1}^{r_v} G_{(1)}[a, (b, k)]\, \mathbf{M}[(b, k), (\beta, k)].$$

Plugging this back into Eq. (19) yields

$$\begin{aligned}
\mathbf{S}_{(1)}[\alpha, (\beta, k)] &= \sum_{a=1}^{r_u} \mathbf{F}_u[\alpha, a] \sum_{(b,k')} G_{(1)}[a, (b, k')]\, \mathbf{M}[(b, k'), (\beta, k)] \\
&= \big(\mathbf{F}_u\, G_{(1)}\, \mathbf{M}\big)[\alpha, (\beta, k)].
\end{aligned} \tag{21}$$

Therefore,

$$\mathbf{S}_{(1)} = \mathbf{F}_u\, G_{(1)}\, (\mathbf{F}_v \otimes \mathbf{I}_{d_i})^\top. \tag{22}$$

We then derive the mode-2 unfolding identity. An entirely symmetric argument starting from Eq. (13) and the definition Eq. (16) gives

$$\mathbf{S}_{(2)} = \mathbf{F}_v\, G_{(2)}\, (\mathbf{F}_u \otimes \mathbf{I}_{d_i})^\top. \tag{23}$$

Conclude the rank bounds. Using Eq. (22)–Eq. (23) and the basic inequality $\operatorname{rank}(\mathbf{ABC}) \le \min\{\operatorname{rank}(\mathbf{A}), \operatorname{rank}(\mathbf{B}), \operatorname{rank}(\mathbf{C})\}$, we obtain

$$\operatorname{rank}(\mathbf{S}_{(1)}) \le \operatorname{rank}(\mathbf{F}_u) \le r_u, \qquad \operatorname{rank}(\mathbf{S}_{(2)}) \le \operatorname{rank}(\mathbf{F}_v) \le r_v.$$

By definition of the Tucker rank, this shows that the spatial Tucker ranks of $\mathbf{S}_{i,c}^{n_u \times n_v}$ along modes 1 and 2 are at most r_u and r_v, respectively. Since the choice of (n_u, n_v) and the coordinate sets $\{u_\alpha\}, \{v_\beta\}$ was arbitrary, the bounds hold for any sampling resolution $n_u \times n_v$. □

E.2 Proof of Corollary 2.6

Proof. We start from the local low-rank field parameterization in Eq. (2), which has a core tensor $G_{i,c} \in \mathbb{R}^{r_u \times r_v \times d_i}$ and two coordinate factor mappings $\phi_u : [0,1] \to \mathbb{R}^{r_u}$ and $\phi_v : [0,1] \to \mathbb{R}^{r_v}$.

Set $r_u = r_v = 1$. Then $G_{i,c}$ becomes a tensor of size $1 \times 1 \times d_i$. Meanwhile, choose constant coordinate factors $\phi_u(u) \equiv 1 \in \mathbb{R}^1$ and $\phi_v(v) \equiv 1 \in \mathbb{R}^1$ for all $u, v \in [0,1]$. By the definition of mode-wise tensor–vector products, contracting $G_{i,c}$ along its first two modes with scalar 1 yields

$$S_{i,c}(u,v) \;=\; G_{i,c} \times_1 \phi_u(u) \times_2 \phi_v(v) \;=\; G_{i,c} \times_1 1 \times_2 1 \;=\; G_{i,c}[1,1,:].$$

Therefore $S_{i,c}(u,v)$ is independent of (u,v), i.e., the field is spatially constant. Denote this constant vector by

$$\mu_{i,c} \;:=\; G_{i,c}[1,1,:] \in \mathbb{R}^{d_i},$$

so that

$$S_{i,c}(u,v) \equiv \mu_{i,c} \;\; \forall (u,v) \in [0,1]^2.$$

This shows that the rank-1 instantiation exactly reduces our field to a single class-wise vector, which is precisely the object communicated in prototype-based methods.

Consider an arbitrary sampling grid of size $n_u \times n_v$ on $[0,1]^2$ with coordinates $\{u_\alpha\}_{\alpha=1}^{n_u}$ and $\{v_\beta\}_{\beta=1}^{n_v}$. Define the sampled tensor $\mathbf{S}_{i,c}^{n_u \times n_v} \in \mathbb{R}^{n_u \times n_v \times d_i}$ by

$$\mathbf{S}_{i,c}^{n_u \times n_v}[\alpha, \beta, :] \;:=\; S_{i,c}(u_\alpha, v_\beta) \in \mathbb{R}^{d_i}.$$

Since $S_{i,c}(u,v) \equiv \mu_{i,c}$ for all (u,v), every spatial location shares the same feature vector, hence

$$\mathbf{S}_{i,c}^{n_u \times n_v}[\alpha, \beta, :] \;=\; \mu_{i,c} \qquad \forall \alpha \in [n_u],\; \beta \in [n_v].$$

Equivalently, letting $\mathbf{1}_{n_u} \in \mathbb{R}^{n_u}$ and $\mathbf{1}_{n_v} \in \mathbb{R}^{n_v}$ be all-ones vectors, we can write

$$\mathbf{S}_{i,c}^{n_u \times n_v} \;=\; \mathcal{G}_{i,c} \times_1 \mathbf{1}_{n_u} \times_2 \mathbf{1}_{n_v},$$

where $\mathbf{1}_{n_u} \in \mathbb{R}^{n_u}$ and $\mathbf{1}_{n_v} \in \mathbb{R}^{n_v}$ are all-ones vectors, and the core tensor $\mathcal{G}_{i,c} \in \mathbb{R}^{1 \times 1 \times d_i}$ satisfies $\mathcal{G}_{i,c}[1,1,:] = \mu_{i,c} \in \mathbb{R}^{d_i}$. Equivalently, $\mathbf{S}_{i,c}^{n_u \times n_v}[\alpha, \beta, :] = \mu_{i,c}$ for all $\alpha \in [n_u]$ and $\beta \in [n_v]$.

Therefore the mode-1 unfolding satisfies $\mathbf{S}_{(1)} = \mathbf{1}_{n_u}(\mathbf{1}_{n_v} \otimes \mu_{i,c})^\top$ and has matrix rank 1; similarly, the mode-2 unfolding has rank 1. Hence the Tucker ranks along the two spatial modes are $(1,1)$.

A prototype-based method represents each client–class pair (i,c) by a single vector (denote it by $\mu_{i,c}$) and uses it as the class-wise shared carrier. Thus, this is exactly the special case of Eq. (2) with $(r_u, r_v) = (1,1)$ and constant coordinate factors, whose sampled tensors are spatially constant and have spatial Tucker ranks $(1,1)$ at any resolution. This completes the proof. □

E.3 Proof of Lemma 2.7

Proof. Here, $\mathbb{E}[\cdot]$ is taken over the spatial coordinates (u,v) on the normalized domain. Consider the prototype fitting objective

$$J(\mu_{i,c}) \;:=\; \mathbb{E}\big\|S_{i,c}^*(u,v) - \mu_{i,c}\big\|_2^2, \tag{24}$$

which measures how well a constant vector $\mu_{i,c}$ approximates the spatially varying ideal field $S_{i,c}^*(u,v)$ in mean squared error.

To identify the best prototype and the resulting irreducible error, we add and subtract the spatial mean $\bar{S}_{i,c}^* := \mathbb{E}[S_{i,c}^*(u,v)]$ inside the squared norm:

$$S_{i,c}^*(u,v) - \mu_{i,c} = \big(S_{i,c}^*(u,v) - \bar{S}_{i,c}^*\big) + \big(\bar{S}_{i,c}^* - \mu_{i,c}\big).$$

Squaring and expanding the norm gives variance term, bias term, and a cross term:

$$\begin{aligned}\big\|S_{i,c}^*(u,v) - \mu_{i,c}\big\|_2^2 = \big\|S_{i,c}^*(u,v) - \bar{S}_{i,c}^*\big\|_2^2 + \big\|\bar{S}_{i,c}^* - \mu_{i,c}\big\|_2^2 \\ + 2\Big\langle S_{i,c}^*(u,v) - \bar{S}_{i,c}^*,\; \bar{S}_{i,c}^* - \mu_{i,c}\Big\rangle.\end{aligned} \tag{25}$$

We now take expectation with respect to (u, v). The key observation is that the cross term vanishes, because the first factor has zero mean: $\mathbb{E}\big[S^*_{i,c}(u,v) - \bar{S}^*_{i,c}\big] = 0$, and $\bar{S}^*_{i,c} - \mu_{i,c}$ is deterministic constant in (u, v). Therefore, applying $\mathbb{E}[\cdot]$ to (25) yields the standard bias–variance decomposition

$$J(\mu_{i,c}) = \underbrace{\mathbb{E}\big\|S^*_{i,c}(u,v) - \bar{S}^*_{i,c}\big\|_2^2}_{:=\,\mathrm{Var}(S^*_{i,c}(u,v))} + \big\|\mu_{i,c} - \bar{S}^*_{i,c}\big\|_2^2. \tag{26}$$

The first term in Eq. (26) depends only on the intrinsic spatial variability of the field, while the second term is a nonnegative penalty measuring how far the chosen prototype $\mu_{i,c}$ is from the optimal one. Consequently, the minimum of $J(\mu_{i,c})$ is attained at $\mu_{i,c} = \bar{S}^*_{i,c}$ and the minimal value equals

$$\min_{\mu_{i,c}} J(\mu_{i,c}) = \mathrm{Var}\big(S^*_{i,c}(u,v)\big),$$

which proves $\epsilon^2_{\mathrm{proto}} = \mathrm{Var}(S^*_{i,c}(u,v))$.

