GEOMETRY-AWARE SLOW-FAST LEARNING FOR DECENTRALIZED FEDERATED LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

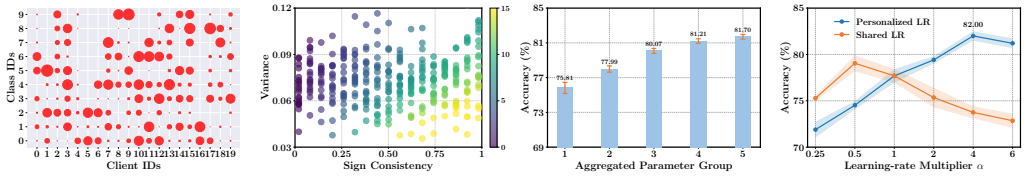
Data heterogeneity in decentralized federated learning is inherently non-uniform: different model parameters exhibit different levels of cross-client stability and adaptation speed, while different clients vary in their compatibility for knowledge exchange. However, most existing approaches still rely on largely uniform optimization and consensus rules, overlooking this fine-grained structure of heterogeneous collaboration. We therefore argue that effective decentralized learning should jointly account for two complementary dimensions: *parameter-wise adaptation timescales* and *client-wise collaboration geometry*. To this end, we propose **Geo**metry-aware **S**low-**F**ast Learning (**GeoSF**), a decentralized learning framework that explicitly models both forms of heterogeneity. At the parameter level, **GeoSF** organizes model parameters into two dynamically evolving groups: *slow* parameters that remain stable across clients and preserve transferable shared knowledge, and *fast* parameters that respond more strongly to local data and support client-specific adaptation. **GeoSF** then matches their distinct roles with different optimization and communication timescales, using smaller learning rates and more frequent synchronization for slow parameters, while allowing fast parameters to evolve more rapidly and synchronize less frequently. At the client level, **GeoSF** adapts peer-to-peer consensus according to the geometric compatibility of neighboring models, promoting beneficial information exchange while suppressing interference from heterogeneous peers. Together, these mechanisms establish a unified slow-fast learning paradigm that jointly coordinates parameter dynamics and client interactions under decentralized heterogeneity. Extensive experiments across feature-skew and label-skew settings, model architectures, communication topologies, and client scales demonstrate that **GeoSF** consistently outperforms representative federated and decentralized baselines.

1 INTRODUCTION

Federated Learning (FL) enables multiple clients to collaboratively train models without directly sharing private data (Li et al., 2021b; 2020b; Yang et al., 2019). However, Traditional Federated Learning (TFL) typically learns a single global model, whose effectiveness can deteriorate substantially when client data follow heterogeneous distributions (Karimireddy et al., 2020; McMahan et al., 2017; Huang et al., 2023). Personalized Federated Learning (PFL) alleviates this limitation by allowing each client to maintain a model adapted to its local data (Sui et al., 2022; Oh et al., 2021; Liang et al., 2020). Despite their effectiveness, most TFL and PFL approaches still rely on a central server for coordination, introducing a communication bottleneck and a single point of failure (Dai et al., 2022; Shi et al., 2023; Li et al., 2025b).

Decentralized Federated Learning (DFL) replaces the central coordinator with direct peer-to-peer communication, improving deployment flexibility and avoiding a single point of failure (Yuan et al., 2024; Li et al., 2026a; Beltrán et al., 2023). Nevertheless, decentralization does not eliminate the fundamental challenge posed by data heterogeneity (Wang et al., 2022; Nedic & Ozdaglar, 2009; Jeong & Kountouris, 2023). Under non-independent and identically distributed (non-IID) data, local optimization drives different clients toward distinct solutions, whereas decentralized consensus encourages their models to move toward those of neighboring clients (Chung et al., 2026; Hsieh et al., 2020). This creates an inherent tension between *local adaptation* and *cross-client consensus*: excessive synchronization may inject conflicting knowledge, while insufficient synchronization limits

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) Data Heterogeneity (b) Parameter Consistency (c) Shared Parameter Selection (d) Different Learning Rates

Figure 1: **Empirical analysis** for parameter properties under data heterogeneity. **(a)** Heterogeneous data distributions across clients. **(b)** Parameters exhibit different levels of cross-client sign consistency and variance, reflecting their suitability for knowledge sharing. **(c)** Aggregating parameters from different consistency-score ranges leads to distinct performance, with highly consistent parameters yielding the best results. **(d)** Parameters respond differently to learning-rate scaling, performance improves with a smaller learning rate for shared parameters ($\alpha = \eta_s/\eta_p$) and a larger one for personalized parameters ($\alpha = \eta_p/\eta_s$).

knowledge propagation and collaboration. Importantly, this tension is not uniform across either model parameters or clients. Different parameters may exhibit different levels of cross-client stability and adaptation speed, while different clients may vary considerably in their compatibility for knowledge exchange. Effective decentralized learning should therefore account for not only *what* knowledge should be shared, but also *how fast* it should evolve and *with whom* it should be exchanged.

Existing personalized DFL methods partially address this challenge by introducing parameter partitioning or selective collaboration (Liu et al., 2024; Zhang et al., 2025; Shi et al., 2026). For example, DFedPGP (Liu et al., 2024) and DFedSMC (Shi et al., 2026) decompose model parameters into shared and client-specific components, while DisPFL (Dai et al., 2022) achieves personalization through client-specific masks. However, these approaches typically rely on predefined parameter partitions or largely uniform optimization and consensus rules, overlooking two fine-grained structures induced by data heterogeneity. First, parameters naturally evolve on different *adaptation timescales*: some parameters remain relatively stable across clients and are suitable for preserving transferable shared knowledge, whereas others respond more rapidly to local distributions and favor client-specific adaptation. Second, clients exhibit different *collaboration geometry*: neighboring models may occupy distinct regions of parameter space, making them unequally compatible for knowledge exchange. This raises two fundamental questions: **1)** How can we identify parameters with different adaptation timescales and coordinate their optimization and synchronization accordingly? **2)** How can we exploit client compatibility to determine from whom knowledge should be acquired during consensus?

We empirically examine these fine-grained effects of heterogeneity. As shown in Fig. 1b, model parameters exhibit markedly different cross-client behaviors in terms of sign consistency and variance: some remain stable in direction and magnitude, while others are more sensitive to client-specific data distributions. This distinction is consequential for decentralized collaboration. Fig. 1c shows that parameters with higher cross-client consistency are substantially more amenable to aggregation, indicating that parameter shareability is inherently non-uniform. Moreover, Fig. 1d reveals that stable shared parameters and client-specific parameters favor different optimization speeds, suggesting that they naturally evolve on distinct adaptation timescales. These observations motivate a *slow-fast* view of model dynamics: stable and transferable parameters should evolve slowly to preserve shared knowledge, whereas sensitive parameters should adapt rapidly to local distributions. Heterogeneity also manifests at the client level. Different local objectives drive neighboring models toward distinct regions of the parameter space, making knowledge exchange between geometrically incompatible clients potentially detrimental. These observations reveal two dimensions of decentralized heterogeneity: *parameter-wise adaptation timescales* and *client-wise collaboration geometry*.

To address these challenges, we propose **Geo**metry-aware **S**low-**F**ast Learning (**`GeoSF`**), a decentralized federated learning framework that jointly models *parameter-wise adaptation timescales* and *client-wise collaboration geometry*. At the parameter level, **`GeoSF`** dynamically groups parameters by cross-client stability: *slow* parameters, with high sign consistency and low variance, preserve shared knowledge, whereas *fast* parameters support client-specific adaptation. We then match these distinct roles with different optimization and communication timescales, assigning smaller learning rates and more frequent synchronization to slow parameters, while allowing fast parameters to evolve more rapidly and synchronize less frequently. At the client level, **`GeoSF`** constructs geometry-aware consensus weights according to the compatibility of neighboring model states, promoting knowledge exchange among compatible clients while suppressing potentially conflicting interactions. In this way,

GeoSF provides a unified mechanism for coordinating *what* knowledge should be shared, *how fast* it should evolve, and *with whom* it should be exchanged. Extensive experiments across heterogeneous data distributions, model architectures, communication topologies, and client scales demonstrate the effectiveness and robustness of **GeoSF**. Our main contributions are summarized as follows:

- We reveal that data heterogeneity in DFL manifests not only across clients but also across model parameters, leading to substantial differences in cross-client stability, shareability, and adaptation speed. Motivated by this observation, we introduce a *slow-fast learning paradigm* that dynamically separates stable shared parameters from rapidly adapting client-specific ones and coordinates them through distinct optimization and synchronization timescales.
- We introduce a *geometry-aware decentralized consensus mechanism* that adapts collaboration strength according to the compatibility of neighboring model states, enabling beneficial knowledge propagation while reducing interference from heterogeneous peers.
- We conduct extensive experiments under both feature-skew and label-skew settings, covering multiple model architectures, communication topologies, and client scales. The results consistently demonstrate the effectiveness, robustness, and scalability of **GeoSF**.

2 Related Work

Due to space constraints, we provide a brief overview here and defer further details to the appendix.

Personalized federated learning. Personalized federated learning (PFL) mitigates the effects of data heterogeneity by tailoring models to individual clients (Zhang et al., 2023a; Yang et al., 2024). Representative methods such as FedPer (Arivazhagan et al., 2019), FedRep (Collins et al., 2021), and FedAS (Yang et al., 2024) share the model backbone while retaining local prediction heads, whereas Ditto (Li et al., 2021c) and pFedMe (T Dinh et al., 2020) promote global-to-local knowledge transfer through regularization. Although these methods improve local adaptability, their reliance on a central server for aggregation can create a communication bottleneck and a potential single point of failure.

Decentralized federated learning. Decentralized federated learning (DFL) enables more flexible deployment through peer-to-peer communication (Li et al., 2021a; Kang et al., 2022; Sadiev et al., 2022; Li et al., 2022). Existing DFL methods either learn a shared model or personalize models for individual clients. D-PSGD (Lian et al., 2017), DFedAvgM (Sun et al., 2022), and DFedADMM (Li et al., 2025a) combine local optimization with client consensus to learn a shared model, which may struggle to meet individual clients' needs. DFedPGP (Liu et al., 2024) and DFedSMC (Shi et al., 2026) instead separate local and shared parameters, applying consensus only to the latter. However, their predefined divisions do not capture differences among parameters or how these differences change during training. In addition, uniform update and aggregation rules further limit performance.

3 Methodology

We consider a decentralized federated system with M clients, where client m holds private data $\mathcal{D}_m$ and model parameters θ_m. Let $\Theta = (\theta_1, \ldots, \theta_M)$ collect all local parameters. The objective is:

$$\min_{\Theta} F(\Theta) \triangleq \frac{1}{M} \sum_{m=1}^{M} F_m(\theta_m), \tag{1}$$

where $F_m(\theta_m) = \mathbb{E}_{(x,y)\sim\mathcal{D}_m}[\ell(\theta_m; (x, y))]$, denotes the client objective function. Without a central server, each client exchanges parameters only with its neighbors and forms a consensus estimate as $\theta_m = \sum_{n\in\mathcal{N}_m} W_{m,n}\theta_n$, where W^t encodes the communication topology and consensus weights (Sun et al., 2022; Zhu et al., 2022). Under data heterogeneity, local models may converge toward different solutions, leading to parameter conflicts during the consensus stage.

3.1 Slow-Fast Decomposition

The core challenge under data heterogeneity is not merely deciding whether local models should be shared or personalized, but identifying *which* parameters are reliable for cross-client sharing. As illustrated in Fig. 1b, heterogeneity affects parameters unevenly: some remain consistent across clients

and capture transferable knowledge, whereas others diverge due to client-specific data distributions. A uniform update and consensus strategy ignores this distinction, potentially disrupting shared knowledge and limiting local adaptation. Therefore, we decompose each local model into shared (*slow*) parameters that preserve stable cross-client knowledge and personalized (*fast*) parameters that support local adaptation. Specifically, the local model of client m at round t is defined as follows:

$$\theta_m^t = [\theta_m^{s,t}, \theta_m^{f,t}], \quad \theta_m^{s,t} = m_m^s \odot \theta_m^t, \quad \theta_m^{f,t} = m_m^f \odot \theta_m^t, \tag{2}$$

where $m_m^s, m_m^f \in \{0,1\}^d$ are complementary masks satisfying $m_m^s + m_m^f = \mathbf{1}$, and the superscripts s and f denote the *slow* and *fast* parameters, respectively.

Under data heterogeneity, shared and personalized parameters play distinct roles. The key challenge is identifying which parameters are sufficiently stable to be shared. To this end, we characterize the stability of each parameter from two perspectives: cross-client *sign consistency* and *variance*.

❶ ***Cross-client sign consistency.*** Cross-client sign consistency measures whether a parameter exhibits a consistent orientation across clients. Specifically, for the j-th parameter, we define:

$$a_{m,j}^t = \frac{1}{|\mathcal{N}_m|} \sum_{n \in \mathcal{N}_m} \operatorname{sign}(\theta_{n,j}^t) \in [-1, 1], \tag{3}$$

where $\operatorname{sign}(\cdot)$ denotes the sign function. The absolute value of $a_{m,j}^t$ measures cross-client agreement on the sign of the j-th parameter. A larger value indicates stronger agreement, suggesting that this coordinate is less affected by conflicting local objectives and is more suitable for consensus. A value near zero indicates sign disagreement, suggesting that heterogeneous local objectives affect this parameter differently. Enforcing consensus on such parameters may cause conflicts across clients. Thus, they are better suited for personalized adaptation.

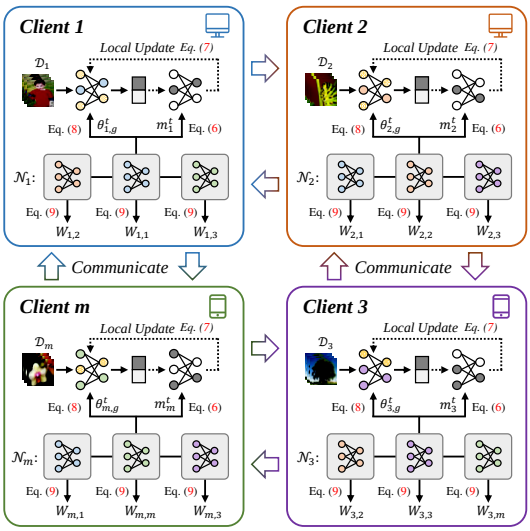


Figure 2: **Schematic** diagram of **GeoSF**. Each client first computes the consensus weights according to Eq. (9), then selects the slow and fast parameters and performs personalized aggregation. It subsequently updates the parameters on two timescales according to Eq. (7) and transmits its model θ_m^t.

❷ ***Cross-client variance.*** Although sign consistency captures whether clients agree on the *direction* of a parameter coordinate, it does not fully characterize cross-client stability. Parameter values may share the same sign yet differ substantially in *magnitude*, making the consensus result overly sensitive to client-specific fluctuations. We therefore complement sign consistency with cross-client variance, which measures the numerical dispersion of each parameter:

$$v_{m,j}^t = \frac{1}{|\mathcal{N}_m|} \sum_{n \in \mathcal{N}_m} \left(\theta_{n,j}^t - \bar{\theta}_j^t\right)^2, \quad \bar{\theta}_j^t = \frac{1}{|\mathcal{N}_m|} \sum_{n \in \mathcal{N}_m} \theta_{n,j}^t. \tag{4}$$

A smaller $v_{m,j}^t$ indicates that the j-th parameter remains numerically stable across clients. Conversely, a larger variance suggests that the parameter is strongly affected by local data distributions, and aggregating it may dilute client-specific information and degrade personalization performance.