By definition, ϵ_r^2 is the best mean-squared approximation error to $S^*_{i,c}$ achievable by our field model in Eq. (2), i.e.,

$$\epsilon_r^2 \;:=\; \inf_{\theta_i^S} \mathbb{E}\big\|S^*_{i,c}(u,v) - S_{i,c}(u,v;\theta_i^S)\big\|_2^2. \tag{27}$$

More generally, for any $r_u, r_v \geq 1$, a constant field $S_{i,c}(u,v) \equiv \mu_{i,c}$ is feasible in Eq. (2) by setting $\phi_u(u) \equiv \mathbf{e}_1$, $\phi_v(v) \equiv \mathbf{e}_1$, $G_{i,c}[1,1,:] = \mu_{i,c}$, and all remaining core entries to zero. Therefore, the feasible set over which the infimum in ϵ_r^2 is taken includes all constant fields used to define $\epsilon^2_{\mathrm{proto}}$. By the standard monotonicity of the infimum under set inclusion, we obtain

$$\epsilon_r^2 \leq \inf_{\mu_{i,c}} \mathbb{E}\big\|S^*_{i,c}(u,v) - \mu_{i,c}\big\|_2^2 = \epsilon^2_{\mathrm{proto}}.$$

This completes the proof. □

E.4 Proof of Theorem 2.8

Proof. During each communication round, the server-side teacher, including the global fields, translators, and global classifier, is fixed for the corresponding E local steps, but may change across rounds. Throughout the proof, $L_i(\theta)$ denotes the realizable objective induced by the current-round teacher, with the round index omitted for simplicity, whereas $L_i^*(\theta)$ denotes the fixed ideal objective induced by the population canonical field S^*.

Fix a client i and omit the client index in θ for brevity. We denote the iterates by $\{\theta_\tau\}_{\tau=0}^{TE-1}$, where each update is a gradient-based step with stepsize η:

$$\theta_{\tau+1} \;=\; \theta_\tau \;-\; \eta\,\hat{\nabla}_\tau. \tag{28}$$

Here $\hat{\nabla}_\tau$ is the stochastic gradient computed from a mini-batch of images, and within each image $K := |\Omega(x)|$ queried points used by the pointwise losses in Eqs. (7)-(8).

Stochastic gradient model and the two variance sources. At each step τ, the client samples a mini-batch $\mathcal{B}_\tau$ (size $B = |\mathcal{B}_\tau|$) of images, and for each image $x \in \mathcal{B}_\tau$, it samples K queried points $\Omega_\tau(x) = \{(u_{x,1}, v_{x,1}), \ldots, (u_{x,K}, v_{x,K})\}$. Let $\widehat{L}_{i,\tau}(\theta)$ be the corresponding empirical objective

$$\widehat{L}_{i,\tau}(\theta) \;:=\; \frac{1}{|\mathcal{B}_\tau|} \sum_{x \in \mathcal{B}_\tau} \frac{1}{K} \sum_{k=1}^{K} \ell(\theta; x, u_{x,k}, v_{x,k}),$$

We define the stochastic gradient estimator as

$$\hat{\nabla}_\tau := \nabla \widehat{L}_{i,\tau}(\theta_\tau),$$

where the gradient of the empirical objective is evaluated at the current iterate θ_τ.

Hereafter, conditioning on θ_τ is understood to also condition on the current-round teacher. We assume the standard unbiasedness condition with respect to the randomness of the mini-batch and point queries:

$$\mathbb{E}\Big[\hat{\nabla}_\tau \mid \theta_\tau\Big] \;=\; \nabla L_i(\theta_\tau). \tag{29}$$

Define the stochastic noise term

$$\xi_\tau \;:=\; \hat{\nabla}_\tau - \nabla L_i(\theta_\tau), \tag{30}$$

so that $\mathbb{E}[\xi_\tau \mid \theta_\tau] = 0$ by Eq. (29). We next explain the two variance contributions and how the factor $1/K$ arises.

(i) Point-query noise per image. Fix an image x and consider the pointwise part of the loss as an expectation over queried coordinates $(u, v) \in \Omega(x)$. Let $\nabla_\theta \bar{\ell}(\theta; x)$ denote the per-image gradient obtained by taking the expectation over the queried point:

$$\nabla_\theta \bar{\ell}(\theta; x) := \mathbb{E}_{(u,v)\sim\Omega(x)}\big[\nabla_\theta \ell(\theta; x, u, v)\big].$$

For the variance analysis, consider the corresponding with-replacement estimator. Let the K-point Monte-Carlo estimator be

$$\widehat{\nabla}_\theta \bar{\ell}(\theta; x) := \frac{1}{K}\sum_{k=1}^{K} \nabla_\theta \ell(\theta; x, u_k, v_k), \quad (u_k, v_k) \overset{i.i.d.}{\sim} \Omega(x).$$

Thus, by linearity of expectation, $\mathbb{E}[\widehat{\nabla}_\theta \bar{\ell}(\theta; x) \mid x] = \nabla_\theta \bar{\ell}(\theta; x)$. Moreover, under conditional independence of the K queried points given x,

$$\begin{aligned}\mathbb{E}\Big[\big\|\widehat{\nabla}_\theta \bar{\ell}(\theta; x) - \nabla_\theta \bar{\ell}(\theta; x)\big\|_2^2 \,\Big|\, x\Big] &= \mathbb{E}\Big[\Big\|\frac{1}{K}\sum_{k=1}^{K}\Big(\nabla_\theta \ell(\theta; x, u_k, v_k) - \nabla_\theta \bar{\ell}(\theta; x)\Big)\Big\|_2^2 \,\Big|\, x\Big] \\ &= \frac{1}{K}\,\mathbb{E}\Big[\big\|\nabla_\theta \ell(\theta; x, u, v) - \nabla_\theta \bar{\ell}(\theta; x)\big\|_2^2 \,\Big|\, x\Big] \;\le\; \frac{\Sigma^2}{K}, \end{aligned} \tag{31}$$

where the equality follows from independence and zero-mean of the summands, and Σ^2 is a uniform upper bound on the per-point conditional variance.

(ii) Mini-batch sampling noise. Even if the per-image expected gradient $\nabla_\theta \bar{\ell}(\theta; x)$ were computed exactly, using a finite mini-batch still introduces sampling noise. Let

$$\widehat{L}_{i,\mathrm{mb}}(\theta) \;:=\; \frac{1}{|\mathcal{B}|}\sum_{x\in\mathcal{B}} \bar{\ell}(\theta; x), \qquad \bar{\ell}(\theta; x) \;:=\; \mathbb{E}_{(u,v)\sim\Omega(x)}\big[\ell(\theta; x, u, v)\big],$$

so that $\nabla \widehat{L}_{i,\mathrm{mb}}(\theta)$ averages the per-image expected gradients over the mini-batch. We assume its mini-batch sampling variance is uniformly bounded:

$$\mathbb{E}\Big[\big\|\nabla \widehat{L}_{i,\mathrm{mb}}(\theta) - \nabla L_i(\theta)\big\|_2^2 \,\Big|\, \theta\Big] \;\le\; \Sigma_{\mathrm{mb}}^2. \tag{32}$$

Combining Eq. (31) and Eq. (32) yields the total variance bound

$$\mathbb{E}\big[\|\xi_\tau\|_2^2 \mid \theta_\tau\big] \;\le\; \Sigma_{\mathrm{mb}}^2 \;+\; \frac{\Sigma^2}{K}. \tag{33}$$

Teacher approximation induces a bounded gradient bias. Because L_i is induced by the learned teacher whereas L_i^* is induced by the ideal population teacher, their gradients may differ. Define the approximation-bias term

$$\delta_\tau \;:=\; \nabla L_i(\theta_\tau) - \nabla L_i^*(\theta_\tau). \tag{34}$$

By the theorem assumption, $\|\delta_\tau\|_2 \;=\; \|\nabla L_i(\theta_\tau) - \nabla L_i^*(\theta_\tau)\|_2 \;\le\; \kappa\,\epsilon_r$. Hence,

$$\|\delta_\tau\|_2^2 \le \kappa^2 \epsilon_r^2. \tag{35}$$

Using Eq. (30) and Eq. (34), we can rewrite the stochastic gradient used in the update as

$$\hat{\nabla}_\tau \;=\; \nabla L_i(\theta_\tau) + \xi_\tau \;=\; \nabla L_i^*(\theta_\tau) + \delta_\tau + \xi_\tau. \tag{36}$$

The term ξ_τ is zero-mean noise, while δ_τ is a deterministic bias caused by teacher approximation.

A smoothness descent inequality on the ideal objective L_i^*. By assumption, L_i^* is M-smooth. This means for any θ, θ',

$$L_i^*(\theta') \;\le\; L_i^*(\theta) + \langle \nabla L_i^*(\theta),\, \theta' - \theta\rangle + \frac{M}{2}\|\theta' - \theta\|_2^2. \tag{37}$$

We apply Eq. (37) at $\theta = \theta_\tau$ and $\theta' = \theta_{\tau+1}$. Using the gradient-based update Eq. (28), we obtain

$$L_i^*(\theta_{\tau+1}) \;\le\; L_i^*(\theta_\tau) \;-\; \eta\,\langle \nabla L_i^*(\theta_\tau),\, \hat{\nabla}_\tau\rangle \;+\; \frac{M\eta^2}{2}\|\hat{\nabla}_\tau\|_2^2. \tag{38}$$

Taking conditional expectation and expanding the terms. Let $a_\tau := \nabla L_i^*(\theta_\tau)$ for readability. We take conditional expectation of Eq. (38) given θ_τ and use the decomposition Eq. (36), i.e., $\hat{\nabla}_\tau = a_\tau + \delta_\tau + \xi_\tau$.

The inner product term. Conditioned on θ_τ, δ_τ is deterministic, and $\mathbb{E}[\xi_\tau \mid \theta_\tau] = 0$. Hence,

$$\begin{aligned}\mathbb{E}\big[\langle a_\tau, \hat{\nabla}_\tau\rangle \mid \theta_\tau\big] &= \mathbb{E}\big[\langle a_\tau, a_\tau + \delta_\tau + \xi_\tau\rangle \mid \theta_\tau\big] \\ &= \langle a_\tau, a_\tau + \delta_\tau\rangle \;+\; \underbrace{\mathbb{E}[\langle a_\tau, \xi_\tau\rangle \mid \theta_\tau]}_{=0} \\ &= \|a_\tau\|_2^2 + \langle a_\tau, \delta_\tau\rangle. \end{aligned} \tag{39}$$

The squared norm term. Using $\hat{\nabla}_\tau = a_\tau + \delta_\tau + \xi_\tau$,

$$\begin{aligned}\mathbb{E}\big[\|\hat{\nabla}_\tau\|_2^2 \mid \theta_\tau\big] &= \mathbb{E}\big[\|a_\tau + \delta_\tau + \xi_\tau\|_2^2 \mid \theta_\tau\big] \\ &= \|a_\tau + \delta_\tau\|_2^2 \;+\; \mathbb{E}\big[\|\xi_\tau\|_2^2 \mid \theta_\tau\big] \;+\; 2\,\underbrace{\mathbb{E}\big[\langle a_\tau + \delta_\tau, \xi_\tau\rangle \mid \theta_\tau\big]}_{=0} \\ &= \|a_\tau\|_2^2 + 2\langle a_\tau, \delta_\tau\rangle + \|\delta_\tau\|_2^2 \;+\; \mathbb{E}\big[\|\xi_\tau\|_2^2 \mid \theta_\tau\big]. \end{aligned} \tag{40}$$

Plugging back and isolating $\|a_\tau\|_2^2$. Substituting Eq. (39) and Eq. (40) into the conditional expectation of Eq. (38) yields

$$\begin{aligned}\mathbb{E}\big[L_i^*(\theta_{\tau+1}) \mid \theta_\tau\big] \;&\le\; L_i^*(\theta_\tau) \;-\; \eta\Big(\|a_\tau\|_2^2 + \langle a_\tau, \delta_\tau\rangle\Big) \\ &\qquad +\; \frac{M\eta^2}{2}\Big(\|a_\tau\|_2^2 + 2\langle a_\tau, \delta_\tau\rangle + \|\delta_\tau\|_2^2 + \mathbb{E}[\|\xi_\tau\|_2^2 \mid \theta_\tau]\Big) \\ &=\; L_i^*(\theta_\tau) \;-\; \eta\Big(1 - \frac{M\eta}{2}\Big)\|a_\tau\|_2^2 \;-\; \eta(1 - M\eta)\langle a_\tau, \delta_\tau\rangle \\ &\qquad +\; \frac{M\eta^2}{2}\|\delta_\tau\|_2^2 \;+\; \frac{M\eta^2}{2}\mathbb{E}[\|\xi_\tau\|_2^2 \mid \theta_\tau]. \end{aligned} \tag{41}$$

At this point, the only nonstandard term is the mixed inner product $\langle a_\tau, \delta_\tau\rangle$. Its sign is unknown, so we upper bound it by its magnitude using Young's inequality.

Controlling the mixed term via Young's inequality. Assume $\eta \le \frac{1}{2M}$, which implies $1 - M\eta \ge \frac{1}{2} \ge 0$. We bound

$$-\langle a_\tau, \delta_\tau\rangle \;\le\; |\langle a_\tau, \delta_\tau\rangle| \;\le\; \frac{1}{2}\|a_\tau\|_2^2 + \frac{1}{2}\|\delta_\tau\|_2^2, \tag{42}$$

where the last inequality is the standard inequality $2|\langle u, v\rangle| \le \|u\|^2 + \|v\|^2$. Multiplying Eq. (42) by the nonnegative coefficient $\eta(1 - M\eta)$ yields

$$-\eta(1 - M\eta)\langle a_\tau, \delta_\tau\rangle \;\le\; \frac{\eta(1 - M\eta)}{2}\|a_\tau\|_2^2 \;+\; \frac{\eta(1 - M\eta)}{2}\|\delta_\tau\|_2^2. \tag{43}$$

Plugging Eq. (43) into Eq. (41) and simplifying, we get

$$\begin{aligned}\mathbb{E}\big[L_i^*(\theta_{\tau+1}) \mid \theta_\tau\big] \;&\le\; L_i^*(\theta_\tau) \;-\; \eta\Big(1 - \frac{M\eta}{2} - \frac{1 - M\eta}{2}\Big)\|a_\tau\|_2^2 \\ &\qquad +\; \Big(\frac{\eta(1 - M\eta)}{2} + \frac{M\eta^2}{2}\Big)\|\delta_\tau\|_2^2 \;+\; \frac{M\eta^2}{2}\mathbb{E}[\|\xi_\tau\|_2^2 \mid \theta_\tau]. \end{aligned}$$

Therefore,

$$\mathbb{E}\big[L_i^*(\theta_{\tau+1}) \mid \theta_\tau\big] \;\le\; L_i^*(\theta_\tau) \;-\; \frac{\eta}{2}\|\nabla L_i^*(\theta_\tau)\|_2^2 \;+\; \frac{\eta}{2}\|\delta_\tau\|_2^2 \;+\; \frac{M\eta^2}{2}\mathbb{E}[\|\xi_\tau\|_2^2 \mid \theta_\tau]. \tag{44}$$

Intuitively, Eq. (44) says that *the ideal objective L_i^* decreases proportionally to the squared gradient norm, up to two penalties: one from the teacher approximation bias δ_τ, and one from stochastic optimization noise ξ_τ.*

Removing conditioning and summing over τ. Taking total expectation of Eq. (44) gives

$$\mathbb{E}\big[L_i^*(\theta_{\tau+1})\big] \;\le\; \mathbb{E}\big[L_i^*(\theta_\tau)\big] \;-\; \frac{\eta}{2}\mathbb{E}\big[\|\nabla L_i^*(\theta_\tau)\|_2^2\big] \;+\; \frac{\eta}{2}\mathbb{E}\big[\|\delta_\tau\|_2^2\big] \;+\; \frac{M\eta^2}{2}\mathbb{E}\big[\|\xi_\tau\|_2^2\big]. \tag{45}$$

Rearranging Eq. (45) yields

$$\frac{\eta}{2}\mathbb{E}\big[\|\nabla L_i^*(\theta_\tau)\|_2^2\big] \;\le\; \mathbb{E}\big[L_i^*(\theta_\tau)\big] - \mathbb{E}\big[L_i^*(\theta_{\tau+1})\big] \;+\; \frac{\eta}{2}\mathbb{E}\big[\|\delta_\tau\|_2^2\big] \;+\; \frac{M\eta^2}{2}\mathbb{E}\big[\|\xi_\tau\|_2^2\big]. \tag{46}$$