In summary, sign consistency and variance capture two complementary properties of reliable shared parameters: sign consistency measures *directional agreement*, whereas variance quantifies *magnitude dispersion*. Based on these properties, we define a parameter stability score:

$$s_{m,j}^t = \left|\hat{a}_{m,j}^t\right| / \left(\sigma_{m,j}^t + \varepsilon\right), \tag{5}$$

where $\sigma_{m,j}^t$ denotes the standard deviation associated with $v_{m,j}^t$, and $\varepsilon > 0$ ensures numerical stability. A larger $s_{m,j}^t$ indicates stronger sign consistency and lower variance across clients, making the corresponding parameter more suitable for aggregation. Based on Eq. (5), we select the top ρ fraction of parameters as slow parameters and classify the remaining parameters as fast parameters:

$$m_{m,j}^{s,t} = \mathbb{I}\left[j \in \texttt{TopK}\left(\{s_{m,j}^t\}_{j=1}^d, \lceil \rho d \rceil\right)\right], \quad m_{m,j}^{f,t} = 1 - m_{m,j}^{s,t}, \tag{6}$$

where $\rho \in (0, 1]$ denotes the proportion of slow parameters. In this way, parameters encoding stable cross-client knowledge evolve and are aggregated on a slow timescale, whereas those dominated by heterogeneous variations evolve rapidly to preserve local adaptability.

3.2 Two-Timescale Local Updates

The aforementioned slow–fast decomposition identifies *which* parameters should carry shared knowledge and which should support local adaptation. However, distinguishing their roles during consensus is not sufficient, as these parameter groups exhibit different sensitivities to evolution rates during local optimization, as illustrated in Fig. 1d. Specifically, slow parameters are intended to preserve stable knowledge across clients and should be updated with a smaller learning rate. In contrast, fast parameters need to adapt quickly to client-specific data distributions and should be updated with a larger learning rate. This two-timescale design prevents stable shared knowledge from being rapidly overwritten by heterogeneous local gradients, while preserving sufficient plasticity for local adaptation. Accordingly, we update them at different evolution rates, with $\eta_s \ll \eta_f$, as defined below:

$$\theta_m^{s,t} \leftarrow \theta_m^{s,t} - \eta_s[m_m^s \odot g_m^t + \mu(\theta_m^{s,t} - \theta_{m,g}^{s,t})], \quad \theta_m^{f,t} \leftarrow \theta_m^{f,t} - \eta_f(m_m^f \odot g_m^t), \tag{7}$$

where $\odot$ denotes the Hadamard product. The shrinkage term $\mu(\theta_m^s - \theta_{m,g}^s)$ further constrains the slow parameters around the shared reference $\theta_{m,g}^s$, reducing client drift and improving cross-client consistency. Thus, the two-timescale update explicitly balances the stability of shared knowledge against the plasticity required for local adaptation.

3.3 Slow-Fast Consensus

Since slow parameters encode stable global knowledge, they should remain consistent across clients throughout training. Performing consensus on these parameters in every communication round limits client drift and enables generalizable information to spread continuously across the federated system. In contrast, fast parameters capture rapidly changing, client-specific knowledge and require greater freedom to adapt to local data. However, keeping them entirely local may lead to overfitting. Synchronizing them too frequently creates the opposite problem, as repeated averaging may remove client-specific information and weaken local adaptation. Periodic synchronization balances these objectives by allowing each client's fast parameters to adapt to its local data distribution while periodically introducing cross-client knowledge to reduce local overfitting. Accordingly, we synchronize slow parameters in every communication round and fast parameters only at fixed intervals:

Algorithm 1 Learning Process of **GeoSF**

Input: Learning rates η_s and η_f, regularization coefficient μ, communication interval s, slow parameter ratio ρ, and temperature coefficient τ.
Output: Personalized models $[\theta_1^T, \ldots, \theta_M^T]$.

for each round $t = 1, \ldots, T$ **do**
  for each client m in parallel **do**
    Obtain m_m^t and $W_{m,n}^t$ by Eqs. (6) and (9).
    if $\mathrm{mod}(t, s) = 0$ **then**
      $\theta_{m,g}^{f,t} \leftarrow \sum_{n \in \mathcal{N}_m} W_{m,n}^t \theta_n^{f,t-1}$.
    end if
    $\theta_{m,g}^{s,t} \leftarrow \sum_{n \in \mathcal{N}_m} W_{m,n}^t \theta_n^{s,t-1}$.
    for each local step $k = 1, \ldots, K$ **do**
      $g_m^{s,t} \leftarrow m_m^s \odot g_m^t$, $g_m^{f,t} \leftarrow m_m^f \odot g_m^t$.
      $\theta_m^{s,t} \leftarrow \theta_m^{s,t} - \eta_s[g_m^{s,t} + \mu(\theta_m^{s,t} - \theta_{m,g}^s)]$.
      $\theta_m^{f,t} \leftarrow \theta_m^{f,t} - \eta_f g_m^{f,t}$.
    end for
  end for
end for

$$\theta_{m,g}^{s,t} \leftarrow \sum_{n \in \mathcal{N}_m} W_{m,n}^t \, \theta_n^{s,t-1}, \ \forall t, \quad \theta_{m,g}^{f,t} \leftarrow \sum_{n \in \mathcal{N}_m} W_{m,n}^t \, \theta_n^{f,t-1}, \ \text{if } \mathrm{mod}(t, s) = 0, \tag{8}$$

where s represents the communication interval of the fast parameters, $\mathcal{N}_m$ denotes the set of neighbors of client m. This asymmetric synchronization maintains global consistency through the slow parameters while allowing the fast parameters to maintain sufficient local adaptability.

3.4 Geometry-Aware Weighted Consensus

The slow–fast mechanism determines *what* parameters should be shared and *when* consensus should be performed, but does not determine *from whom* that knowledge should be acquired. Under severe data heterogeneity, neighboring clients may converge toward substantially different local solutions, making uniform consensus potentially detrimental even when the shared parameters are properly selected. Therefore, we leverage model geometry as a proxy for cross-client compatibility: clients

closer in parameter space are more likely to share similar optimization states and should thus be assigned larger weights during synchronization.

Specifically, for each pair of neighboring clients $(m, n) \in \mathcal{E}$, we define $A^t_{mn} = \exp(-\|\theta^{s,t}_m - \theta^{s,t}_n\|^2/\tau)$, where τ controls the sensitivity to model discrepancy. These pairwise affinities form the weighted adjacency matrix A^t, while D^t is the corresponding degree matrix with $D^t_{mm} = \sum_n A^t_{mn}$, characterizing the total affinity of client m to its neighbors. We then construct the graph Laplacian L^t and the geometry-aware mixing matrix W^t as:

$$W^t = I - \alpha_t L^t, \quad L^t = D^t - A^t, \quad A^t_{mn} = \exp\big(-\|\theta^{s,t}_m - \theta^{s,t}_n\|^2/\tau\big), \ (m,n) \in \mathcal{E}, \tag{9}$$

where $\alpha_t = 1/(1 + \max_m(D^t_{mm}))$ ensures valid mixing weights while retaining a positive self-weight for each client. This construction converts pairwise model compatibility into topology-aware consensus weights: geometrically compatible neighboring clients are assigned larger weights, whereas contributions from neighbors with dissimilar model states are automatically downweighted. Consequently, the consensus process adapts to the local geometry of client models rather than assigning equal importance to all connected clients, thereby reducing conflicts during parameter aggregation while preserving effective knowledge sharing among compatible neighbors.

4 Theoretical Analysis

Assumption 1 (Block Smoothness)**.** *For each client m, F_m is continuously differentiable. There exist constants $L_s, L_f, L_{sf}, L_{fs} > 0$ such that $\nabla_s F_m(\theta^{s,t}_m, \theta^{f,t}_m)$ is L_s-Lipschitz with respect to $\theta^{s,t}_m$ and L_{sf}-Lipschitz with respect to $\theta^{f,t}_m$, while $\nabla_f F_m(\theta^{s,t}_m, \theta^{f,t}_m)$ is L_f-Lipschitz with respect to $\theta^{f,t}_m$ and L_{fs}-Lipschitz with respect to $\theta^{s,t}_m$. We further define $\chi := \max\{L_{sf}, L_{fs}\}/\sqrt{L_s L_f}$.*

Assumption 2 (Bounded Variance)**.** *The stochastic gradients are unbiased. Moreover, their variances are uniformly bounded as $\mathbb{E}\|g^{s,t}_{m,k} - \mathbb{E}[g^{s,t}_{m,k}]\|^2 \le \sigma_s^2$ and $\mathbb{E}\|g^{f,t}_{m,k} - \mathbb{E}[g^{f,t}_{m,k}]\|^2 \le \sigma_f^2$.*

Assumption 3 (Slow Gradient Diversity)**.** *The heterogeneity of slow gradients is bounded. There exists $\sigma_g \ge 0$ such that $\frac{1}{M}\sum_{m=1}^M \|\nabla_s F_m(\theta^{s,t}_m, \theta^{f,t}_m) - \frac{1}{M}\sum_{m=1}^M \nabla_s F_m(\theta^{s,t}_m, \theta^{f,t}_m)\|^2 \le \sigma_g^2$.*

Theorem 1. *Suppose Assumptions 1–3 hold, and let $F^\star$ denote the minimum value of F. Let $\mathcal{W}$ be the set of consensus weight matrices considered during training, with $\lambda = \max_{W\in\mathcal{W}} \|W - P\|_2 < 1$. Let $\eta_s = a_s/(L_s K\sqrt{T})$ and $\eta_f = a_f/(L_f K\sqrt{T})$, where $a_s, a_f > 0$ satisfy the required stepsize and absorption conditions. If $a_s/L_s \ll a_f/L_f$, then $\eta_s \ll \eta_f$. Define $\Delta^{s,t} = \|\frac{1}{M}\sum_{m=1}^M \nabla_s F_m(\bar\theta^{s,t}_m, \theta^{f,t}_m)\|^2$ and $\Delta^{f,t} = \frac{1}{M}\sum_{m=1}^M \|\nabla_f F_m(\bar\theta^{s,t}_m, \theta^{f,t}_m)\|^2$. Then, we have:*

$$\frac{1}{T}\sum_{t=0}^{T-1}\Big(\frac{a_s}{L_s}\mathbb{E}[\Delta^{s,t}] + \frac{a_f}{L_f}\mathbb{E}[\Delta^{f,t}]\Big) \le \mathcal{O}\Big(\frac{\Delta}{\sqrt{T}} + \frac{\Gamma_1}{\sqrt{T}} + \frac{\Gamma_2}{T} + \frac{\Gamma_3}{(1-\lambda)^2 T}\Big), \tag{10}$$

where $\Delta = F(\bar\Theta^{s,0}, \Theta^{f,0}) - F^\star$, and $\Gamma_1, \Gamma_2, \Gamma_3$ are defined as:

$$\Gamma_1 = (1+\chi^2)\Big(\frac{a_s^2\sigma_s^2}{L_s K M} + \frac{a_f^2\sigma_f^2}{L_f K}\Big),$$
$$\Gamma_2 = \Big(\frac{a_s^3 + \chi^2 a_f a_s^2}{L_s}\Big)(\sigma_s^2 + \sigma_g^2) + \Big(\frac{a_f^3 + \chi^2 a_s a_f^2}{L_f}\Big)\sigma_f^2, \ \Gamma_3 = \Big(\frac{a_s^3 + \chi^2 a_f a_s^2}{L_s}\Big)(\sigma_s^2 + \sigma_g^2). \tag{11}$$

Remark 1. *Theorem 1 shows that **GeoSF** achieves an $\mathcal{O}(1/\sqrt{T})$ convergence rate over any connected graph satisfying $1 - \lambda > 0$. The convergence bound depends on the spectral gap $1 - \lambda$ and the variance terms σ_s^2, σ_f^2, and σ_g^2. A larger spectral gap, corresponding to stronger network connectivity, leads to better convergence performance. Moreover, by selecting more stable parameters for consensus, **GeoSF** reduces cross-client divergence among shared parameters, thereby limiting the effects of data heterogeneity and improving convergence performance.*

5 Experiments

5.1 Experimental Setup

Datasets and Model. We conducted experiments across different skew scenarios. For the **feature-skew** scenario, we used *Office10* (Gong et al., 2012) and *Digit5* (Zhou et al., 2020), which contain

Table 1: Comparison with other state-of-the-art algorithms in **feature-skew** scenarios. Best average results are in **bold**, while the best results for each client are underlined. See Sec. 5.2 for details.

Method	*Office10*						*Digit5*						
	Amaz	*Calt*	*DSLR*	*Webc*	***C.Avg***	***D.Avg***	*SYN*	*USPS*	*SVHN*	*MNTM*	*MNT*	***C.Avg***	***D.Avg***
SOLO	83.44	64.59	64.98	94.37	76.84±0.85	75.21±0.59	96.70	99.90	89.16	95.05	99.06	95.98±0.05	95.80±0.05
FedAvg	85.46	64.35	74.26	87.84	77.98±0.51	75.68±0.33	98.34	99.90	92.55	96.42	99.48	97.34±0.02	97.23±0.03
Ditto	84.62	64.00	84.39	85.14	79.54±1.37	75.53±0.82	98.00	99.88	92.12	95.94	99.38	97.06±0.04	96.94±0.04
D-PSGD	84.41	66.37	63.29	88.29	75.59±0.21	75.55±0.85	98.24	99.93	92.90	96.66	99.44	97.43±0.06	97.32±0.06
DFedAvgM	84.62	65.95	71.31	85.14	76.75±1.16	75.58±0.56	98.40	99.94	92.88	96.71	99.44	97.47±0.04	97.37±0.05
DisPFL	85.66	67.32	81.43	95.05	82.37±0.55	78.36±0.84	98.21	99.91	91.62	91.62	99.38	96.98±0.02	96.85±0.02
DFedSMDC	81.90	66.78	69.62	88.96	76.81±0.95	75.26±0.19	96.93	99.90	87.66	95.00	99.30	95.76±0.16	95.57±0.17
DFedPGP	76.97	60.32	59.49	86.71	70.87±1.38	69.64±0.53	98.06	99.89	91.80	96.29	99.47	97.10±0.06	96.98±0.06
GeoSF	88.16	73.07	85.23	94.14	**85.15**±0.76	**82.00**±0.69	98.73	99.93	92.28	97.43	99.40	**97.55**±0.02	**97.45**±0.02

Table 2: Performance comparison with state-of-the-art methods under **label skew** scenarios. where Pat stands for pathological heterogeneity. Please see details in Sec. 5.2.

Method	*Cifar10*			*Cifar100*			*Tiny-Imagenet*		
	Pat	$Dir_{0.1}$	$Dir_{0.5}$	Pat	$Dir_{0.1}$	$Dir_{0.5}$	Pat	$Dir_{0.1}$	$Dir_{0.5}$
SOLO	85.90±0.10	87.25±0.10	64.46±0.21	58.37±0.57	44.23±0.38	28.72±0.08	35.78±0.20	31.41±0.36	14.61±0.16
FedAvg	88.95±0.22	88.75±0.22	71.99±0.04	64.03±0.32	51.09±0.06	35.47±0.03	40.79±0.19	38.26±0.22	19.79±0.13
Ditto	88.34±0.19	87.75±0.16	70.27±0.23	62.49±0.44	48.87±0.29	33.80±0.16	38.56±0.16	35.72±0.17	18.10±0.28
D-PSGD	89.09±0.16	88.86±0.06	72.09±0.32	63.80±0.34	50.47±0.24	35.26±0.37	40.06±0.06	38.03±0.41	19.42±0.28
DFedAvgM	89.25±0.20	88.84±0.12	72.79±0.18	64.35±0.36	50.89±0.25	35.38±0.49	40.45±0.52	37.94±0.22	19.35±0.27
DisPFL	90.04±0.11	89.05±0.17	71.89±0.27	65.50±0.18	49.78±0.50	31.63±0.33	37.75±0.41	33.36±0.23	14.88±0.08
DFedSMDC	88.59±0.02	87.94±0.12	71.27±0.07	63.03±0.03	48.93±0.06	32.54±0.10	39.63±0.37	38.06±0.37	21.10±0.14
DFedPGP	87.25±0.12	87.04±0.12	69.07±0.28	59.50±0.44	45.86±0.21	30.68±0.10	36.77±0.36	35.19±0.19	19.05±0.40
GeoSF	**91.56**±0.06	**90.74**±0.11	**79.97**±0.27	**67.10**±0.25	**53.04**±0.33	**38.18**±0.06	**44.07**±0.24	**39.84**±0.12	**21.83**±0.15

four and five domains, respectively. For the **label-skew** scenario, we use four image datasets: *Cifar10* (Krizhevsky et al., 2009), *Cifar100* (Krizhevsky et al., 2009), *Flowers102* (Nilsback & Zisserman, 2008), and *Tiny-Imagenet* (Le & Yang, 2015), and one text dataset, *AGNews* (Zhang et al., 2015). We simulated label heterogeneity using two partitioning strategies, detailed in the appendix.

For model architectures, we adopt *ResNet18* (He et al., 2016) for image datasets and *fastText* (Joulin et al., 2017) for the text dataset by default. We further evaluate larger *ViT-B/16* (Dosovitskiy, 2020) models and smaller *CNN* (Zhang et al., 2023a) to assess performance across different model scales.

Baselines. We compare **GeoSF** against representative baselines, including SOLO, FedAvg (McMahan et al., 2017), Ditto (Li et al., 2021c), D-PSGD (Lian et al., 2017), DFedAvgM (Sun et al., 2022), DisPFL (Dai et al., 2022), DFedSMDC (Shi et al., 2026), and DFedPGP (Liu et al., 2024). For FedAvg, D-PSGD, and DFedAvgM, we report local model performance. To further assess our slow–fast mechanism, we remove the consensus module from **GeoSF** and compare the results with state-of-the-art PFL methods, including FedPer (Arivazhagan et al., 2019), FedRep (Collins et al., 2021), FedALA (Zhang et al., 2023b), GPFL (Zhang et al., 2023a), and FedAS (Yang et al., 2024).

Evaluation Metrics. We report performance at both the data and client levels using ***D.Avg*** and ***C.Avg***, respectively, with *D.Avg* serving as the primary evaluation metric. Let N_m^c and N_m denote the numbers of correctly classified samples and total samples for client m, respectively. The two metrics are defined as $\textit{D.Avg} = \sum_{m=1}^{M} N_m^c / \sum_{m=1}^{M} N_m$; $\textit{C.Avg} = \frac{1}{M} \sum_{m=1}^{M} N_m^c / N_m$.

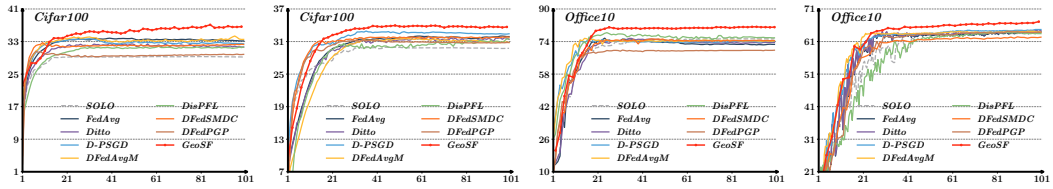
Implementation Details. For all datasets, we use a default batch size of 128, except for *Tiny-Imagenet* (Le & Yang, 2015), where the batch size is set to 256. We adopt SGD with a learning rate of 0.05, set the number of local epochs to 5 and the number of clients to 20, and train all methods for 200 communication rounds. We report the mean and standard deviation over three independent runs.

5.2 Comparison to State-of-the-Arts.

Different Heterogeneity Scenarios. We evaluate **GeoSF** against representative methods under both *feature-skewed* and *label-skewed* scenarios, with results reported in Tabs. 1 and 2. **GeoSF** consistently outperforms competing methods across these settings. Specifically, it achieves 82.00% accuracy on *Office10* (Gong et al., 2012), exceeding the strongest baseline by 3.64%. Under *label*

Table 3: Performance comparison across **diverse experimental settings**, including different model architectures, local epochs E, larger client scales M, and text datasets. See Sec. 5.2 for details.

Method	*Office10*				*Cifar100*		*AGNews*		
	Model		*Local epochs*		*Scalability*		*Text*		
	CNN	ViT	$E=1$	$E=10$	$M=50$	$M=100$	Pat	$Dir_{0.1}$	$Dir_{0.5}$
SOLO	73.74±0.28	63.62±1.27	73.71±0.45	75.45±0.10	43.38±0.20	41.26±0.20	71.75±1.40	96.08±0.29	68.81±3.26
FedAvg	75.16±0.77	64.22±0.30	72.58±1.45	76.95±0.97	50.91±0.45	47.75±0.20	66.75±8.81	96.05±0.00	70.39±0.14
Ditto	74.71±0.69	64.59±0.36	75.21±1.16	76.55±0.58	47.19±0.26	42.65±0.32	65.73±9.02	95.85±0.11	70.77±0.71
D-PSGD	74.29±0.62	64.69±1.33	71.50±0.86	76.34±0.56	50.62±0.31	48.01±0.24	68.91±9.39	96.08±0.00	70.78±0.60
DFedAvgM	74.05±0.57	65.14±0.68	69.58±1.34	77.47±0.10	50.75±0.38	48.14±0.34	73.30±9.61	96.08±0.00	72.40±3.24
DisPFL	73.13±0.07	63.82±1.04	76.97±0.46	79.44±0.54	42.40±0.19	47.22±0.27	58.51±1.39	95.91±0.00	72.00±0.27
DFedSMDC	75.02±0.67	62.51±0.46	73.44±1.15	75.39±0.97	47.28±0.12	44.04±0.39	80.38±0.32	95.86±0.11	74.83±2.10
DFedPGP	73.63±0.21	64.09±0.45	68.58±0.43	71.60±0.72	42.75±0.26	39.90±0.33	80.92±0.75	95.86±0.11	74.67±1.66
GeoSF	**76.42**±0.51	**66.35**±0.60	**80.99**±0.17	**82.04**±0.16	**51.29**±0.32	**49.02**±0.36	**93.21**±0.06	**97.75**±0.20	**85.42**±3.10



(a) *Cifar100* & *ResNet18*. (b) *Cifar100* & *ViT*. (c) *Office10* & *ResNet18*. (d) *Office10* & *ViT*.

Figure 3: **Convergence curves** of different algorithms across datasets (*Cifar100* and *Office10*) and model architectures (*ResNet18* and *ViT*). Please see details in Sec. 5.2.

skew on *Cifar10* (Krizhevsky et al., 2009), it surpasses the best baseline by 7.18%. As shown in Tab. 3, **GeoSF** also performs best on the text dataset, improving by 12.29% under the *Pat* setting. Moreover, its consistent gains over local training confirm the advantages of cross-client parameter consensus. These results demonstrate that **GeoSF** effectively handles diverse forms and degrees of data heterogeneity by selecting suitable parameters for consensus.

Different Experimental Configurations. We further evaluate **GeoSF** across different model architectures, numbers of local training epochs, and client scales, with the results reported in Tab. 3. **GeoSF** consistently outperforms the competing methods using both *CNN* and *ViT* architectures, demonstrating that its effectiveness is not tied to a particular backbone. At $E=1$ and $E=10$ local epochs, **GeoSF** surpasses the strongest baseline by 4.02% and 2.60%, respectively, showing that its advantage persists under both short and extended local training. When the number of clients increases to 50 and 100, **GeoSF** achieves accuracies of 51.29% and 49.02%, respectively, while retaining the best performance in both settings. Overall, these results confirm the robustness and scalability of **GeoSF** across variations in model architecture, local epochs, and system scale.

Different Communication Topologies. We evaluate **GeoSF** on *Cifar100* (Krizhevsky et al., 2009) under five representative communication topologies: ring, grid, exponential, random undirected, and fully connected graphs, with the results reported in Tab. 4. **GeoSF** achieves accuracies of 42.21%, 51.24%, 52.33%, 53.04%, and 55.65%, respectively, outperforming the competing methods under every topology. Notably, under the fully connected topology, it outperforms the best baseline by 4.62%. Moreover, the results show that performance generally improves as network connectivity increases. This trend is consistent with our theoretical analysis: communication graphs with stronger connectivity generally exhibit larger spectral gaps, which facilitate faster consensus and more efficient information propagation across clients.

Table 4: Performance under **different communication topologies**. See details in Sec. 5.2.

Method	*Ring*	*Grid*	*Exp*	*Undirect*	*Full*
D-PSGD	45.47	47.31	48.87	50.47	50.60
DFedAvgM	45.50	47.48	49.44	50.89	51.03
DisPFL	45.53	46.20	46.40	49.78	50.20
DFedSMDC	45.84	48.11	48.37	48.93	48.93
DFedPGP	44.09	43.99	44.62	45.86	45.22
GeoSF	**49.21**	**51.24**	**52.33**	**53.04**	**55.65**

Convergence Visualization. Fig. 3 compares the convergence behavior of different methods across multiple datasets and model architectures. Under various scenarios, **GeoSF** rapidly reaches competitive accuracy and remains stable throughout subsequent training, demonstrating efficient and robust convergence under diverse experimental configurations.

5.3 Ablation Study

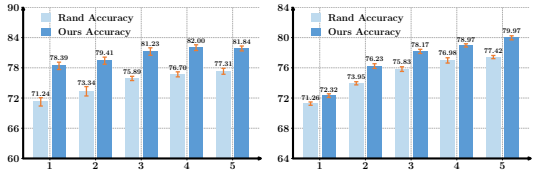
Effectiveness of Different Modules. We assess the contributions of Two-Timescale Update (TTU), Slow-Fast Consensus (SFC), and Geometry-Aware Weighting (GAW) through ablation studies in Tab. 5. Individually, each component improves accuracy over the baseline by 0.90%–2.14%. Combining these components yields larger gains. Specifically, the combination of TTU and SFC achieves an accuracy of 80.18%, while integrating all three components further increases the accuracy to 82.00%, outperforming the baseline by 6.45%. These progressive improvements demonstrate that the three components are complementary, jointly addressing different aspects of model optimization and cross-client consensus under heterogeneous data distributions.

Slow-Fast Parameter Selection. We evaluate the effectiveness of the slow–fast decomposition strategy in Fig. 4. Specifically, we fix the learning rate of slow parameters at $\eta_s = 0.05$ and vary the fast-parameter learning rate through the ratio $\alpha = \eta_f/\eta_s$. Under each setting, we compare random selection of shared parameters with selection based on the stability score defined in Eq. (5). Score-based selection consistently outperforms random selection across all values of α, showing that its advantage is not tied to a particular learning-rate configuration. This suggests that the decomposition identifies stable parameters suitable for consensus, reducing conflicts from data heterogeneity and improving knowledge exchange among clients.

Table 5: **Ablation analysis** on the contributions of different modules. Please see details in Sec. 5.3.

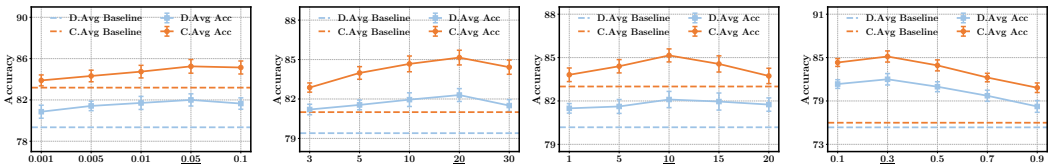
Module			*Test Accuracy*					
TTU	SFC	GAW	*Amaz*	*Calt*	*DSLR*	*Webc*	***C.Avg***	***D.Avg***
			84.41	66.37	63.29	88.29	75.59	75.55
✓			85.83	68.02	69.35	90.41	78.41	77.45
	✓		85.03	70.42	68.71	86.31	77.62	77.69
		✓	84.83	67.02	68.35	89.41	77.41	76.45
✓	✓		87.00	70.84	81.12	93.05	83.00	80.18
✓		✓	86.22	70.28	72.15	95.95	81.15	79.42
	✓	✓	85.18	69.93	68.35	93.92	79.34	78.39
✓	✓	✓	88.16	73.07	85.23	94.14	**85.15**	**82.00**

Hyperparameter Analysis. Fig. 5 analyzes the sensitivity of **GeoSF** to μ, s, τ, and ρ. **GeoSF** maintains strong performance across reasonable ranges, demonstrating its robustness to parameter selection. In particular, it achieves an accuracy of 82.00% when $\mu = 0.05$. The results for ρ and s reveal a trade-off between collaboration and personalization. An excessively large ρ may force heterogeneous parameters into consensus and trigger conflicts, whereas a small ρ restricts shared knowledge exchange. Similarly, overly frequent consensus on fast parameters may weaken personalization, while infrequent consensus increases the risk of local overfitting. Thus, appropriate hyperparameters are crucial for balancing collaboration and local adaptation.



(a) Comparison on ***Office10*** (b) Comparison on ***Cifar10***

Figure 4: Experimental validation of the effectiveness of the **decomposition strategy**.



(a) Shrinkage coefficient μ (b) Communication interval s (c) Temperature coefficient τ (d) Shared-parameter ratio ρ

Figure 5: **Hyperparameter analysis** of **GeoSF** for shrinkage coefficient μ, communication interval s, temperature coefficient τ, and shared-parameter ratio p. See details in Sec. 5.3.

6 Conclusion

In this paper, we propose **GeoSF** to mitigate performance degradation caused by data heterogeneity in decentralized federated learning. By exploiting differences in parameter behavior across clients, **GeoSF** dynamically separates model parameters according to their cross-client sign consistency and variance. A two-timescale update and slow-fast consensus strategy balances cross-client consistency with local adaptability, while geometry-aware weighting controls information exchange based on model compatibility to reduce conflicts among heterogeneous clients. We establish the convergence of **GeoSF** for nonconvex objectives. Extensive experiments under both feature and label skew further show that **GeoSF** consistently outperforms representative baseline methods.