Summing Eq. (46) over $\tau = 0, 1, \ldots, TE-1$ gives a telescoping sum on the objective:

$$\begin{aligned}\frac{\eta}{2}\sum_{\tau=0}^{TE-1}\mathbb{E}\big[\|\nabla L_i^*(\theta_\tau)\|_2^2\big] \;\le\; \mathbb{E}\big[L_i^*(\theta_0)\big] - \mathbb{E}\big[L_i^*(\theta_{TE})\big] \\ +\; \frac{\eta}{2}\sum_{\tau=0}^{TE-1}\mathbb{E}\big[\|\delta_\tau\|_2^2\big] \;+\; \frac{M\eta^2}{2}\sum_{\tau=0}^{TE-1}\mathbb{E}\big[\|\xi_\tau\|_2^2\big].\end{aligned} \tag{47}$$

Because $L_i^*(\theta) \ge \inf_\theta L_i^*(\theta)$ for all θ, we can lower bound $\mathbb{E}[L_i^*(\theta_{TE})] \ge \inf_\theta L_i^*(\theta)$ and obtain

$$\mathbb{E}\big[L_i^*(\theta_0)\big] - \mathbb{E}\big[L_i^*(\theta_{TE})\big] \;\le\; \mathbb{E}\big[L_i^*(\theta_0)\big] - \inf_\theta L_i^*(\theta) \;=:\; \Delta,$$

where Δ is the initial suboptimality of the ideal objective. It is the quantity that naturally appears when telescoping the decrease of L_i^*.

Substituting the bias and variance bounds. Using Eq. (35) and Eq. (33), we have for every τ:

$$\mathbb{E}\big[\|\delta_\tau\|_2^2\big] \le \kappa^2\epsilon_r^2, \qquad \mathbb{E}\big[\|\xi_\tau\|_2^2\big] \le \Sigma_{\mathrm{mb}}^2 + \frac{\Sigma^2}{K}.$$

Plugging these into Eq. (47) yields

$$\frac{\eta}{2}\sum_{\tau=0}^{TE-1}\mathbb{E}\big[\|\nabla L_i^*(\theta_\tau)\|_2^2\big] \;\le\; \Delta \;+\; \frac{\eta}{2}\cdot TE\cdot\kappa^2\epsilon_r^2 \;+\; \frac{M\eta^2}{2}\cdot TE\cdot\Big(\Sigma_{\mathrm{mb}}^2 + \frac{\Sigma^2}{K}\Big). \tag{48}$$

Dividing both sides of Eq. (48) by $\eta TE/2$ gives the averaged bound

$$\frac{1}{TE}\sum_{\tau=0}^{TE-1}\mathbb{E}\big[\|\nabla L_i^*(\theta_\tau)\|_2^2\big] \;\le\; \frac{2\Delta}{\eta TE} \;+\; M\eta\Sigma_{\mathrm{mb}}^2 + \frac{M\eta\Sigma^2}{K} \;+\; \kappa^2\epsilon_r^2. \tag{49}$$

This completes the proof. □

F Communication and computation overhead

F.1 Complexity of Local Computation

This appendix reports the client local computation cost in terms of FLOPs. We follow the standard FLOPs accounting used in deep learning systems. Let $\mathcal{F}$ denote the *forward-pass* FLOPs of a network on a minibatch; then one training update (forward + backward) costs approximately $3\mathcal{F}$, since the backward pass is about twice the forward cost for common architectures (Huo et al., 2018). The notation used in this subsection is summarized as follows. For client i, we define:

- n_i: local training samples; B: minibatch size; E: local epochs per round.
- $U_i := E\lceil \frac{n_i}{B} \rceil$: local training steps per round.
- $\mathcal{F}_i$: forward FLOPs of the client model f_i on one minibatch of size B.
  (Thus one local training step on f_i costs $\approx 3\mathcal{F}_i$.)
- $\mathcal{F}_{\mathrm{aux}}$: forward FLOPs of an **auxiliary/proxy model** trained on the client.
- $\mathcal{F}_{\mathrm{pex}}$: forward FLOPs of the pFedMoE shared proxy feature extractor.
- $\mathcal{F}_{\mathrm{gate}}$: forward FLOPs of the pFedMoE personalized gating network.

Table 2: Theoretical per-round **client-side** local computation complexity in FLOPs (dominant terms). $U_i = E\lceil n_i/B \rceil$ is the number of local training steps per round, and one training step costs $\approx 3\times$ forward FLOPs.

Method	Per-round local FLOPs (dominant terms)	Overhead source
FedAvg / Local-only	$3U_i\mathcal{F}_i$	Single-model training
FedProto	$3U_i\mathcal{F}_i + O(U_i B d_p) + O(C_i d_p)$	Prototype distance
FedTGP	$3U_i\mathcal{F}_i + O(U_i B d_p) + O(C_i d_p)$	Prototype distance
FedSA	$3U_i\mathcal{F}_i + O(U_i B d_p) + O(C_i d_p)$	Anchor similarities
FedRE	$3U_i\mathcal{F}_i + U_i^{\mathrm{rep}}\mathcal{F}_i + O\big(n_i(d_r + C)\big)$	Extra feature extraction + RAP entanglement
FedKD	$3U_i\mathcal{F}_i + 3U_i\mathcal{F}_{\mathrm{aux}}$	Trains auxiliary model on-device
FedSCE	$3U_i\mathcal{F}_i + 3U_i\mathcal{F}_{\mathrm{aux}} + O(\frac{1}{\varrho} m_{\mathrm{svd}} r_{\mathrm{sub}}^2)$	Auxiliary model + subspace update
pFedMoE	$3U_i\big(\mathcal{F}_i + \mathcal{F}_{\mathrm{pex}} + \mathcal{F}_{\mathrm{gate}}\big) + O(U_i B d_p)$	Proxy extractor + sample-wise gating
FedGen	$3U_i\mathcal{F}_i + 3U_i\frac{B_G}{B}\mathcal{F}_i^h + U_i\,\mathcal{F}_G$	Extra synthetic minibatch + generator forward
FedZGE	$3U_i\mathcal{F}_i + (N_{\mathrm{dir}} + 1)U_i^{\mathrm{qry}}\mathcal{F}_i$	Forward-only black-box queries
FedCF	$3U_i\mathcal{F}_i + U_i\mathcal{F}_{\mathrm{teach}}(BK) + 3U_i^{\mathrm{lrcf}}\mathcal{F}_{\mathrm{lrcf}}$	Point-sampled forward + small field fitting

- $U_i^{\mathrm{rep}} := \lceil n_i/B \rceil = U_i/E$: number of additional representation-extraction minibatches in FedRE.
- d_r: dimensionality of the FedRE entangled representation ($d_r = 512$ in our implementation).
- $\mathcal{F}_G$: forward FLOPs of a **generator** executed on the client.
- d_p: dimension of class prototypes; C_i: # classes seen by client i.
- K: # sampled points per image (so each minibatch queries $\approx BK$ points).

Then the **baseline local training cost** (Local-only) is

$$\mathrm{FLOPs}_i^{\mathrm{FedAvg}} \;\approx\; 3U_i\,\mathcal{F}_i. \tag{50}$$

Prototype methods (FedProto / FedTGP / FedSA / FedRE) These methods train only the client model f_i and add lightweight vector operations (prototype/anchor distances or similarities). Thus their dominant term remains $3U_i\mathcal{F}_i$.

FedProto, FedTGP and FedSA. Clients compute class prototypes and regularize towards global prototypes (distance in $\mathbb{R}^{d_p}$). The additional FLOPs are dominated by prototype distance computations:

$$\mathrm{FLOPs}_i^{\mathrm{FedProto}} \;\approx\; 3U_i\mathcal{F}_i \;+\; O(U_i \cdot B \cdot d_p) + O(C_i d_p), \tag{51}$$

where d_p is the representation dimension and C_i is the number of classes observed by client i.

FedRE. FedRE performs the standard local model update and then constructs an entangled representation–label pair. In our implementation, after the E-epoch local update, each client performs one additional forward pass over its local training set to extract representations and construct the class prototypes used by RAP. Let

$$U_i^{\mathrm{rep}} = \left\lceil \frac{n_i}{B} \right\rceil = \frac{U_i}{E}.$$

The dominant client-side computation is therefore

$$\mathrm{FLOPs}_i^{\mathrm{FedRE}} \approx 3U_i\mathcal{F}_i + U_i^{\mathrm{rep}}\mathcal{F}_i + O\big(n_i(d_r + C)\big). \tag{52}$$

The final representation-entanglement operations are lightweight compared with the additional backbone forward pass.

Proxy-model methods (FedKD / FedSCE / pFedMoE / FedGen / FedZGE) These methods typically incur larger local FLOPs because they either (i) train additional proxy models, or (ii) perform extra passes on public/synthetic data.

In **FedKD**, both the local model and the auxiliary model are trained on the client device, the dominant local FLOPs are

$$\mathrm{FLOPs}_i^{\mathrm{FedKD}} \;\approx\; 3U_i\big(\mathcal{F}_i + \mathcal{F}_{\mathrm{aux}}\big). \tag{53}$$

FedSCE trains both a local model and an auxiliary model, and adds periodic SVD-based subspace updates. We write this as an amortized term with update period ϱ:

$$\mathrm{FLOPs}_i^{\mathrm{FedSCE}} \;\approx\; 3U_i\big(\mathcal{F}_i + \mathcal{F}_{\mathrm{aux}}\big) \;+\; O\!\left(\frac{1}{\varrho}\, m_{\mathrm{svd}}\, r_{\mathrm{sub}}^2\right), \tag{54}$$

where m_{svd} is a layer-size scale and r_{sub} is the subspace rank.

pFedMoE. At every local step, pFedMoE jointly optimizes the heterogeneous local model, the shared proxy feature extractor, and the personalized gating network. Hence, its dominant client-side computation is

$$\mathrm{FLOPs}_i^{\mathrm{pFedMoE}} \approx 3U_i\big(\mathcal{F}_i + \mathcal{F}_{\mathrm{pex}} + \mathcal{F}_{\mathrm{gate}}\big) + O(U_i B d_p), \tag{55}$$

where the last term represents the lightweight sample-wise fusion of the two expert representations. With our CNN-5-style 512-dimensional proxy extractor and a two-layer gating network with hidden width $m = 128$, the resulting computation costs are 8.52, 141.05, and 404.25 $\times 10^4$ GFLOPs/round on CIFAR-10, CIFAR-100, and Tiny-ImageNet, respectively.

FedGen trains on an additional generated mini-batch per local step in the feature space. Let B_G be the generated mini-batch size, $\mathcal{F}_i^h$ the forward FLOPs of the head h_i alone on one generated mini-batch. With one real mini-batch and one generated mini-batch per step ($B_G \approx B$), the local cost is

$$\mathrm{FLOPs}_i^{\mathrm{FedGen}} \;\approx\; 3U_i\mathcal{F}_i + 3U_i\frac{B_G}{B}\mathcal{F}_i^h + U_i\,\mathcal{F}_G. \tag{56}$$

FedZGE queries client models on server-synthesized inputs. On the client, this query stage is *forward-only* (cost $\mathcal{F}_i$ per query minibatch), but zeroth-order estimation typically requires evaluating multiple perturbations. Let $|\hat{\mathcal{X}}|$ be the number of synthesized samples queried per round, and let

$$U_i^{\mathrm{qry}} \;=\; \left\lceil \frac{|\hat{\mathcal{X}}|}{B} \right\rceil \tag{57}$$

be the number of query minibatches. With N_{dir} random directions (plus one base evaluation), the query FLOPs scale as $(N_{\mathrm{dir}}+1)U_i^{\mathrm{qry}}\mathcal{F}_i$. Thus,

$$\mathrm{FLOPs}_i^{\mathrm{FedZGE}} \;\approx\; 3U_i\mathcal{F}_i \;+\; (N_{\mathrm{dir}}+1)\,U_i^{\mathrm{qry}}\,\mathcal{F}_i. \tag{58}$$

We set $N_{\mathrm{dir}} = 10$ by default in both our implementation and complexity estimation, following the common setting used in FedZGE.

FedCF keeps the dominant local backprop cost close to single-model training, and adds forward-only teacher computations on point-sampled queries. Specifically, for each minibatch we sample K points per image, yielding about BK point queries. Let $\mathcal{F}_{\mathrm{teach}}(BK)$ denote the forward FLOPs of the global-side components (e.g., global field/classifier and lightweight translators) evaluated on BK points; this term typically scales linearly with BK and is substantially smaller than a full backbone forward/backward. Additionally, local field fitting updates a small continuous-field module with forward FLOPs $\mathcal{F}_{\mathrm{lrcf}}$ for U_i^{lrcf} steps per round (with cached features), yielding

$$\mathrm{FLOPs}_i^{\mathrm{FedCF}} \;\approx\; 3U_i\mathcal{F}_i \;+\; U_i\,\mathcal{F}_{\mathrm{teach}}(BK) \;+\; 3U_i^{\mathrm{lrcf}}\,\mathcal{F}_{\mathrm{lrcf}}. \tag{59}$$

In typical settings, the global-side evaluation uses only shallow heads (MLPs/linear layers) and point queries (K is small and does not require dense-grid evaluation), hence the added term $U_i\mathcal{F}_{\mathrm{teach}}(BK)$ is usually a small fraction of the backbone training cost $3U_i\mathcal{F}_i$. Likewise, $\mathcal{F}_{\mathrm{lrcf}}$ involves only low-rank field evaluation (and no backbone), and the number of field steps U_i^{lrcf} is typically much smaller than the number of backbone training steps U_i; therefore $3U_i^{\mathrm{lrcf}}\mathcal{F}_{\mathrm{lrcf}} \ll 3U_i\mathcal{F}_i$. Crucially, unlike proxy-model / public-data methods, we do not introduce an additional trainable auxiliary model on-device, so we avoid the dominant "$+\,3U_i\mathcal{F}_{\mathrm{aux}}$" overhead.

From Table 2, FedKD and FedSCE introduce substantial additional computation because a complete homogeneous auxiliary model is trained together with the local model. pFedMoE reduces this cost by sharing only a homogeneous proxy feature extractor, but still jointly trains the proxy extractor, the local expert, and a sample-wise gating network. FedGen introduces additional synthetic-data processing, while FedZGE requires repeated black-box queries. In contrast, FedCF keeps the dominant cost as single-model training, and only adds (i) one-way teacher evaluation on sampled points and (ii) a lightweight field-fitting term. The concrete communication cost under different experimental settings is reported in the main experimental section.

Table 3: Theoretical per-round **server–client** communication cost.

Method	Per-round communication (dominant terms)	Overhead source
FedProto	$\sum_{i=1}^{N} d_p\,(C_i + C)$	Prototype upload/download
FedTGP	$\sum_{i=1}^{N} d_p\,(C_i + C)$	Prototype upload/download
FedSA	$\sum_{i=1}^{N} d_p\,C_i \;+\; N d_p C$	Anchor upload/download
FedRE	$N(d_r + C) + N\lvert\theta_{\text{head}}\rvert$	Entangled-pair upload + classifier broadcast
FedKD	$2N\,\lvert\theta_{\text{pro}}\rvert$	Proxy model broadcast
FedSCE	$2N\,\lvert\theta_{\text{pro}}\rvert$	Proxy model broadcast
pFedMoE	$2N\,\lvert\theta_{\text{pex}}\rvert$	Proxy extractor upload/download
FedGen	$2\sum_{i=1}^{N}\lvert\theta_i^h\rvert + N\lvert\theta_{\text{gen}}\rvert$	Generator + head broadcast
FedZGE	$N\lvert\hat{X}\rvert\Big((N_{\text{dir}}{+}1)d_x + (N_{\text{dir}}{+}2)C\Big)$	Sample-level black-box queries
FedCF	$\sum_{i=1}^{N}\lvert\theta_i^S\rvert \;+\; \sum_{i=1}^{N}\lvert\theta_i^G\rvert \;+\; N\lvert\theta^M\rvert \;+\; \frac{N}{R_{\text{tr}}}\,\lvert\theta_{\text{tr}}\rvert$	Continuous fields + translators

F.2 Communication cost

The communication cost here is computed as the total number of scalar values exchanged due to mutual communication between the server and all clients in one global round. Throughout this part we use the following notation:

- N: number of clients per round.
- C: total number of classes.
- C_i: number of classes that actually appear on client i.
- d_p: dimension of class prototypes (e.g., d_p=512 in our experiments).
- $|\theta_i^h|$: number of parameters in the local classifier on client i.
- $|\theta_{pro}|$: number of parameters in the proxy model.
- $|\theta_{gen}|$: number of parameters in the generator of FedGen.
- d_r: dimension of the FedRE entangled representation.
- $|\theta_{\text{head}}|$: number of parameters in the FedRE global classifier.
- $|\theta_{\text{pex}}|$: number of parameters in the pFedMoE shared proxy feature extractor.

Unless otherwise stated, we only count the dominant terms and ignore negligible constants (e.g., a few architecture labels). We follow the communication-cost computation protocol in FedTGP. For methods already covered in FedTGP, we omit further discussion. For the newly introduced methods, the communication cost is computed as follows:

FedSA. FedSA follows the prototype-based communication protocol, and its communication cost is similar to that of FedTGP.