REFERENCES

Manoj Ghuhan Arivazhagan, Vinay Aggarwal, Aaditya Kumar Singh, and Sunav Choudhary. Federated learning with personalization layers. *arXiv preprint arXiv:1912.00818*, 2019. 3, 7, 14, 15, 16, 18

Enrique Tomás Martínez Beltrán, Mario Quiles Pérez, Pedro Miguel Sánchez Sánchez, Sergio López Bernal, Gérôme Bovet, Manuel Gil Pérez, Gregorio Martínez Pérez, and Alberto Huertas Celdrán. Decentralized federated learning: Fundamentals, state of the art, frameworks, trends, and challenges. *IEEE Communications Surveys & Tutorials*, 25(4):2983–3013, 2023. 1

Fei Chen, Mi Luo, Zhenhua Dong, Zhenguo Li, and Xiuqiang He. Federated meta-learning with fast convergence and efficient communication. *arXiv preprint arXiv:1802.07876*, 2018. 14

Min Chen, Yang Xu, Hongli Xu, and Liusheng Huang. Enhancing decentralized federated learning for non-iid data on heterogeneous devices. In *2023 IEEE 39th International Conference on Data Engineering (ICDE)*, pp. 2289–2302. IEEE, 2023. 14

Wu-Chun Chung, Chao-Ai Lo, Yan-Hui Lin, Zhi-Hao Chen, and Che-Lun Hung. Decentralized federated learning with non-iid data: Challenges, trends, and future opportunities. *ACM Computing Surveys*, 58(8):1–41, 2026. 1

Liam Collins, Hamed Hassani, Aryan Mokhtari, and Sanjay Shakkottai. Exploiting shared representations for personalized federated learning. In *International conference on machine learning*, pp. 2089–2099. PMLR, 2021. 3, 7, 14, 15, 16, 17, 18

Rong Dai, Li Shen, Fengxiang He, Xinmei Tian, and Dacheng Tao. Dispfl: Towards communication-efficient personalized federated learning via decentralized sparse training. *arXiv preprint arXiv:2206.00187*, 2022. 1, 2, 7, 14, 15, 16, 17, 19

Alexey Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 7, 15, 19

Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach. *Advances in neural information processing systems*, 33:3557–3568, 2020. 14

Boqing Gong, Yuan Shi, Fei Sha, and Kristen Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *2012 IEEE conference on computer vision and pattern recognition*, pp. 2066–2073. IEEE, 2012. 6, 7, 14, 15, 17

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016. 7, 15, 18, 19

Kevin Hsieh, Amar Phanishayee, Onur Mutlu, and Phillip Gibbons. The non-iid data quagmire of decentralized machine learning. In *International Conference on Machine Learning*, pp. 4387–4398. PMLR, 2020. 1

Wenke Huang, Mang Ye, Zekun Shi, He Li, and Bo Du. Rethinking federated learning with domain shift: A prototype view. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16312–16322. IEEE, 2023. 1

Eunjeong Jeong and Marios Kountouris. Personalized decentralized federated learning with knowledge distillation. In *ICC 2023-IEEE International Conference on Communications*, pp. 1982–1987. IEEE, 2023. 1

Yihan Jiang, Jakub Konečnỳ, Keith Rush, and Sreeram Kannan. Improving federated learning personalization via model agnostic meta learning. *arXiv preprint arXiv:1909.12488*, 2019. 14

Zhanhong Jiang, Aditya Balu, Chinmay Hegde, and Soumik Sarkar. Collaborative deep learning in fixed topology networks. *Advances in Neural Information Processing Systems*, 30, 2017. 17

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomáš Mikolov. Bag of tricks for efficient text classification. In *Proceedings of the 15th conference of the European chapter of the association for computational linguistics: volume 2, short papers*, pp. 427–431, 2017. 7, 15

Shivam Kalra, Junfeng Wen, Jesse C Cresswell, Maksims Volkovs, and Hamid R Tizhoosh. Decentralized federated learning through proxy model sharing. *Nature communications*, 14(1):2899, 2023. 14

Jiawen Kang, Dongdong Ye, Jiangtian Nie, Jiang Xiao, Xianjun Deng, Siming Wang, Zehui Xiong, Rong Yu, and Dusit Niyato. Blockchain-based federated learning for industrial metaverses: Incentive scheme with optimal aoi. In *2022 IEEE International Conference on Blockchain (Blockchain)*, pp. 71–78. IEEE, 2022. 3

Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. Scaffold: Stochastic controlled averaging for federated learning. In *International conference on machine learning*, pp. 5132–5143. PMLR, 2020. 1

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 7, 8, 15, 19

Anusha Lalitha, Shubhanshu Shekhar, Tara Javidi, and Farinaz Koushanfar. Fully decentralized federated learning. In *Third workshop on bayesian deep learning (NeurIPS)*, volume 2, pp. 1–9, 2018. 17

Anusha Lalitha, Osman Cihan Kilinc, Tara Javidi, and Farinaz Koushanfar. Peer-to-peer federated learning on graphs. *arXiv preprint arXiv:1901.11173*, 2019. 14

Ya Le and Xuan Yang. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7):3, 2015. 7, 15, 17

Boyue Li, Shicong Cen, Yuxin Chen, and Yuejie Chi. Communication-efficient distributed optimization in networks with gradient tracking and variance reduction. *Journal of Machine Learning Research*, 21(180):1–51, 2020a. 14

Jun Li, Yumeng Shao, Kang Wei, Ming Ding, Chuan Ma, Long Shi, Zhu Han, and H Vincent Poor. Blockchain assisted decentralized federated learning (blade-fl): Performance analysis and resource allocation. *IEEE Transactions on Parallel and Distributed Systems*, 33(10):2401–2415, 2021a. 3

Qinbin Li, Zeyi Wen, Zhaomin Wu, Sixu Hu, Naibo Wang, Yuan Li, Xu Liu, and Bingsheng He. A survey on federated learning systems: Vision, hype and reality for data privacy and protection. *IEEE Transactions on Knowledge and Data Engineering*, 35(4):3347–3366, 2021b. 1

Qinglun Li, Li Shen, Guanghao Li, Quanjun Yin, and Dacheng Tao. Dfedadmm: Dual constraint controlled model inconsistency for decentralize federated learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(6):4803–4815, 2025a. 3, 14

Qinglun Li, Miao Zhang, Nan Yin, Quanjun Yin, Li Shen, and Xiaochun Cao. Asymmetrically decentralized federated learning. *IEEE Transactions on Computers*, 74(8):2745–2756, 2025b. 1

Qinglun Li, Yingqi Liu, Miao Zhang, Xiaochun Cao, Quanjun Yin, and Li Shen. Unveiling the power of multiple gossip steps: A stability-based generalization analysis in decentralized training. *Advances in Neural Information Processing Systems*, 38:73808–73827, 2026a. 1

Qinglun Li, Miao Zhang, Yingqi Liu, Quanjun Yin, Li Shen, and Xiaochun Cao. Boosting the performance of decentralized federated learning via catalyst acceleration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2026b. 14

Shuangtong Li, Tianyi Zhou, Xinmei Tian, and Dacheng Tao. Learning to collaborate in decentralized learning of personalized models. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9756–9765. IEEE, 2022. 3

Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions. *IEEE signal processing magazine*, 37(3):50–60, 2020b. 1

Tian Li, Shengyuan Hu, Ahmad Beirami, and Virginia Smith. Ditto: Fair and robust federated learning through personalization. In *International conference on machine learning*, pp. 6357–6368. PMLR, 2021c. 3, 7, 14, 15, 16, 17, 19

Xiangru Lian, Ce Zhang, Huan Zhang, Cho-Jui Hsieh, Wei Zhang, and Ji Liu. Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent. *Advances in neural information processing systems*, 30, 2017. 3, 7, 14, 15, 16, 17, 18, 19

Paul Pu Liang, Terrance Liu, Liu Ziyin, Nicholas B Allen, Randy P Auerbach, David Brent, Ruslan Salakhutdinov, and Louis-Philippe Morency. Think locally, act globally: Federated learning with local and global representations. *arXiv preprint arXiv:2001.01523*, 2020. 1

Yingqi Liu, Yifan Shi, Qinglun Li, Baoyuan Wu, Xueqian Wang, and Li Shen. Decentralized directed collaboration for personalized federated learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 23168–23178, 2024. 2, 3, 7, 14, 15, 16, 17, 19

Jun Luo and Shandong Wu. Adapt to adaptation: Learning personalization for cross-silo federated learning. In *IJCAI: proceedings of the conference*, volume 2022, pp. 2166, 2022. 14

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pp. 1273–1282. PMLR, 2017. 1, 7, 16, 19

Angelia Nedic and Asuman Ozdaglar. Distributed subgradient methods for multi-agent optimization. *IEEE Transactions on automatic control*, 54(1):48–61, 2009. 1

Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *2008 Sixth Indian conference on computer vision, graphics & image processing*, pp. 722–729. IEEE, 2008. 7, 15, 18

Jaehoon Oh, Sangmook Kim, and Se-Young Yun. Fedbabu: Towards enhanced representation for federated image classification. *arXiv preprint arXiv:2106.06042*, 2021. 1

Krishna Pillutla, Kshitiz Malik, Abdel-Rahman Mohamed, Mike Rabbat, Maziar Sanjabi, and Lin Xiao. Federated learning with partial model personalization. In *International Conference on Machine Learning*, pp. 17716–17758. PMLR, 2022. 20

Abdurakhmon Sadiev, Ekaterina Borodich, Aleksandr Beznosikov, Darina Dvinskikh, Saveliy Chezhegov, Rachael Tappenden, Martin Takáč, and Alexander Gasnikov. Decentralized personalized federated learning: Lower bounds and optimal algorithm for all personalization modes. *EURO Journal on Computational Optimization*, 10:100041, 2022. 3

Yifan Shi, Li Shen, Kang Wei, Yan Sun, Bo Yuan, Xueqian Wang, and Dacheng Tao. Improving the model consistency of decentralized federated learning. In *International conference on machine learning*, pp. 31269–31291. PMLR, 2023. 1, 14

Yifan Shi, Yingqi Liu, Yan Sun, Zihao Lin, Xueqian Wang, Li Shen, Xiaochun Cao, and Dacheng Tao. Decentralized partial model personalization with guaranteed nonconvex convergence. *IEEE Transactions on Networking*, 2026. 2, 3, 7, 14, 15, 16, 17, 19, 21

Yi Sui, Junfeng Wen, Yenson Lau, Brendan Leigh Ross, and Jesse C Cresswell. Find your friends: Personalized federated learning with the right collaborators. *arXiv preprint arXiv:2210.06597*, 2022. 1

Tao Sun, Dongsheng Li, and Bao Wang. Decentralized federated averaging. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(4):4289–4301, 2022. 3, 7, 14, 15, 16, 17, 19

Canh T Dinh, Nguyen Tran, and Josh Nguyen. Personalized federated learning with moreau envelopes. *Advances in neural information processing systems*, 33:21394–21405, 2020. 3, 14

Hossein Taheri, Aryan Mokhtari, Hamed Hassani, and Ramtin Pedarsani. Quantized decentralized stochastic learning over directed graphs. In *International conference on machine learning*, pp. 9324–9333. PMLR, 2020. 14

Lun Wang, Yang Xu, Hongli Xu, Min Chen, and Liusheng Huang. Accelerating decentralized federated learning in heterogeneous edge computing. *IEEE Transactions on Mobile Computing*, 22(9):5001–5016, 2022. 1

Stefanie Warnat-Herresthal, Hartmut Schultze, Krishnaprasad Lingadahalli Shastry, Sathyanarayanan Manamohan, Saikat Mukherjee, Vishesh Garg, Ravi Sarveswara, Kristian Händler, Peter Pickkers, N Ahmad Aziz, et al. Swarm learning for decentralized and confidential clinical machine learning. *Nature*, 594(7862):265–270, 2021. 14

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

Xiyuan Yang, Wenke Huang, and Mang Ye. Fedas: Bridging inconsistency in personalized federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11986–11995, 2024. 3, 7, 14, 15, 16, 17, 18

Haishan Ye and Tong Zhang. Deepca: Decentralized exact pca with linear convergence rate. *Journal of Machine Learning Research*, 22(238):1–27, 2021. 14

Haishan Ye, Ziang Zhou, Luo Luo, and Tong Zhang. Decentralized accelerated proximal gradient descent. *Advances in Neural Information Processing Systems*, 33:18308–18317, 2020. 14

Liangqi Yuan, Ziran Wang, Lichao Sun, Philip S Yu, and Christopher G Brinton. Decentralized federated learning: A survey and perspective. *IEEE Internet of Things Journal*, 11(21):34617–34638, 2024. 1

Anqi Zhang, Ping Zhao, Wenke Lu, and Guanglin Zhang. Personalized decentralized federated learning: A privacy-enhanced and byzantine-resilient approach. *IEEE Transactions on Computational Social Systems*, 12(5):3206–3217, 2025. 2

Jianqing Zhang, Yang Hua, Hao Wang, Tao Song, Zhengui Xue, Ruhui Ma, Jian Cao, and Haibing Guan. Gpfl: Simultaneously learning global and personalized feature information for personalized federated learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5041–5051, 2023a. 3, 7, 14, 15, 16, 17, 18, 19

Jianqing Zhang, Yang Hua, Hao Wang, Tao Song, Zhengui Xue, Ruhui Ma, and Haibing Guan. Fedala: Adaptive local aggregation for personalized federated learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 11237–11244, 2023b. 7, 15, 16, 17, 18

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. 15

Michael Zhang, Karan Sapra, Sanja Fidler, Serena Yeung, and Jose M Alvarez. Personalized federated learning with first order model optimization. *arXiv preprint arXiv:2012.08565*, 2020. 14

Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28, 2015. 7, 15

Kaiyang Zhou, Yongxin Yang, Timothy Hospedales, and Tao Xiang. Learning to generate novel domains for domain generalization. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVI 16*, pp. 561–578. Springer, 2020. 6, 14, 15, 18

Tongtian Zhu, Fengxiang He, Lan Zhang, Zhengyang Niu, Mingli Song, and Dacheng Tao. Topology-aware generalization of decentralized sgd. In *International Conference on Machine Learning*, pp. 27479–27503. PMLR, 2022. 3

Further details on our work and results are provided in the appendix. Its contents are as follows:

- Appendix A: More detailed discussion of related work.
- Appendix B: More details about the configuration in the experiment.
- Appendix C: More comparative experiments and visualization results.
- Appendix D: Proof of the convergence in Theorem 2.
- Appendix E: A statement on the reproducibility of this work.
- Appendix F: A statement on the use of AI in this work.
- Appendix G: Discussion on the limitations and future work.
- Appendix H: Pseudocode of the key function in the proposed method.

A Related Work

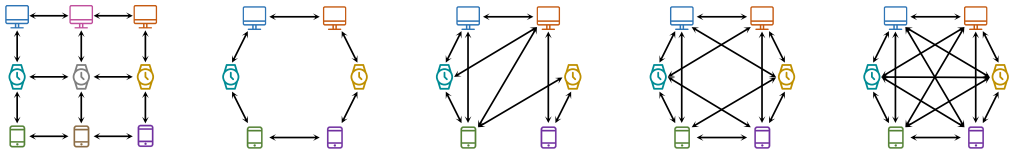
Personalized Federated Learning. Federated learning enables knowledge sharing by aggregating models trained on distributed client data. However, a single global model often fails to accommodate heterogeneous local data distributions. Personalized Federated Learning (PFL) addresses this limitation by introducing client-specific model components or optimization objectives (Zhang et al., 2023a; Yang et al., 2024). Existing PFL methods can be broadly categorized into meta-learning, model partitioning, personalized aggregation, and regularization-based approaches. Meta-learning methods (Jiang et al., 2019), such as Per-FedAvg (Fallah et al., 2020) and FedMeta (Chen et al., 2018), learn a global model that can be rapidly personalized through local fine-tuning. Model-partitioning methods, including FedPer (Arivazhagan et al., 2019), FedRep (Collins et al., 2021), and FedAS (Yang et al., 2024), share the model backbone while retaining client-specific classification heads. FedFomo (Zhang et al., 2020) and APPLE (Luo & Wu, 2022) construct personalized models by adaptively aggregating models from other clients. Regularization-based methods, such as Ditto (Li et al., 2021c) and pFedMe (T Dinh et al., 2020), promote knowledge transfer by constraining personalized models relative to the global model. Although these approaches improve local adaptability, they often rely on predefined model-partitioning schemes and centralized aggregation, which limits flexibility and introduces a potential single point of failure.

Decentralized Federated Learning. Decentralized Federated Learning (DFL) eliminates the central server and enables peer-to-peer communication among neighboring clients (Ye & Zhang, 2021; Taheri et al., 2020; Li et al., 2020a; Chen et al., 2023), thereby avoiding single points of failure and improving deployment flexibility (Kalra et al., 2023; Ye et al., 2020; Lalitha et al., 2019; Warnat-Herresthal et al., 2021). Existing DFL methods can be broadly divided into centralized and personalized approaches. Among centralized approaches, D-PSGD (Lian et al., 2017) pioneered the integration of stochastic gradient descent with decentralized consensus, while DFedAvgM (Sun et al., 2022), DFedADMM (Li et al., 2025a), and DFedCata (Li et al., 2026b) accelerated convergence by refining local optimization. DFedSAM (Shi et al., 2023) further improved model generalization through adversarial perturbations. However, learning a single shared model can make it difficult to accommodate client-specific needs. Personalized approaches address this issue: DisPFL (Dai et al., 2022) uses personalized masks to reduce communication costs and support collaboration among clients, DFedPGP (Liu et al., 2024) adopts a personalized backbone architecture, and DFedSMC (Shi et al., 2026) incorporates sharpness-aware optimization to improve the robustness of personalized models. Despite these advances, most existing methods rely on predefined parameter decompositions and overlook differences in parameter behavior during local optimization and aggregation, potentially limiting performance under data heterogeneity.

B Experimental Details

B.1 Datasets and models

We evaluate our method against representative state-of-the-art baselines under two forms of statistical heterogeneity: feature skew and label skew. For **feature skew**, we conduct experiments on *Office10* (Gong et al., 2012) and *Digit5* (Zhou et al., 2020), both of which contain 10 classes and treat each domain as an individual client. Specifically, *Office10* consists of four domains, namely



(a) Grid (b) Ring (c) Undirect (d) Exp (e) Full

Figure 6: Illustration of five representative **communication topologies**, including (a) grid, (b) ring, (c) random undirected graph, (d) exponential (Exp), and (e) fully connected graph.

Amazon (Amaz), *Caltech (Calt)*, *DSLR*, and *Webcam (Webc)*, corresponding to four clients. *Digit5* consists of five domains, namely *SVHN*, *USPS*, *SynthDigits (SYN)*, *MNIST-M (MNTM)*, and *MNIST*, corresponding to five clients. The domain-specific visual characteristics of these datasets provide natural benchmarks for evaluating robustness to heterogeneous feature distributions.

Table 6: Detailed **experimental configurations** for different datasets, where *Sites* denotes the default number of clients assigned to each dataset. Please see more details in Sec. B.1.

Experimental Dataset	Scenario	Modality	Class	Sites	Train / Test	Batch	Input Size
Office10 (Gong et al., 2012)	Feature skew	Image	10	4	3.8K / 1.3K	128	$3 \times 64 \times 64$
Digit5 (Zhou et al., 2020)	Feature skew	Image	10	5	122K / 41K	128	$3 \times 32 \times 32$
Cifar10 (Krizhevsky et al., 2009)	Label skew	Image	10	20	45K / 15K	128	$3 \times 32 \times 32$
Cifar100 (Krizhevsky et al., 2009)	Label skew	Image	100	20	45K / 15K	128	$3 \times 32 \times 32$
Flowers102 (Nilsback & Zisserman, 2008)	Label skew	Image	102	20	6K / 2K	128	$3 \times 64 \times 64$
Tiny (Le & Yang, 2015)	Label skew	Image	200	20	83K / 28K	256	$3 \times 64 \times 64$
AGNews (Zhang et al., 2015)	Label skew	Text	4	20	96K / 32k	128	200 Tokens

For **label skew**, we conduct experiments on four image datasets, namely *Cifar10* (Krizhevsky et al., 2009), *Cifar100* (Krizhevsky et al., 2009), *Flowers102* (Nilsback & Zisserman, 2008), and *Tiny-Imagenet* (Le & Yang, 2015), as well as the text dataset *AGNews* (Zhang et al., 2015). These datasets contain 10, 100, 102, 200, and 4 classes, respectively, covering tasks with substantially different levels of classification complexity. We employ 20 clients by default and further consider settings with 50 and 100 clients to evaluate scalability. Two partitioning strategies are adopted to simulate label heterogeneity. Under the pathological partition, each client is randomly a fixed number of randomly sampled classes: 2 classes for *AGNews* (Zhang et al., 2015) and *Cifar10* (Krizhevsky et al., 2009), 10 classes for *Cifar100* (Krizhevsky et al., 2009), 15 classes for *Flowers102* (Nilsback & Zisserman, 2008), and 20 classes for *Tiny-Imagenet* (Le & Yang, 2015). Under the Dirichlet partition, denoted by Dir_β, class proportions are sampled from a Dirichlet distribution to produce varying degrees of label skew. Following (Zhang et al., 2024), we repartition each dataset into training and test sets at a ratio of $0.75/0.25$. Detailed dataset statistics and experimental configurations are summarized in Tab. 6.

For the model architectures, we use *ResNet18* (He et al., 2016) as the default backbone for image classification and fastText (Joulin et al., 2017) for text classification. To further evaluate the robustness of **GeoSF** to architectural variations, we repeat the experiments using a *CNN* (Zhang et al., 2023a), *ResNet50* (He et al., 2016), and *ViT-B/16* (Dosovitskiy, 2020). These architectures span convolutional networks of varying capacities as well as a Transformer-based model.

B.2 Baseline Methods

We evaluate **GeoSF** against representative state-of-the-art decentralized baselines, including D-PSGD (Lian et al., 2017), DFedAvgM (Sun et al., 2022), DisPFL (Dai et al., 2022), DFedPGP (Liu et al., 2024), and DFedSMDC (Shi et al., 2026). To isolate the effectiveness of the slow–fast learning mechanism, we remove the geometry-aware weighted consensus module and compare the resulting variant with representative personalized federated learning methods, including FedPer (Arivazhagan et al., 2019), FedRep (Collins et al., 2021), Ditto (Li et al., 2021c), FedALA (Zhang et al., 2023b), GPFL (Zhang et al., 2023a), and FedAS (Yang et al., 2024). This two-level evaluation demonstrates both the overall competitiveness of **GeoSF** and the individual benefits of its core components.

First, we briefly introduce the baselines for decentralized federated learning.

- D-PSGD (Lian et al., 2017): By adopting decentralized parallel stochastic gradient descent, each node exchanges and averages parameters with its neighbors, alleviating the communication bottleneck caused by a central server. In our implementation, each node performs multiple local epochs rather than a stochastic gradient step before communication.
- DFedAvgM (Sun et al., 2022): Introduces local momentum into decentralized FedAvg to accelerate local convergence and stabilize the training process.
- DisPFL (Dai et al., 2022): Introduces personalized sparse masks that adapt to empirical risk, reducing communication and enabling local personalization.
- DFedPGP (Liu et al., 2024): Achieves decentralized personalized federated learning by decoupling the model into a shared backbone and a private classifier, while applying gradient pushing with a debiasing mechanism over directed asymmetric networks.
- DFedSMDC (Shi et al., 2026): Extends DFedPGP with adversarial perturbations to the backbone, smoothing the loss landscape and improving personalized-model generalization.

We then describe the representative personalized federated learning methods used for comparison.

- SOLO: Trains each client model exclusively on its local dataset without model sharing.
- FedAvg (McMahan et al., 2017): Selects a subset of clients for local training and aggregates their updated models on the server, while the locally updated models can also serve as baselines for evaluating personalized methods.
- FedPer (Arivazhagan et al., 2019): Updates the entire model locally, aggregates only the shared backbone, and retains the classification head on each client to achieve personalization.
- FedRep (Collins et al., 2021): Decouples the model into a shared backbone and a local classifiers, which are optimized alternately while the other component remains frozen.
- FedALA (Zhang et al., 2023b): Adaptively incorporates beneficial global knowledge through parameter-wise aggregation weights learned before local training.
- Ditto (Li et al., 2021c): Facilitates mutual learning and knowledge transfer by maintaining a global model and a local model for each client and enforcing regularization between them.
- GPFL (Zhang et al., 2023a): Introduces learnable global class embeddings to encode shared knowledge and employs conditional gates to construct global and personalized branches, enabling each client to jointly learn globally shared and locally personalized representations.
- FedAS (Yang et al., 2024): Addresses intra-client inconsistency and inter-client asynchrony in PFL by aligning feature representations from new and previous backbones on each client and performing Fisher-information-based weighted aggregation at the server.

B.3 Detailed Configuration

Table 7: A list of notations and descriptions.

	Description		Description
M	Number of clients	T	Communication rounds
E	Local epochs	λ	Spectral contraction.
s	Communication interval	μ	Regularization coefficient
η_s	Slow learning rate	W	Consensus weight matrix
η_f	Fast learning rate	A	Model distance matrix
m_m^s	Slow parameter mask	ρ	Shared parameter ratio
m_m^f	Fast parameter mask	τ	Temperature parameter
θ_m	Model of client m	C	Consensus rounds

In addition to the hyperparameter settings reported in the main paper, we set the default learning rate for the slow parameters of **GeoSF** to 0.05 and selected the optimal learning rate for the fast parameters from $\{0.05, 0.1, 0.15, 0.2, 0.25\}$. For the baseline methods, we follow the hyperparameter search strategies recommended in their original studies. Specifically, for DFedAvgM (Sun et al., 2022), we select the momentum coefficient μ from $\{0.01, 0.1, 0.3, 0.5\}$ and set $\mu = 0.1$. For DisPFL (Dai et al., 2022), we select the sparsity level from $\{0.1, 0.3, 0.5, 0.7, 0.9\}$ and set it to 0.5. For DFedPGP (Liu et al., 2024), we set the number of classifier update rounds E_l equal to the number of backbone update rounds E_g. For DFedSMDC (Shi et al., 2026), we select the perturbation magnitude ρ from $\{0.01, 0.1, 0.3, 0.5, 0.7\}$ and set $\rho = 0.1$.

For the PFL baselines, we employ grid search to determine the best-performing configurations. SOLO and FedAvg (McMahan et al., 2017) introduce no method-specific hyperparameters. FedPer (Arivazhagan et al., 2019) adopts the same local optimization settings as FedAvg, while only the model

backbone is aggregated. For FedRep (Collins et al., 2021), both the classifier update rounds E_l and the backbone update rounds E_g are set to the number of local epochs E. For Ditto (Li et al., 2021c), we select the regularization coefficient λ from $\{0.1, 0.3, 0.5, 0.7\}$ and set $\lambda = 0.5$. For FedALA (Zhang et al., 2023b), we select the local data proportion s and the number of model layers p involved in ALA from $\{20\%, 40\%, 60\%, 80\%, 100\%\}$ and $\{1, 2, 3, 4, 5\}$, respectively, with $s = 80\%$ and $p = 2$ yielding the best performance. For GPFL (Zhang et al., 2023a), we selected $\lambda = 0.5$ from $\{0.1, 0.3, 0.5, 0.7\}$ for magnitude-level global guidance and $\mu = 0.1$ from $\{0.05, 0.1, 0.3, 0.5\}$ for ℓ_2 regularization. Tab. 7 summarizes the notation used throughout the experiments.

Implementation Details. For all datasets, we use a batch size of 128, except for *Tiny-Imagenet* (Le & Yang, 2015) (256). We adopt SGD with a learning rate of 0.01, set the number of local epochs to 5 and clients to 20, and run 200 communication rounds, after which most methods show little or no improvement. All experiments are conducted on an NVIDIA GeForce RTX 3090 GPU.

C Additional Experiments

C.1 Consensus round

Tab. 8 summarizes the connectivity degrees and corresponding spectral gaps of different network topologies, while Tab. 9 reports the effect of varying the number of consensus rounds on model accuracy. Increasing the number of consensus rounds generally improves performance by enabling information to propagate beyond immediate neighborhoods, thereby facilitating network-wide knowledge sharing and enhancing model generalization. Similarly, more strongly connected topologies tend to achieve higher accuracy, consistent with their larger spectral gaps and faster consensus, the corresponding communication graphs are illustrated in Fig. 6. In particular, a fully connected network enables every client to exchange information directly with all other clients during each consensus round (Jiang et al., 2017; Lalitha et al., 2018), achieving an average accuracy of 55.65%, an improvement of 11.42% over the 44.23% obtained without inter-client communication. These performance gains typically come at the cost of increased communication overhead due to denser connectivity and additional consensus rounds. Therefore, balancing predictive performance and communication efficiency is essential when designing decentralized learning algorithms.

Table 8: Performance across **different topologies** and their corresponding spectral gaps ($1 - \lambda$) and node degrees D. See Sec. C.1 for details.

Network Topology	D	$1 - \lambda$	***Accuracy***
Disconnected	0	0	44.23±0.38
Ring Topology	2	$\approx 16\pi^2/3m^2$	49.21±0.16
Grid Topology	4	$\mathcal{O}\left(1/m \log_2(m)\right)$	51.24±0.11
Exponential	8	$2/(1 + \log_2(m))$	52.33±0.20
Fully-connected	19	1	55.65±0.11

Table 9: Performance with different numbers of **consensus rounds** ($C = 2, 4$) for decentralized methods on *Office10* under ring and exponential topologies. Please see more details in Sec. C.1.

Method	*Ring* & $C = 2$		*Ring* & $C = 4$		*Exp* & $C = 2$		*Exp* & $C = 4$	
	C.Avg	***D.Avg***	***C.Avg***	***D.Avg***	***C.Avg***	***D.Avg***	***C.Avg***	***D.Avg***
D-PSGD (Lian et al., 2017)	73.36±0.12	74.32±0.19	73.57±1.39	74.95±0.68	73.83±0.65	74.55±0.52	75.10±1.32	75.05±0.56
DFedAvgM (Sun et al., 2022)	73.17±1.13	74.53±0.49	73.91±1.39	74.92±0.70	74.39±0.60	75.42±0.71	73.95±1.09	74.95±0.41
DisPFL (Dai et al., 2022)	79.60±0.09	76.60±0.34	81.59±0.94	78.34±0.46	81.59±0.14	78.94±0.71	81.88±0.57	79.31±0.61
DFedSMDC (Shi et al., 2026)	70.78±2.33	70.48±1.36	73.91±1.30	73.08±1.28	68.88±1.48	68.77±1.64	72.66±2.31	72.61±1.56
DFedPGP (Liu et al., 2024)	64.84±0.28	65.46±0.79	66.19±1.59	66.59±0.71	64.95±0.57	65.75±1.11	67.67±0.51	67.19±0.28
GeoSF (Ours)	**82.72**±1.90	**80.81**±0.67	**83.63**±0.74	**81.60**±0.29	**83.60**±1.30	**81.65**±0.62	**83.90**±0.71	**81.81**±0.24

C.2 Comparison of personalized scenarios

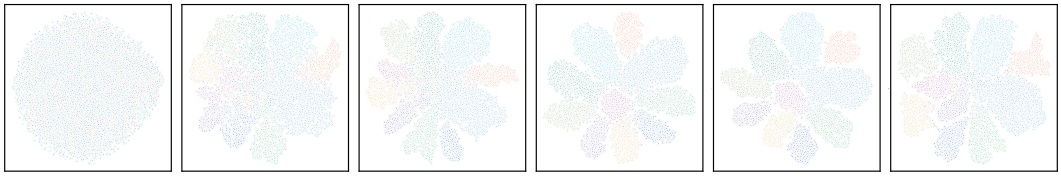
To further evaluate the effectiveness and generalizability of the proposed fast–slow learning framework, we adapt it to the personalized federated learning setting (Zhang et al., 2023b;a). Specifically, we remove the geometry-consistent consensus module and replace decentralized consensus with server-side averaging of the slow variables, yielding a PFL variant termed FedCGF. We compare FedCGF against representative PFL baselines, with the results reported in Tab. 10. The results show that the proposed framework remains effective under both *feature skew* and *label skew* scenarios, indicating that its benefits are not limited to DFL. In particular, on *Office10* (Gong et al., 2012), FedCGF outperforms the strongest baseline, FedAS (Yang et al., 2024), by approximately 2.73%.

Table 10: **Performance comparison** under PFL scenarios, where FedCGF denotes a variant that replaces geometric consistency consensus with server-side averaging. See Sec. C.2 for more details.