FedRE. Each participating client uploads one d_r-dimensional entangled representation together with one C-dimensional entangled-label encoding. The server trains and broadcasts a global linear classifier whose parameter count is $|\theta_{\text{head}}| = d_r C + C$. Therefore,

$$\text{Comm}_{\text{FedRE}} = N(d_r + C) + N(d_r C + C). \tag{60}$$

FedSCE. FedSCE maintains a homogeneous auxiliary model with parameters θ_{pro} on the server, requiring bidirectional transmission between the clients and the server. Additionally, it computes two

scalar distances $F_{i,1}$ and $F_{i,2}$ to characterize its update bias in the feature and parameter spaces; their computational cost is negligible and thus omitted from the overhead analysis. Therefore, the overall communication cost is similar to that of FedKD.

pFedMoE. pFedMoE communicates only the shared homogeneous proxy feature extractor, whereas the heterogeneous local model, prediction head, and personalized gating network remain client-specific. Hence,

$$\mathrm{Comm}_{\mathrm{pFedMoE}} = 2N|\theta_{\mathrm{pex}}|. \tag{61}$$

Our proxy extractor consists of two 5×5 convolutional layers with 16 and 32 output channels, adaptive average pooling to 5×5, and two fully connected layers. Because the communicated proxy extractor does not contain a class-dependent prediction head, this communication cost is identical across CIFAR-10, CIFAR-100, and Tiny-ImageNet.

FedZGE. FedZGE performs data-free black-box queries on client models using server-synthesized inputs. In each round, the server sends a synthesized set $\hat{X}$ of size $|\hat{X}|$ to every participating client, where each input contains d_x scalars (e.g., $d_x = 3HW$ for an $H \times W$ RGB image). For zeroth-order gradient estimation, the server additionally sends N_{dir} perturbed versions of the same set, so the client is queried on $(N_{\mathrm{dir}}+1)$ input sets in total. Each client returns the predicted logits (dimension C per sample) on these queried inputs, and the server also broadcasts one set of ensembled logits on $\hat{X}$ for distillation. Therefore, the downlink cost per round is $N|\hat{X}|\big((N_{\mathrm{dir}}+1)d_x + C\big)$, and the uplink cost is $N|\hat{X}|(N_{\mathrm{dir}}+1)C$. Summing both directions yields

$$\mathrm{Comm}_{\mathrm{FedZGE}} = N|\hat{X}|\Big((N_{\mathrm{dir}}+1)d_x + (N_{\mathrm{dir}}+2)C\Big).$$

FedCF. In each communication round, client i uploads its local class-wise continuous field, and the server broadcasts a global continuous teacher, a global classifier, and architecture translators back to clients. If client i uploads the parameters of its local field S_i and the server broadcasts the parameters of the global continuous field, the global classifier, and the translators, the average per-round communication cost is

$$\mathrm{Comm}_{\mathrm{FedCF}} = \underbrace{\sum_{i=1}^{N} \big|\theta_i^S\big|}_{\text{local fields}} + \underbrace{\sum_{i=1}^{N} \big|\theta_i^G\big| + N\big|\theta^M\big|}_{\text{global field \& classifier}} + \underbrace{\frac{N}{R_{\mathrm{tr}}}\,|\theta_{\mathrm{tr}}|}_{\text{translators}},$$

where R_{tr} is the translator update period, and $|\theta_{\mathrm{tr}}|$ denotes the number of scalars of the translator module(s) broadcast per participating client (e.g., including the feature translator and the coordinate translator). Here θ_i^S denotes the parameters of the local low-rank field S_i on client i, including two 1D factor networks and its C_i class cores. The dominant term scales as

$$\big|\theta_i^S\big| \approx C_i\, r_u r_v d_i,$$

where C_i is the number of classes observed on client i, d_i is the feature dimension of client i. Similarly, client i only receives global cores for its own label set $\mathcal{C}_i$, so

$$\big|\theta_i^G\big| \approx C_i\, r_s r_t d,$$

where (r_s, r_t) are the ranks of the global field and d is the shared canonical dimension.

Table 3 highlights how different collaboration carriers lead to fundamentally different communication costs. Conventional prototype-based methods transmit class-wise vectors, while FedRE further entangles local class prototypes into a single representation–label pair and broadcasts a global classifier. These methods remain communication-efficient, but communicate highly compressed discrete knowledge. Proxy-model methods instead communicate homogeneous auxiliary components. FedKD and FedSCE exchange complete proxy models, whereas pFedMoE communicates only a shared proxy feature extractor. FedGen and FedZGE rely on generated samples or sample-level queries, with FedZGE incurring particularly high traffic due to repeated zeroth-order queries. FedCF instead communicates compact low-rank continuous fields, i.e., function parameters rather than raw samples or complete auxiliary models, providing a more expressive yet controllable collaboration carrier under severe cross-architecture heterogeneity.

G Additional Experimental Details

G.1 Experimental Setup

Datasets CIFAR-10 and CIFAR-100 consist of 32×32 RGB images with 10 and 100 classes. In our federated setting, each client further splits its local data into 70% training, 15% validation, and 15% held-out test sets; all reported accuracies are averaged over the client-side test sets. Tiny-ImageNet contains 100k training images and an official validation set. We use the standard 64×64 RGB version, redistribute all available images across 30 clients, and perform a 70/15/15 train/validation/test split locally.

Model Heterogeneity. Across all datasets, we simulate realistic cross-architecture deployments by assigning different backbones to different clients. On CIFAR-10, we use a LeNet-style CNN, AlexNet, and ResNet-10; on CIFAR-100, we use AlexNet, ResNet-10, and ViT; and on Tiny-ImageNet, we use MobileNetV3, ResNet-18, and ViT. The ViT uses 4×4 patches, 6 Transformer layers, a hidden dimension of 512, and 8 attention heads. In all cases, the 30 clients are partitioned across the three backbones in a fixed 3:4:3 ratio so that no single backbone dominates the collaboration.

Baselines. We compare FedCF with nine representative heterogeneous federated learning methods. All methods are evaluated under the same client partitions and heterogeneous backbone pools described above.

Prototype-based methods. FedProto and FedTGP maintain a class prototype for each category. Each client first computes class-wise mean feature vectors locally, uploads them to the server, and then regularizes its local features toward the aggregated global prototypes. Building on this, FedSA further introduces learnable semantic anchors and strengthens alignment via similarity-based regularization. FedRE constructs an entangled representation–label pair from local class prototypes to train a global classifier.

Proxy-model methods. We adopt five distillation-based baselines that rely on auxiliary models or synthetic data, and group them into two subclasses. (i) Proxy models trained on local data. FedKD trains, on each client, a homogeneous auxiliary model in addition to the local heterogeneous backbone, and performs distillation between the local backbone and the globally aggregated auxiliary model. FedSCE further imposes a subspace constraint on the auxiliary model and periodically updates this subspace to improve representation quality. In our implementation, FedKD and FedSCE use the same lightweight CNN proxy model with two convolutional layers followed by three fully connected layers. pFedMoE shares a homogeneous proxy feature extractor and combines it with the local feature extractor through a personalized gating network for sample-wise feature fusion. (ii) Proxy models not trained on local data. FedGen trains a conditional generator on the server based on client-side classifiers; the trained generator is broadcast to clients, which locally generate labeled pseudo-samples for distillation. FedZGE likewise maintains a generator and a global model on the server, and only sends randomly perturbed pseudo-samples to clients and collects their forward predictions; using zeroth-order gradient estimation, it updates the generator to produce high-quality distillation samples without real data.

G.2 Implementation Details

Federated protocol. We use $N = 30$ clients in all experiments. Unless otherwise specified, all N clients participate in every communication round (no client sampling), so no additional variability is introduced by client sampling. We run a fixed number of communication rounds per dataset (100 rounds for CIFAR-10/100 and a slightly larger budget for Tiny-ImageNet), chosen so that all methods have largely converged. All local backbones (including those used by baselines) are trained with Adam using an initial learning rate of 3×10^{-4} and weight decay 10^{-5}. Mini-batch size is set to $B = 64$ for CIFAR-10/100 and Tiny-ImageNet.

FedCF-specific hyperparameters. For continuous-field fitting, we cache intermediate feature maps on each client and train a small LRTFR-style field module with a separate Adam optimizer (learning rate 10^{-2}) for a few epochs per round. We set the field ranks (r_u, r_v) and (r_s, r_t) to small values (e.g., 4–8) depending on the backbone and dataset.