PFL Method	*Client-side Performance*				*Office10*		*Cifar100*	
	Amaz	*Calt*	*DSLR*	*Webc*	*C.Avg*	*D.Avg*	*C.Avg*	*D.Avg*
FedPer (Arivazhagan et al., 2019)	84.55±0.85	66.90±0.87	75.95±2.07	81.53±2.61	77.23±1.08	75.84±0.65	46.51±0.37	46.56±0.36
FedRep (Collins et al., 2021)	85.16±0.94	63.21±1.45	80.32±2.60	86.01±2.72	78.93±0.86	75.29±0.48	44.95±0.51	45.00±0.52
FedALA (Zhang et al., 2023b)	87.51±0.74	64.97±0.59	82.48±3.73	81.70±1.99	79.17±1.54	76.53±0.80	46.99±0.76	46.97±0.75
GPFL (Zhang et al., 2023a)	86.58±0.10	64.92±0.38	81.48±3.73	86.56±0.64	79.88±1.08	76.66±0.36	45.72±0.75	45.79±0.72
FedAS (Yang et al., 2024)	85.18±1.12	66.84±0.22	89.87±2.07	92.34±1.15	83.56±0.64	78.18±0.37	49.93±0.30	49.96±0.46
FedCGF (Ours)	89.35±0.90	71.17±0.52	86.08±0.74	87.84±1.21	**83.61**±0.92	**80.91**±0.50	**51.78**±0.22	**51.86**±0.21

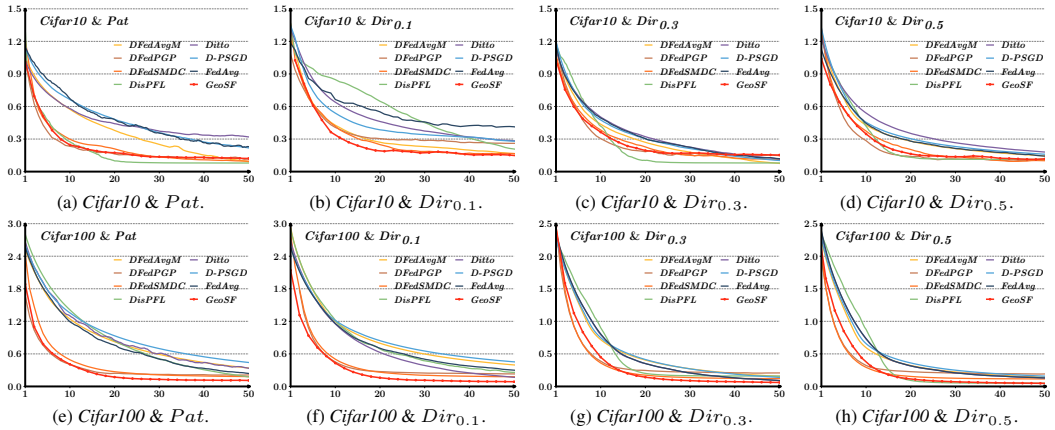
C.3 Training Loss Curves and t-SNE Visualizations

Fig. 7 presents t-SNE visualizations of the feature representations learned on *Digit5* (Zhou et al., 2020) at different training stages. As training progresses, samples from the same class form increasingly compact clusters, while the separation between different classes becomes more pronounced. Compared with D-PSGD (Lian et al., 2017), which incorporates no additional mechanisms, **GeoSF** produces more clearly separated class clusters, providing qualitative evidence of its ability to learn more discriminative feature representations. Furthermore, Fig. 8 presents the training loss curves on *Cifar10* and *Cifar100* under various heterogeneous settings. Across most settings, **GeoSF** exhibits a rapid decrease in training loss followed by stable convergence, demonstrating its favorable optimization behavior under data heterogeneity.



(a) Initial (b) $R = 5$ (c) $R = 10$ (d) $R = 50$ (e) Final (f) Baseline

Figure 7: t-SNE visualization of features extracted by *ResNet18* (He et al., 2016) on the *Digit5* (Zhou et al., 2020) dataset, where R denotes the communication round.



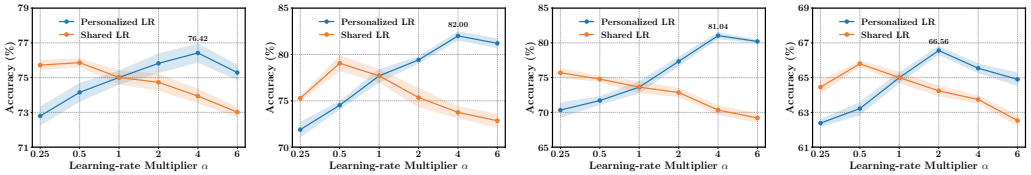
(a) *Cifar10* & Pat. (b) *Cifar10* & $Dir_{0.1}$. (c) *Cifar10* & $Dir_{0.3}$. (d) *Cifar10* & $Dir_{0.5}$.

(e) *Cifar100* & Pat. (f) *Cifar100* & $Dir_{0.1}$. (g) *Cifar100* & $Dir_{0.3}$. (h) *Cifar100* & $Dir_{0.5}$.

Figure 8: **Convergence curves** of different algorithms under varying levels of heterogeneity (Pat and $Dir_{0.5}$). Please refer to Sec. C.3 for a detailed discussion.

C.4 Additional Heterogeneity Experiments

Tab. 11 presents additional results under heterogeneous settings, covering *Flowers102* (Nilsback & Zisserman, 2008) and *label-skew* datasets partitioned according to a Dirichlet distribution with $\beta = 0.3$. **GeoSF** consistently outperforms the competing methods across most evaluated settings, demonstrating its effectiveness under diverse forms of data heterogeneity. Notably, on the 100-class



(a) Accuracy on *CNN* (b) Accuracy on *ResNet18* (c) Accuracy on *ResNet50* (d) Accuracy on *ViT-B/16*

Figure 9: **Performance sensitivity** to the learning rates of shared and personalized parameters.

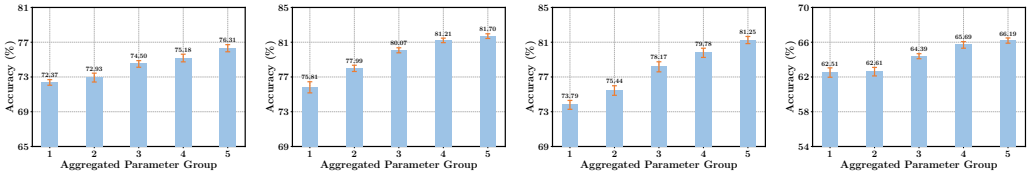
Cifar100 (Krizhevsky et al., 2009) dataset, **GeoSF** achieves an accuracy of 47.46%, exceeding the strongest baseline by 6.86%. Overall, these results further substantiate the robustness of **GeoSF** across datasets with varying scales, class cardinalities, and levels of heterogeneity.

Table 11: **Additional experiments** on *Flowers102* and $Dir_{0.3}$ scenarios. See Sec. C.4 for details.

Method	$Dir_{0.3}$				*Flowers102*			
	AGNews	*Cifar10*	*Cifar100*	*Tiny*	*Pat*	$Dir_{0.1}$	$Dir_{0.3}$	$Dir_{0.5}$
SOLO	74.12±0.88	73.45±0.27	31.83±0.18	17.90±0.14	51.15±0.27	51.06±0.27	34.44±0.86	26.96±0.22
FedAvg (McMahan et al., 2017)	74.78±0.15	78.64±0.10	37.59±0.29	23.14±0.28	57.79±0.59	58.07±0.45	45.49±0.24	38.70±0.81
Ditto (Li et al., 2021c)	74.66±0.08	76.74±0.20	38.69±0.39	21.02±0.14	56.33±0.52	56.74±1.03	42.49±1.28	36.25±0.46
D-PSGD (Lian et al., 2017)	74.57±0.09	78.38±0.10	40.13±0.12	23.50±0.06	57.63±0.33	**58.17**±0.95	45.10±0.44	39.67±0.14
DFedAvgM (Sun et al., 2022)	74.72±0.10	78.64±0.25	40.37±0.08	23.44±0.25	58.33±0.70	57.94±0.51	44.76±0.30	38.94±0.14
DisPFL (Dai et al., 2022)	74.59±0.09	78.46±0.05	37.38±0.68	17.92±0.22	57.31±0.52	52.85±0.25	36.48±0.28	30.85±0.68
DFedSMDC (Shi et al., 2026)	77.56±1.46	77.04±0.13	40.60±0.37	**24.61**±0.55	53.07±1.32	55.26±0.72	41.21±0.77	35.05±1.19
DFedPGP (Liu et al., 2024)	76.88±1.07	75.53±0.06	35.03±0.28	22.17±0.49	48.99±0.72	52.78±1.08	39.15±0.30	32.85±0.08
GeoSF (Ours)	**90.68**±0.94	**82.91**±0.09	**47.46**±0.28	24.22±0.12	**59.91**±0.61	57.29±0.56	**46.87**±0.84	**40.95**±0.68

C.5 Motivations behind different models

Fig. 9 illustrates the relationship between training accuracy and learning-rate configurations across different model architectures, including *CNN* (Zhang et al., 2023a), *ResNet18*, *ResNet50* (He et al., 2016), and *ViT* (Dosovitskiy, 2020). Across all models, assigning a lower learning rate to stable parameters consistently yields stronger performance. This observation supports our two-timescale learning strategy, in which stable parameters are updated conservatively to preserve shared knowledge, whereas personalized parameters adopt a higher learning rate to rapidly adapt to local data distributions. Furthermore, Fig. 10 examines the effect of aggregating parameter groups at different locations within each model. Aggregating the stable parameters generally achieves superior performance across the evaluated architectures, supporting our hypothesis that these parameters are particularly well suited for cross-client aggregation.

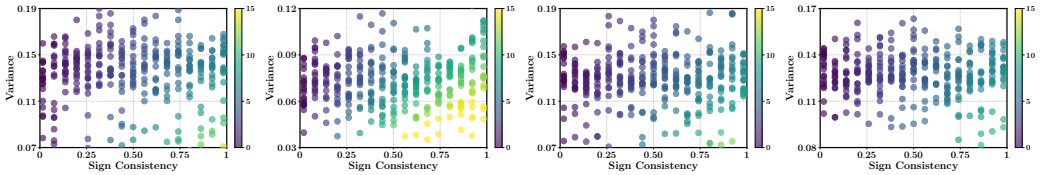


(a) Accuracy on *CNN* (b) Accuracy on *ResNet18* (c) Accuracy on *ResNet50* (d) Accuracy on *ViT-B/16*

Figure 10: **Performance sensitivity** to the placement of shared parameters. See details in Sec. C.5

C.6 Motivations behind different models

We assess the stability of individual parameters across clients in four model architectures using sign consistency and variance, from which an overall stability score is derived, as shown in Fig. 11. Sign consistency values span the full interval $[0, 1]$, revealing substantial variation across parameter coordinates. This observation indicates that individual parameters respond differently to heterogeneous client data and consequently exhibit distinct levels of stability. The consistent pattern observed across all four architectures suggests that parameter-level heterogeneity is a general characteristic of deep models in federated learning rather than a phenomenon specific to a particular architecture.



(a) Stability of *CNN* (b) Stability of *ResNet18* (c) Stability of *ResNet50* (d) Stability of *ViT-B/16*

Figure 11: **Empirical analysis** of cross-client parameter stability, measured by *sign consistency* and *variance*, across different models. Please see more details in Sec. C.6

D Convergence Analysis of GeoSF

D.1 Problem Formulation

The local model of client m is decomposed into a slow parameter block and a fast parameter block. Let $\Theta^f = (\theta_1^f, \dots, \theta_M^f), \Theta^s = (\theta_1^s, \dots, \theta_M^s)$. The optimization problem is:

$$\min_{\Theta^s, \Theta^f} F(\Theta^s, \Theta^f) = \frac{1}{M} \sum_{m=1}^{M} F_m(\theta_m^s, \theta_m^f). \tag{12}$$

GeoSF simultaneously updates the slow and fast parameters at each local iteration as follows:

$$\theta_{m,k+1}^{s,t} = \theta_{m,k}^{s,t} - \eta_s g_{m,k}^{s,t}, \quad \theta_{m,k+1}^{f,t} = \theta_{m,k}^{f,t} - \eta_f g_{m,k}^{f,t}, \quad k = 0, \dots, K-1, \tag{13}$$

where both stochastic gradients are evaluated at the same local iterate.

We define the stationarity measures for the slow/shared and fast/personalized blocks as $\Delta^{s,t} = \|\frac{1}{M}\sum_{m=1}^{M} \nabla_s F_m(\bar{\theta}_m^{s,t}, \theta_m^{f,t})\|^2$ and $\Delta^{f,t} = \frac{1}{M}\sum_{m=1}^{M} \|\nabla_f F_m(\bar{\theta}_m^{s,t}, \theta_m^{f,t})\|^2$, respectively.

The accumulated local drifts are defined as $D_s^t = \frac{1}{M}\sum_{m=1}^{M}\sum_{k=0}^{K-1} \mathbb{E}\|\theta_{m,k}^{s,t} - \theta_m^{s,t}\|^2$ and $D_f^t = \frac{1}{M}\sum_{m=1}^{M}\sum_{k=0}^{K-1} \mathbb{E}\|\theta_{m,k}^{f,t} - \theta_m^{f,t}\|^2$. The slow-block consensus error is $C_s^t = \frac{1}{M}\sum_{m=1}^{M} \mathbb{E}\|\theta_m^{s,t} - \bar{\theta}_m^{s,t}\|^2$. For convenience, we also set $\hat{L}_s = L_s(1+\chi^2)$ and $\hat{L}_f = L_f(1+\chi^2)$.

D.2 Assumptions

Assumption 4 (Block Smoothness)**.** *For each client m, F_m is continuously differentiable. There exist constants $L_s, L_f, L_{sf}, L_{fs} > 0$ such that $\nabla_s F_m(\theta_m^{s,t}, \theta_m^{f,t})$ is L_s-Lipschitz with respect to $\theta_m^{s,t}$ and L_{sf}-Lipschitz with respect to $\theta_m^{f,t}$, while $\nabla_f F_m(\theta_m^{s,t}, \theta_m^{f,t})$ is L_f-Lipschitz with respect to $\theta_m^{f,t}$ and L_{fs}-Lipschitz with respect to $\theta_m^{s,t}$. We further define $\chi := \max\{L_{sf}, L_{fs}\}/\sqrt{L_s L_f}$.*

Assumption 5 (Bounded Variance)**.** *The stochastic gradients are unbiased. Moreover, their variances are uniformly bounded as $\mathbb{E}\|g_{m,k}^{s,t} - \mathbb{E}[g_{m,k}^{s,t}]\|^2 \le \sigma_s^2$ and $\mathbb{E}\|g_{m,k}^{f,t} - \mathbb{E}[g_{m,k}^{f,t}]\|^2 \le \sigma_f^2$.*

Assumption 6 (Slow Gradient Diversity)**.** *The heterogeneity of slow gradients is bounded. There exists $\sigma_g \ge 0$ such that $\frac{1}{M}\sum_{m=1}^{M} \|\nabla_s F_m(\theta_m^{s,t}, \theta_m^{f,t}) - \frac{1}{M}\sum_{m=1}^{M} \nabla_s F_m(\theta_m^{s,t}, \theta_m^{f,t})\|^2 \le \sigma_g^2$.*

D.3 Preliminary Lemmas

Lemma 1. *Under Assumption 4, each objective F_m is jointly smooth with respect to the slow and fast parameters. Specifically, for any client m and any two points θ_m^t and $\tilde{\theta}_m^t$, we have:*

$$F_m(\tilde{\theta}_m^{s,t}, \tilde{\theta}_m^{f,t}) - F_m(\theta_m^{s,t}, \theta_m^{f,t}) \le \langle \nabla_s F_m(\theta_m^{s,t}, \theta_m^{f,t}), \tilde{\theta}_m^{s,t} - \theta_m^{s,t} \rangle + \langle \nabla_f F_m(\theta_m^{s,t}, \theta_m^{f,t}), \tilde{\theta}_m^{f,t} - \theta_m^{f,t} \rangle + \frac{\hat{L}_s}{2}\|\tilde{\theta}_m^{s,t} - \theta_m^{s,t}\|^2 + \frac{\hat{L}_f}{2}\|\tilde{\theta}_m^{f,t} - \theta_m^{f,t}\|^2.$$

Proof. The result follows by applying Lemma 20 of (Pillutla et al., 2022) to F_m. □

Lemma 2. *Define the slow-block stochastic-gradient noise at client m and local step k as $\xi_{m,k}^{s,t} = g_{m,k}^{s,t} - \nabla_s F_m(\theta_{m,k}^{s,t}, \theta_{m,k}^{f,t})$. Under Assumption 5, its average over all clients and local steps satisfies:*

$$\mathbb{E}\Big\| \frac{1}{MK} \sum_{m=1}^{M} \sum_{k=0}^{K-1} \xi_{m,k}^{s,t} \Big\|^2 \le \frac{\sigma_s^2}{MK}.$$

Proof. By the unbiasedness and conditional independence of stochastic gradient, we have:

$$\mathbb{E}\Big\| \frac{1}{MK} \sum_{m=1}^{M} \sum_{k=0}^{K-1} \xi_{m,k}^{s,t} \Big\|^2 = \frac{1}{M^2K^2} \sum_{m=1}^{M} \sum_{k=0}^{K-1} \mathbb{E}\|\xi_{m,k}^{s,t}\|^2 \le \frac{\sigma_s^2}{MK}.$$

□

Lemma 3. *Suppose Assumptions 4–6 hold. Let $\mathcal{A}_s, \mathcal{A}_f > 0$. If $K \ge 2$ and the learning rates satisfy $3K^2(\mathcal{A}_s\eta_s^2 L_s^2 + \mathcal{A}_f\eta_f^2 L_{fs}^2) \le \mathcal{A}_s, 3K^2(\mathcal{A}_s\eta_s^2 L_{sf}^2 + \mathcal{A}_f\eta_f^2 L_f^2) \le \mathcal{A}_f, C = 4K^2(K-1)$ then:*

$$\begin{aligned}\mathcal{A}_s D_s^t + \mathcal{A}_f D_f^t \le & C\{\mathcal{A}_s\eta_s^2[\sigma_s^2 + 6(\mathbb{E}[\Delta^{s,t}] + \sigma_g^2 + L_s^2 C_s^t)] \\ & + \mathcal{A}_f\eta_f^2[\sigma_f^2 + 6(\mathbb{E}[\Delta^{f,t}] + L_{fs}^2 C_s^t)]\}.\end{aligned}$$

For $K = 1$, both local-drift terms are zero, and the result holds trivially.

Proof. By adding and subtracting the gradients at the initial point, and then applying bounded stochastic-gradient variance and block smoothness, we obtain:

$$\begin{aligned}\mathbb{E}\|g_{m,k}^{s,t}\|^2 \le & \sigma_s^2 + 3\|\nabla_s F_m(\theta_m^{s,t}, \theta_m^{f,t})\|^2 + 3L_s^2\|\theta_{m,k}^{s,t} - \theta_m^{s,t}\|^2 + 3L_{sf}^2\|\theta_{m,k}^{f,t} - \theta_m^{f,t}\|^2, \\ \mathbb{E}\|g_{m,k}^{f,t}\|^2 \le & \sigma_f^2 + 3\|\nabla_f F_m(\theta_m^{s,t}, \theta_m^{f,t})\|^2 + 3L_{fs}^2\|\theta_{m,k}^{s,t} - \theta_m^{s,t}\|^2 + 3L_f^2\|\theta_{m,k}^{f,t} - \theta_m^{f,t}\|^2.\end{aligned} \tag{14}$$

Under the learning-rate conditions $3K^2(\mathcal{A}_s\eta_s^2 L_s^2 + \mathcal{A}_f\eta_f^2 L_{fs}^2) \le \mathcal{A}_s$ and $3K^2(\mathcal{A}_s\eta_s^2 L_{sf}^2 + \mathcal{A}_f\eta_f^2 L_f^2) \le \mathcal{A}_f$, the weighted parameter shift satisfies:

$$\begin{aligned}& \mathbb{E}\left[\mathcal{A}_s\|\theta_{m,k+1}^{s,t} - \theta_m^{s,t}\|^2 + \mathcal{A}_f\|\theta_{m,k+1}^{f,t} - \theta_m^{f,t}\|^2\right] \\ & \le \left(1 + \frac{2}{K-1}\right)\left[\mathcal{A}_s\|\theta_{m,k}^{s,t} - \theta_m^{s,t}\|^2 + \mathcal{A}_f\|\theta_{m,k}^{f,t} - \theta_m^{f,t}\|^2\right] \\ & \quad + 3K\mathcal{A}_s\eta_s^2\|\nabla_s F_m(\theta_m^{s,t}, \theta_m^{f,t})\|^2 + 3K\mathcal{A}_f\eta_f^2\|\nabla_f F_m(\theta_m^{s,t}, \theta_m^{f,t})\|^2 \\ & \quad + K(\mathcal{A}_s\eta_s^2\sigma_s^2 + \mathcal{A}_f\eta_f^2\sigma_f^2).\end{aligned} \tag{15}$$

Since the initial local drifts are zero, unrolling the resulting recursion, summing over k, averaging over all clients, and setting $C = 4K^2(K-1)$, we obtain:

$$\begin{aligned}\mathcal{A}_s D_s^t + \mathcal{A}_f D_f^t \le & C\left(\mathcal{A}_s\eta_s^2\sigma_s^2 + \mathcal{A}_f\eta_f^2\sigma_f^2\right) + \frac{3C\mathcal{A}_s\eta_s^2}{M}\sum_{m=1}^{M}\mathbb{E}\|\nabla_s F_m(\theta_m^{s,t}, \theta_m^{f,t})\|^2 \\ & + \frac{3C\mathcal{A}_f\eta_f^2}{M}\sum_{m=1}^{M}\mathbb{E}\|\nabla_f F_m(\theta_m^{s,t}, \theta_m^{f,t})\|^2 \\ \le & C\{\mathcal{A}_s\eta_s^2[\sigma_s^2 + 6(\mathbb{E}[\Delta^{s,t}] + \sigma_g^2 + L_s^2 C_s^t)] + \mathcal{A}_f\eta_f^2[\sigma_f^2 + 6(\mathbb{E}[\Delta^{f,t}] + L_{fs}^2 C_s^t)]\}.\end{aligned} \tag{16}$$

The final inequality follows from the gradient-diversity and block-wise smoothness assumptions. □

Lemma 4. *Suppose Assumptions 4–6 and the conditions of Lemma 3 hold. Assume that the shared variables are identically initialized across all clients, W is doubly stochastic, and the learning rate satisfies Eq. (20). Let $P = \frac{1}{M}\mathbf{1}\mathbf{1}^\top$ and $\lambda = \|W - P\|_2 < 1$. For notational simplicity, define $\mathrm{I}_t = \sigma_s^2 + 6\sigma_g^2 + 6\mathbb{E}[\Delta^{s,t}]$ and $\mathrm{II}_t = \sigma_f^2 + 6\mathbb{E}[\Delta^{f,t}]$. Then, we have:*

$$\sum_{t=0}^{T-1} C_s^t \le \frac{2\eta_s^2 K^2}{(1-\lambda)^2}\sum_{t=0}^{T-1}\Big\{\mathrm{I}_t + 12K(K-1)(L_s^2 + L_{sf}^2)\left(\eta_s^2\mathrm{I}_t + \eta_f^2\mathrm{II}_t\right)\Big\}.$$

Proof. Based on Lemma 5 in (Shi et al., 2026) and the double stochasticity of W, the consensus error of the shared parameters can be expressed as:

$$\Theta^{s,t} - P\Theta^{s,t} = -\eta_s\sum_{j=0}^{t-1}(W^{t-j} - P)\mathcal{G}_s^j. \tag{17}$$

Applying the weighted Cauchy–Schwarz inequality and summing over t, we obtain:

$$\sum_{t=0}^{T-1} C_s^t \le \frac{\eta_s^2}{M(1-\lambda)} \sum_{t=0}^{T-1}\sum_{j=0}^{t-1} \lambda^{t-j}\mathbb{E}\|\mathcal{G}_s^j\|_F^2 \le \frac{\eta_s^2}{M(1-\lambda)^2} \sum_{t=0}^{T-1} \mathbb{E}\|\mathcal{G}_s^t\|_F^2. \tag{18}$$

Combining Assumptions 4 and 6, Lemma 3 with $\mathcal{A}_s = \mathcal{A}_f = 1$, we obtain:

$$\begin{aligned}\frac{1}{M}\mathbb{E}\|\mathcal{G}_s^t\|_F^2 \le& K^2\sigma_s^2 + \frac{3K^2}{M}\sum_{m=1}^{M}\mathbb{E}\|\nabla_s F_m(\theta_m^{s,t},\theta_m^{f,t})\|^2 + 3KL_s^2 D_s^t + 3KL_{sf}^2 D_f^t \\ \le& K^2\Big\{12K(K-1)(L_s^2+L_{sf}^2)\left[\eta_s^2\left(\mathrm{I}_t + 6L_s^2 C_s^t\right) + \eta_f^2\left(\mathrm{II}_t + 6L_{fs}^2 C_s^t\right)\right] \\ &+ \mathrm{I}_t + 6L_s^2 C_s^t\Big\},\end{aligned} \tag{19}$$

under the stepsize condition:

$$\frac{\eta_s^2 K^2}{(1-\lambda)^2}[6L_s^2 + 72K(K-1)(L_s^2+L_{sf}^2)(\eta_s^2 L_s^2 + \eta_f^2 L_{fs}^2)] \le 1/2. \tag{20}$$

Substituting Eq. (19) into Eq. (18) and invoking Eq. (20) completes the proof. □

D.4 One-Round Joint Descent

The average slow and personalized fast parameters evolve as $\bar{\theta}_m^{s,t+1} = \bar{\theta}_m^{s,t} - \frac{\eta_s}{M}\sum_{m=1}^{M}\sum_{k=0}^{K-1} g_{m,k}^{s,t}$ and $\theta_m^{f,t+1} = \theta_m^{f,t} - \eta_f \sum_{k=0}^{K-1} g_{m,k}^{f,t}$, respectively. Applying Lemma 1, we obtain:

$$\begin{aligned}&F(\bar{\Theta}^{s,t+1},\Theta^{f,t+1}) - F(\bar{\Theta}^{s,t},\Theta^{f,t}) \\ &\le \underbrace{\frac{1}{M}\sum_{m=1}^{M}\left\langle \nabla_s F_m(\bar{\theta}_m^{s,t},\theta_m^{f,t}), \bar{\theta}_m^{s,t+1} - \bar{\theta}_m^{s,t}\right\rangle}_{T_{1,s}} + \underbrace{\frac{1}{M}\sum_{m=1}^{M}\left\langle \nabla_f F_m(\bar{\theta}_m^{s,t},\theta_m^{f,t}), \theta_m^{f,t+1} - \theta_m^{f,t}\right\rangle}_{T_{1,f}} \\ &+ \underbrace{\frac{\hat{L}_s}{2M}\sum_{m=1}^{M}\|\bar{\theta}_m^{s,t+1} - \bar{\theta}_m^{s,t}\|^2}_{T_{2,s}} + \underbrace{\frac{\hat{L}_f}{2M}\sum_{m=1}^{M}\|\theta_m^{f,t+1} - \theta_m^{f,t}\|^2}_{T_{2,f}}.\end{aligned} \tag{21}$$

We next bound these four terms separately.

D.4.1 Bounding $T_{1,s}$

To bound $T_{1,s}$, define $G^{s,t} = \nabla_s F(\bar{\Theta}^{s,t},\Theta^{f,t})$ and $G_m^{s,t} = \nabla_s F_m(\bar{\theta}_m^{s,t},\theta_m^{f,t})$. Substituting the averaged slow-parameter update into the expression for $T_{1,s}$, we obtain:

$$T_{1,s} = -\frac{\eta_s}{M}\sum_{m=1}^{M}\sum_{k=0}^{K-1}\langle G^{s,t}, g_{m,k}^{s,t}\rangle. \tag{22}$$

Taking the conditional expectation and applying Assumption 5, we obtain:

$$\begin{aligned}\mathbb{E}[T_{1,s}] =& -\frac{\eta_s}{M}\sum_{m=1}^{M}\sum_{k=0}^{K-1}\mathbb{E}\Big[\big\langle G^{s,t}, \nabla_s F_m(\theta_{m,k}^{s,t},\theta_{m,k}^{f,t})\big\rangle\Big] \\ =& -\frac{\eta_s}{M}\sum_{m=1}^{M}\sum_{k=0}^{K-1}\mathbb{E}\Big[\big\langle G^{s,t}, G_m^{s,t}\big\rangle\Big] - \frac{\eta_s}{M}\sum_{m=1}^{M}\sum_{k=0}^{K-1}\mathbb{E}\Big[\big\langle G^{s,t}, \nabla_s F_j(\theta_{m,k}^{s,t},\theta_{m,k}^{f,t}) - G_m^{s,t}\big\rangle\Big] \\ =& -\eta_s K\mathbb{E}\|G^{s,t}\|^2 - \frac{\eta_s}{M}\sum_{j=1}^{M}\sum_{k=0}^{K-1}\mathbb{E}\Big[\big\langle G^{s,t}, e_{m,k}^{s,t}\big\rangle\Big],\end{aligned} \tag{23}$$

where $e_{m,k}^{s,t} = \nabla_s F_j(\theta_{m,k}^{s,t}, \theta_{m,k}^{f,t}) - \nabla_s F_m(\bar{\theta}_m^{s,t}, \theta_m^{f,t})$. Then, applying Assumption 4, we obtain:

$$\begin{aligned}
\mathbb{E}[T_{1,s}] \leq & -\eta_s K \mathbb{E}\|G^{s,t}\|^2 + \frac{\eta_s}{M}\sum_{m=1}^{M}\sum_{k=0}^{K-1}\mathbb{E}\left[\left|\langle G^{s,t}, e_{m,k}^{s,t}\rangle\right|\right] \\
\leq & -\eta_s K \mathbb{E}\|G^{s,t}\|^2 + \frac{\eta_s}{2M}\sum_{m=1}^{M}\sum_{k=0}^{K-1}\mathbb{E}\|G^{s,t}\|^2 + \frac{\eta_s}{2M}\sum_{m=1}^{M}\sum_{k=0}^{K-1}\mathbb{E}\|e_{m,k}^{s,t}\|^2 \\
\leq & -\frac{\eta_s K}{2}\mathbb{E}\|G^{s,t}\|^2 + \frac{\eta_s}{M}\sum_{m=1}^{M}\sum_{k=0}^{K-1}\mathbb{E}[L_s^2\|\theta_{m,k}^{s,t} - \bar{\theta}_m^{s,t}\|^2 + L_{sf}^2\|\theta_{m,k}^{f,t} - \theta_m^{f,t}\|^2] \\
\leq & -\frac{\eta_s K}{2}\mathbb{E}[\Delta^{s,t}] + 2\eta_s L_s^2 D_s^t + \eta_s L_{sf}^2 D_f^t + 2\eta_s K L_s^2 C_s^t.
\end{aligned} \tag{24}$$