G.3 Experimental Analysis of Continuous Feature Fields

We provide additional details for the diagnostic experiments in Figures 3,4, and 5, which jointly examine the geometry, distributional fit, and cross-architecture interpretability of the proposed continuous feature fields. Throughout this section we follow the notation in Sec. 2.2. For client i, let $\Phi_i(x) \in \mathbb{R}^{H_i \times W_i \times d_i}$ denote the backbone feature grid at the target layer for input x, with spatial indices (h, w) mapped to normalized coordinates $(u, v) \in [0, 1]^2$. Local continuous fields $S_{i,c}(u, v)$ are learned by minimizing the regression loss in Eq. (3) on the sampled point set

$$D_i^s = \Big\{((u, v), z, y) : (x, y) \in D_i,\ (h, w) \in \Omega(x),\ z = \Phi_i(x)_{h,w,:}\Big\},$$

and are aligned to a canonical space via the feature translator A_i and coordinate translator ψ_i, yielding global fields $S_c(s, t)$ defined on $(s, t) = \psi_i(u, v)$.

3D surfaces of class-direction projections (Figure 3). Figure 3 visualizes how the continuous fields capture class-conditional spatial structure along discriminative directions. We fix a client i and a class c, and select two images x_1, x_2 from that client belonging to class c. Let $w_{i,c} \in \mathbb{R}^{d_i}$ denote the last-layer weight of the local classifier for class c on client i. For a given sample x and a spatial location corresponding to normalized coordinate (u, v) (with underlying index (h, w) on the grid), we define the real class-direction response as

$$f_{i,c}^{\text{real}}(u, v; x) \;=\; w_{i,c}^{\top} \Phi_i(x)_{h,w,:}, \tag{62}$$

which measures the projection of the true backbone feature at (u, v) onto the local class direction $w_{i,c}$. Replacing the discrete feature with the local continuous field output gives the local-field class-direction response

$$f_{i,c}^{\text{loc-field}}(u, v) \;=\; w_{i,c}^{\top} S_{i,c}(u, v), \tag{63}$$

which reflects how the learned continuous field $S_{i,c}$ responds along the same discriminative direction.

To examine how the canonicalized local field is scored along the global classifier's class-c discriminative direction, we first map the local field into the canonical feature space via $z_{i,c}^{\text{canon}}(u, v) = A_i(S_{i,c}(u, v))$, and let g_{glob} denote the global classifier with penultimate feature mapping $\varphi_{\text{glob}}(\cdot)$ and final weight w_c^{glob} for class c. The resulting local-field under global-classifier response is defined as

$$f_c^{\text{glob-field}}(u, v) \;=\; \big(w_c^{\text{glob}}\big)^{\top} \varphi_{\text{glob}}\big(z_{i,c}^{\text{canon}}(u, v)\big), \tag{64}$$

which measures the class-c discriminative response of the canonicalized continuous field under the global classifier. In Figure 3, we take a ResNet client on the CIFAR-100 task as an example and plot, on a dense coordinate grid, the surfaces of the scalar fields $f_{i,c}^{\text{real}}(u, v; x_1)$, $f_{i,c}^{\text{real}}(u, v; x_2)$, $f_{i,c}^{\text{loc-field}}(u, v)$, and $f_c^{\text{glob-field}}(u, v)$, where $(u, v) \in [0, 1]^2$. The real-sample surfaces $f_{i,c}^{\text{real}}(u, v; x_j)$ exhibit pronounced sample-level noise and substantial shape variation across different samples, indicating that a single global prototype cannot faithfully capture such fine-grained spatial geometry. In contrast, the responses induced by the local continuous field are much smoother, preserve stable discriminative ridge structures, and display highly similar shapes under both the local and global classifiers' class directions. This suggests that the learned continuous field captures stable class-conditional spatial structure beyond individual sample variations.

Reconstruction error of continuous fields vs. prototypes (Figure 4). Figure 4 quantitatively evaluates whether continuous fields provide a more faithful approximation to the sampled feature distribution than discrete prototypes, both in the local feature spaces and in the canonical space.

In the local space of client i, we define the mean-squared reconstruction error of the continuous field as

$$\text{MSE}_{\text{field}}^{\text{loc}}(i) \;=\; \frac{1}{|D_i^s|} \sum_{((u,v),z,y) \in D_i^s} \big\| S_{i,y}(u, v) - z \big\|_2^2, \tag{65}$$

where $z = \Phi_i(x)_{h,w,:}$ is the true backbone feature at the sampled location and $S_{i,y}(u, v)$ is the corresponding field prediction. Let

$$\mu_{i,c}^{\text{loc}} \;=\; \frac{1}{|D_{i,c}^s|} \sum_{((u,v),z,y) \in D_i^s :\, y=c} z \tag{66}$$

denote the standard class-mean prototype of client i in its local feature space, where $D_{i,c}^s$ is the subset of D_i^s with label c. The reconstruction error of this prototype is

$$\mathrm{MSE}_{\mathrm{proto}}^{\mathrm{loc}}(i) \;=\; \frac{1}{|D_i^s|} \sum_{((u,v),z,y)\in D_i^s} \big\|\mu_{i,y}^{\mathrm{loc}} - z\big\|_2^2. \tag{67}$$

We then repeat this comparison in the canonical space. For each sample $((u,v),z,y) \in D_i^s$, we construct the canonical feature

$$z^{\mathrm{canon}} = A_i(z), \quad (s,t) = \psi_i(u,v).$$

Let μ_c^{glob} denote the global class-mean prototype in the canonical space (aggregated across all clients), and $S_c(s,t)$ be the global continuous field for class c. The canonical-space reconstruction errors of the continuous field and the prototype for client i are defined as

$$\mathrm{MSE}_{\mathrm{field}}^{\mathrm{glob}}(i) = \frac{1}{|D_i^s|} \sum_{((u,v),z,y)\in D_i^s} \big\|S_y\big(\psi_i(u,v)\big) - A_i(z)\big\|_2^2, \tag{68}$$

$$\mathrm{MSE}_{\mathrm{proto}}^{\mathrm{glob}}(i) = \frac{1}{|D_i^s|} \sum_{((u,v),z,y)\in D_i^s} \big\|\mu_y^{\mathrm{glob}} - A_i(z)\big\|_2^2. \tag{69}$$

Figure 4 illustrates that, compared to constant-vector prototypes, function-valued fields can fit the class-conditional feature distribution much more tightly, especially when reconciling disparate feature modes produced by multiple models under severe heterogeneity.

Proto-CAM vs. Continuous field-CAM (Figure 5). Figure 5 examines cross-architecture interpretability by comparing Grad-CAM visualizations derived from prototypes and from the global continuous fields. Both visualizations use standard Grad-CAM, but differ in the scalar score that is explained.

For a given client i, input x, and its predicted or ground-truth class y, we first compute the global-average pooled feature at the target layer, map it into the canonical space, and measure its cosine similarity with the global prototype:

$$s_{\mathrm{proto}}(x) \;=\; \cos\Big(A_i(\mathrm{GAP}\big(\Phi_i(x)\big)),\, \mu_y^{\mathrm{glob}}\Big), \tag{70}$$

where $\mathrm{GAP}(\cdot)$ is global average pooling over (h,w). This scalar $s_{\mathrm{proto}}(x)$ is used as the *Proto-CAM* score. We backpropagate $\partial s_{\mathrm{proto}}/\partial \Phi_i(x)$ and apply Grad-CAM to obtain a spatial heatmap that highlights locations whose features are most aligned with the global prototype.

In contrast, the Field-CAM score aggregates the cosine similarity between the canonicalized local feature at each position and the global continuous field at the corresponding canonical coordinate. For each sampled location $(h,w) \in \Omega(x)$ with normalized coordinate (u,v) and canonical coordinate $(s,t) = \psi_i(u,v)$, we form the canonical feature $A_i(\Phi_i(x)_{h,w,:})$ and define

$$s_{\mathrm{field}}(x) \;=\; \frac{1}{|\Omega(x)|} \sum_{(h,w)\in\Omega(x)} \cos\Big(A_i\left(\Phi_i(x)_{h,w,:}\right), S_y\big(\psi_i(u,v)\big)\Big). \tag{71}$$

We then backpropagate $\partial s_{\mathrm{field}}/\partial \Phi_i(x)$ and apply the same Grad-CAM procedure to produce the Field-CAM heatmap. Figure 5 compares Proto-CAM and Field-CAM on CIFAR-100 across three heterogeneous backbones (AlexNet, ResNet, ViT), reflecting semantic misalignment of constant-vector prototypes under heterogeneous geometry. By contrast, Field-CAM consistently focuses on similar semantic parts across architectures, indicating that the global continuous fields S_c indeed encode a backbone-agnostic spatial semantic template. We report additional visualizations for more clients and classes in Figure 7, which show similar trends.

G.4 Additional Results

Additional results under $\mathrm{Dir}_{0.3}$. Table 4 summarizes results under $\mathrm{Dir}_{0.3}$ across datasets. Our method consistently outperforms prototype methods, and remains competitive with proxy-model baselines while achieving the best Tiny-ImageNet accuracy in this setting. These additional results further indicate that low-rank continuous class feature fields preserve richer intra-class structure than discrete prototypes, thereby enabling more effective heterogeneous collaboration under realistic non-IID data.

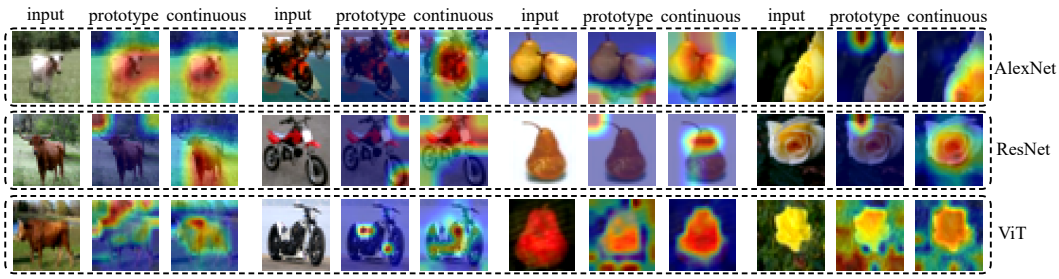


Figure 7: Prototype- vs. continuous-based visualizations on CIFAR-100. Both heatmaps are generated by Grad-CAM.

Table 4: Classification accuracies (%) across different datasets.

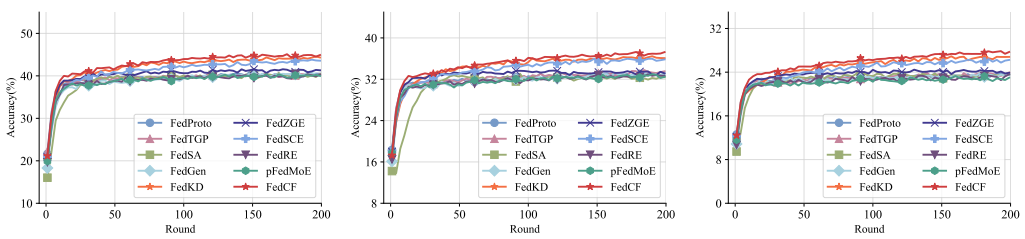
Method Group	Datasets	*CIFAR-10*	*CIFAR-100*	*Tiny-ImageNet*
	Settings	$Dir_{0.3}$	$Dir_{0.3}$	$Dir_{0.3}$
Proxy-model methods	FedKD	77.80	44.25	26.93
	FedGen	74.21	37.85	23.34
	FedSCE	77.07	**44.53**	26.57
	FedZGE	71.02	38.32	24.42
	pFedMoE	73.84	38.00	23.13
Prototype methods	FedProto	73.71	38.54	23.42
	FedTGP	74.53	39.34	23.82
	FedSA	73.98	38.30	23.12
	FedRE	74.38	38.44	23.45
Ours	FedCF	**77.93**	43.71	**27.81**

Convergence on Tiny-ImageNet. We report the accuracy convergence on Tiny-ImageNet. Figure 8 shows that our method converges faster across all settings. This advantage emerges early and remains stable, suggesting that alignment to continuous feature functions provides richer and more robust cross-architecture supervision than aligning single-point prototypes. When model heterogeneity is large and semantics are hard to reconcile, prototype-aggregation methods can quickly saturate.

Architecture-wise gains and cost trade-off. Figure 9a shows that our method improves performance for every backbone group on Tiny-ImageNet. Compared with the strongest prototype baseline, it achieves gains of +3.3% on MobileNet, +3.1% on ResNet, and +1.8% on ViT. This supports our key claim that aligning and aggregating class knowledge in a canonical continuous function space yields more universal semantics for cross-architecture learning, rather than favoring any particular model family. Figure 9b further highlights the efficiency of low-rank continuous feature functions as the carrier. Our method incurs substantially lower computation and communication than most proxy-model baselines. Together, these results verify that continuous feature fields can improve the compute–communication trade-off without relying on proxy data.

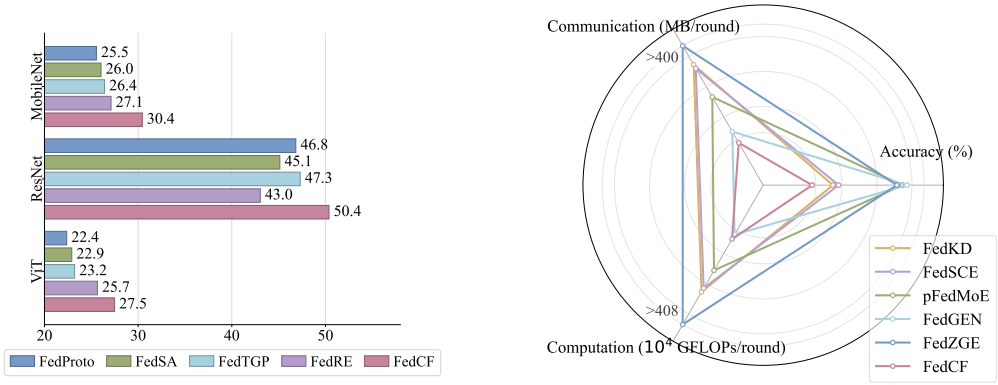
Multi-point distillation. Eq. (8) distills predictions queried at multiple coordinates into one image-level prediction. To examine whether these coordinate-level teachers introduce conflicting supervision, we compare multi-point KD with three variants on CIFAR-100. *Multi-point* is our default strategy, which averages the KL divergence over all queried coordinates. *Mean-teacher* first averages the teacher distributions across coordinates and then performs distillation. *Single-point* uses only one queried coordinate per image. *GT-filtered* removes coordinates that assign weak probability to the ground-truth class before aggregation.

To quantify the consistency of coordinate-level teachers, we report the Jensen–Shannon (JS) divergence among their predicted distributions, where a smaller value indicates stronger agreement. We also report the average probability assigned by the aggregated teacher to the ground-truth class (GT probability), where a larger value indicates more reliable supervision.



(a) Tiny-ImageNet(*Pat*). (b) Tiny-ImageNet($Dir_{0.1}$). (c) Tiny-ImageNet($Dir_{0.3}$).

Figure 8: Convergence curves of methods on Tiny-ImageNet. The accuracy at each round is the average test accuracy over all clients.



(a) Per-architecture accuracy. (b) Computation and communication.

Figure 9: Performance and efficiency on Tiny-ImageNet. (a) Per-architecture average accuracy. (b) Computation and communication costs; outward indicates worse performance.

Table 5: Comparison of multi-point distillation strategies on CIFAR-100.

Strategy	Acc. (%) ↑	JS divergence ↓	GT probability ↑
Multi-point (FedCF)	59.30	0.034	0.812
Mean-teacher	59.25	0.032	0.815
Single-point	57.41	0.047	0.757
GT-filtered	59.52	0.028	0.853

As shown in Table 5, multi-point and mean-teacher distillation achieve nearly identical accuracy and similarly low JS divergence, indicating that teacher predictions from different coordinates are largely consistent rather than conflicting. In contrast, single-point distillation yields lower accuracy, larger teacher disagreement, and weaker support for the ground-truth class. GT filtering further reduces disagreement but yields only a marginal +0.22 pp gain. These results suggest that the main benefit of multi-point distillation comes from aggregating multiple pieces of local evidence and reducing single-coordinate noise; we therefore retain unfiltered multi-point distillation without introducing an additional ground-truth-dependent filtering rule.

Table 6: Hyperparameter sensitivity on CIFAR-100.

Hyperparameter	Setting	Acc. (%) ↑	Align MSE ↓
Field rank r	4	57.7	0.0045
	6	59.0	0.0031
	12	59.4	0.0027
Sampling budget K	8	58.6	0.0037
	12	59.3	0.0030
	16	59.5	0.0028
Translator update period R_{tr}	5	59.0	0.0033
	10	59.3	0.0030
	15	58.4	0.0041

Hyperparameter sensitivity. We further examine the sensitivity of FedCF to the local field rank r, sampling budget K, and translator update period R_{tr} on CIFAR-100 under $\mathrm{Dir}_{0.1}$. Besides average test accuracy, we report Align MSE to measure the discrepancy between the global field and translator-aligned local features in the canonical space. As shown in Table 6, increasing r or K from small values improves both accuracy and alignment, while the gains gradually saturate. In contrast, overly sparse translator updates make the alignment stale and degrade performance. Overall, moderate hyperparameter values already provide a favorable accuracy–stability trade-off.