D.4.2 Bounding $T_{1,f}$

Let $G_m^{f,t} = \nabla_f F_m(\bar{\theta}_m^{s,t}, \theta_m^{f,t})$. Substituting the update of the fast parameters into $T_{1,f}$ and using the unbiasedness of stochastic gradients yield the following decomposition:

$$\begin{aligned}
T_{1,f} = & -\frac{\eta_f}{M}\sum_{m=1}^{M}\sum_{k=0}^{K-1}\langle G_m^{f,t}, \nabla_f F_m(\theta_{m,k}^{s,t}, \theta_{m,k}^{f,t})\rangle \\
= & -\frac{\eta_f}{M}\sum_{m=1}^{M}\sum_{k=0}^{K-1}\|G_m^{f,t}\|^2 - \frac{\eta_f}{M}\sum_{m=1}^{M}\sum_{k=0}^{K-1}\langle G_m^{f,t}, \nabla_f F_m(\theta_{m,k}^{s,t}, \theta_{m,k}^{f,t}) - G_m^{f,t}\rangle \\
= & -\eta_f K \Delta^{f,t} - \frac{\eta_f}{M}\sum_{m=1}^{M}\sum_{k=0}^{K-1}\langle G_m^{f,t}, e_{m,k}^{f,t}\rangle,
\end{aligned} \tag{25}$$

where $e_{m,k}^{f,t} = \nabla_f F_m(\theta_{m,k}^{s,t}, \theta_{m,k}^{f,t}) - \nabla_f F_m(\bar{\theta}_m^{s,t}, \theta_m^{f,t})$. Then, applying Assumption 4 and the definition of C_s^t, we obtain:

$$\begin{aligned}
\mathbb{E}[T_{1,f}] \leq & -\eta_f K \mathbb{E}[\Delta^{f,t}] + \frac{\eta_f}{2M}\sum_{m=1}^{M}\sum_{k=0}^{K-1}\mathbb{E}\|G_m^{f,t}\|^2 + \frac{\eta_f}{2M}\sum_{m=1}^{M}\sum_{k=0}^{K-1}\mathbb{E}\|e_{m,k}^{f,t}\|^2 \\
\leq & -\frac{\eta_f K}{2}\mathbb{E}[\Delta^{f,t}] + \frac{\eta_f}{M}\sum_{m=1}^{M}\sum_{k=0}^{K-1}\mathbb{E}\left[L_f^2\|\theta_{m,k}^{f,t} - \theta_m^{f,t}\|^2 + L_{fs}^2\|\theta_{m,k}^{s,t} - \bar{\theta}_m^{s,t}\|^2\right] \\
\leq & -\frac{\eta_f K}{2}\mathbb{E}[\Delta^{f,t}] + \eta_f L_f^2 D_f^t + 2\eta_f L_{fs}^2 D_s^t + 2\eta_f K L_{fs}^2 C_s^t.
\end{aligned} \tag{26}$$

This establishes the desired upper bound on $T_{1,f}$.

D.4.3 Bounding $T_{2,s}$

According to the evolution of the averaged slow parameters, $T_{2,s}$ can be expressed as:

$$T_{2,s} = \frac{\hat{L}_s \eta_s^2}{2M}\sum_{m=1}^{M}\mathbb{E}\Big\|\frac{1}{M}\sum_{m=1}^{M}\sum_{k=0}^{K-1} g_{m,k}^{s,t}\Big\|^2. \tag{27}$$

Decomposing the gradient as $g_{m,k}^{s,t} = G_m^{s,t} + e_{m,k}^{s,t} + \xi_{m,k}^{s,t}$, and applying Assumption 6 and Lemma 2, we obtain:

$$\begin{aligned}
\mathbb{E}[T_{2,s}] = & \frac{\hat{L}_s \eta_s^2}{2M}\sum_{m=1}^{M}\mathbb{E}\Big\|\frac{1}{M}\sum_{m=1}^{M}\sum_{k=0}^{K-1} g_{m,k}^{s,t}\Big\|^2 \\
\leq & \frac{3\hat{L}_s \eta_s^2}{2}\Big(K^2\mathbb{E}\|G^{s,t}\|^2 + \mathbb{E}\Big\|\frac{1}{M}\sum_{m=1}^{M}\sum_{k=0}^{K-1} e_{m,k}^{s,t}\Big\|^2 + \mathbb{E}\Big\|\frac{1}{M}\sum_{m=1}^{M}\sum_{k=0}^{K-1}\xi_{m,k}^{s,t}\Big\|^2\Big) \\
\leq & \frac{3\hat{L}_s \eta_s^2}{2}\Big(K^2\mathbb{E}[\Delta^{s,t}] + 4KL_s^2 D_s^t + 2KL_{sf}^2 D_f^t + 4K^2 L_s^2 C_s^t + \frac{K\sigma_s^2}{M}\Big)
\end{aligned} \tag{28}$$

$$= \frac{3\hat{L}_s\eta_s^2K^2}{2}\mathbb{E}[\Delta^{s,t}] + \frac{3\hat{L}_s\eta_s^2K}{2M}\sigma_s^2 + 6\hat{L}_s\eta_s^2KL_s^2D_s^t + 3\hat{L}_s\eta_s^2KL_{sf}^2D_f^t + 6\hat{L}_s\eta_s^2K^2L_s^2C_s^t.$$

This establishes the desired upper bound on $T_{2,s}$.

D.4.4 Bounding $T_{2,f}$

According to the local update process of the fast parameters, we have:

$$T_{2,f} = \frac{\hat{L}_f\eta_f^2}{2M}\sum_{m=1}^{M}\mathbb{E}\Big\|\sum_{k=0}^{K-1}g_{m,k}^{f,t}\Big\|^2. \tag{29}$$

Decomposing the gradient as $g_{m,k}^{f,t} = G_m^{f,t} + e_{m,k}^{f,t} + \xi_{m,k}^{f,t}$, and applying the squared triangle inequality, Assumptions 4 and 5, and Lemma 2, we obtain:

$$\begin{aligned}
\mathbb{E}[T_{2,f}] =& \frac{\hat{L}_f\eta_f^2}{2M}\sum_{m=1}^{M}\mathbb{E}\Big\|\sum_{k=0}^{K-1}g_{m,k}^{f,t}\Big\|^2 \\
\le& \frac{3\hat{L}_f\eta_f^2}{2M}\sum_{m=1}^{M}\mathbb{E}\Big[K^2\|G_m^{f,t}\|^2 + K\sum_{k=0}^{K-1}\|e_{m,k}^{f,t}\|^2 + \Big\|\sum_{k=0}^{K-1}\xi_{m,k}^{f,t}\Big\|^2\Big] \\
\le& \frac{3\hat{L}_f\eta_f^2K^2}{2}\mathbb{E}[\Delta^{f,t}] + \frac{3\hat{L}_f\eta_f^2K}{2M}\sum_{m=1}^{M}\sum_{k=0}^{K-1}\mathbb{E}\|e_{m,k}^{f,t}\|^2 + \frac{3\hat{L}_f\eta_f^2K}{2}\sigma_f^2 \\
\le& \frac{3\hat{L}_f\eta_f^2K^2}{2}\mathbb{E}[\Delta^{f,t}] + \frac{3\hat{L}_f\eta_f^2K}{2}\left(2L_f^2D_f^t + 4L_{fs}^2D_s^t + 4KL_{fs}^2C_s^t\right) + \frac{3\hat{L}_f\eta_f^2K}{2}\sigma_f^2 \\
=& \frac{3\hat{L}_f\eta_f^2K^2}{2}\mathbb{E}[\Delta^{f,t}] + \frac{3\hat{L}_f\eta_f^2K}{2}\sigma_f^2 + 3\hat{L}_f\eta_f^2KL_f^2D_f^t + 6\hat{L}_f\eta_f^2KL_{fs}^2D_s^t + 6\hat{L}_f\eta_f^2K^2L_{fs}^2C_s^t.
\end{aligned} \tag{30}$$

D.4.5 One-Round Descent

Substituting Eqs. (24), (26), (28), and (30) into Eq. (21), we obtain:

$$\begin{aligned}
&\mathbb{E}\left[F(\bar{\Theta}^{s,t+1},\Theta^{f,t+1}) - F(\bar{\Theta}^{s,t},\Theta^{f,t})\right] \\
&\le -\Big(\frac{\eta_sK}{2} - \frac{3\hat{L}_s\eta_s^2K^2}{2}\Big)\mathbb{E}[\Delta^{s,t}] - \Big(\frac{\eta_fK}{2} - \frac{3\hat{L}_f\eta_f^2K^2}{2}\Big)\mathbb{E}[\Delta^{f,t}] \\
&\quad + \frac{3\hat{L}_s\eta_s^2K}{2M}\sigma_s^2 + \frac{3\hat{L}_f\eta_f^2K}{2}\sigma_f^2 + \mathcal{A}_sD_s^t + \mathcal{A}_fD_f^t + \mathcal{B}C_s^t,
\end{aligned} \tag{31}$$

where $\mathcal{A}_s = 2\eta_sL_s^2 + 2\eta_fL_{fs}^2 + 6\hat{L}_s\eta_s^2KL_s^2 + 6\hat{L}_f\eta_f^2KL_{fs}^2$, $\mathcal{A}_f = \eta_sL_{sf}^2 + \eta_fL_f^2 + 3\hat{L}_s\eta_s^2KL_{sf}^2 + 3\hat{L}_f\eta_f^2KL_f^2$, and $\mathcal{B} = 2\eta_sKL_s^2 + 2\eta_fKL_{fs}^2 + 6\hat{L}_s\eta_s^2K^2L_s^2 + 6\hat{L}_f\eta_f^2K^2L_{fs}^2$.

Then, applying Lemma 3 and collecting the gradient and variance terms in Eq. (31), we obtain:

$$\begin{aligned}
&\mathbb{E}\left[F(\bar{\Theta}^{s,t+1},\Theta^{f,t+1}) - F(\bar{\Theta}^{s,t},\Theta^{f,t})\right] \\
&\le -\Big(\frac{\eta_sK}{2} - \frac{3\hat{L}_s\eta_s^2K^2}{2} - 6C\mathcal{A}_s\eta_s^2\Big)\mathbb{E}[\Delta^{s,t}] - \Big(\frac{\eta_fK}{2} - \frac{3\hat{L}_f\eta_f^2K^2}{2} - 6C\mathcal{A}_f\eta_f^2\Big)\mathbb{E}[\Delta^{f,t}] \\
&\quad + \Big(\frac{3\hat{L}_sK}{2M} + C\mathcal{A}_s\Big)\eta_s^2\sigma_s^2 + 6C\mathcal{A}_s\eta_s^2\sigma_g^2 + \Big(\frac{3\hat{L}_fK}{2} + C\mathcal{A}_f\Big)\eta_f^2\sigma_f^2 + \tilde{\mathcal{B}}C_s^t,
\end{aligned} \tag{32}$$

where $C = 4K^2(K-1)$ and $\tilde{\mathcal{B}} = \mathcal{B} + 6C(\mathcal{A}_s\eta_s^2L_s^2 + \mathcal{A}_f\eta_f^2L_{fs}^2)$.

D.5 Main Convergence Theorem

Theorem 2. *Suppose Assumptions 4–6 hold, and let $F^\star$ denote the minimum value of F. Let $\mathcal{W}$ be the set of consensus weight matrices considered during training, with $\lambda = \max_{W\in\mathcal{W}}\|W - P\|_2 < 1$. Let $\eta_s = a_s/(L_sK\sqrt{T})$ and $\eta_f = a_f/(L_fK\sqrt{T})$, where $a_s, a_f > 0$ satisfy the*

required stepsize and absorption conditions. If $a_s/L_s \ll a_f/L_f$, then $\eta_s \ll \eta_f$. Define $\Delta^{s,t} = \|\frac{1}{M}\sum_{m=1}^{M} \nabla_s F_m(\bar{\theta}_m^{s,t}, \theta_m^{f,t})\|^2$ and $\Delta^{f,t} = \frac{1}{M}\sum_{m=1}^{M} \|\nabla_f F_m(\bar{\theta}_m^{s,t}, \theta_m^{f,t})\|^2$. Then, we have:

$$\frac{1}{T}\sum_{t=0}^{T-1}\Big(\frac{a_s}{L_s}\mathbb{E}[\Delta^{s,t}] + \frac{a_f}{L_f}\mathbb{E}[\Delta^{f,t}]\Big) \le \mathcal{O}\Big(\frac{\Delta}{\sqrt{T}} + \frac{\Gamma_1}{\sqrt{T}} + \frac{\Gamma_2}{T} + \frac{\Gamma_3}{(1-\lambda)^2 T}\Big),$$

where $\Delta = F(\bar{\Theta}^{s,0}, \Theta^{f,0}) - F^\star$, and $\Gamma_1, \Gamma_2, \Gamma_3$ are defined as:

$$\Gamma_1 = (1+\chi^2)\Big(\frac{a_s^2\sigma_s^2}{L_s K M} + \frac{a_f^2\sigma_f^2}{L_f K}\Big),$$

$$\Gamma_2 = \Big(\frac{a_s^3 + \chi^2 a_f a_s^2}{L_s}\Big)(\sigma_s^2 + \sigma_g^2) + \Big(\frac{a_f^3 + \chi^2 a_s a_f^2}{L_f}\Big)\sigma_f^2,\ \Gamma_3 = \Big(\frac{a_s^3 + \chi^2 a_f a_s^2}{L_s}\Big)(\sigma_s^2 + \sigma_g^2).$$

Proof. Summing Eq. (32) from $t = 0$ to $T-1$, using the lower bound $F \ge F^\star$, and applying Lemma 4, we obtain:

$$\begin{aligned}
&\Big(\frac{\eta_s K}{2} - \mathrm{Q}_s\Big)\sum_{t=0}^{T-1}\mathbb{E}[\Delta^{s,t}] + \Big(\frac{\eta_f K}{2} - \mathrm{Q}_f\Big)\sum_{t=0}^{T-1}\mathbb{E}[\Delta^{f,t}] \\
&\le F(\bar{\Theta}^{s,0}, \Theta^{f,0}) - F^\star + T\Big(\frac{3\hat{L}_s K}{2M} + C\mathcal{A}_s\Big)\eta_s^2\sigma_s^2 \\
&+ 6TC\mathcal{A}_s\eta_s^2\sigma_g^2 + T\Big(\frac{3\hat{L}_f K}{2} + C\mathcal{A}_f\Big)\eta_f^2\sigma_f^2 \\
&\quad + \frac{2T\tilde{\mathcal{B}}\eta_s^2 K^2}{(1-\lambda)^2}\Big\{\big[1 + 12K(K-1)(L_s^2 + L_{sf}^2)\eta_s^2\big](\sigma_s^2 + 6\sigma_g^2) \\
&\quad + 12K(K-1)(L_s^2 + L_{sf}^2)\eta_f^2\sigma_f^2\Big\},
\end{aligned} \tag{33}$$

where

$$\begin{aligned}
\mathrm{Q}_s &= \frac{3\hat{L}_s\eta_s^2 K^2}{2} + 6C\mathcal{A}_s\eta_s^2 + \frac{12\tilde{\mathcal{B}}\eta_s^2 K^2}{(1-\lambda)^2}\big[1 + 12K(K-1)(L_s^2 + L_{sf}^2)\eta_s^2\big], \\
\mathrm{Q}_f &= \frac{3\hat{L}_f\eta_f^2 K^2}{2} + 6C\mathcal{A}_f\eta_f^2 + \frac{12\tilde{\mathcal{B}}\eta_s^2 K^2}{(1-\lambda)^2}\big[12K(K-1)(L_s^2 + L_{sf}^2)\eta_f^2\big].
\end{aligned} \tag{34}$$

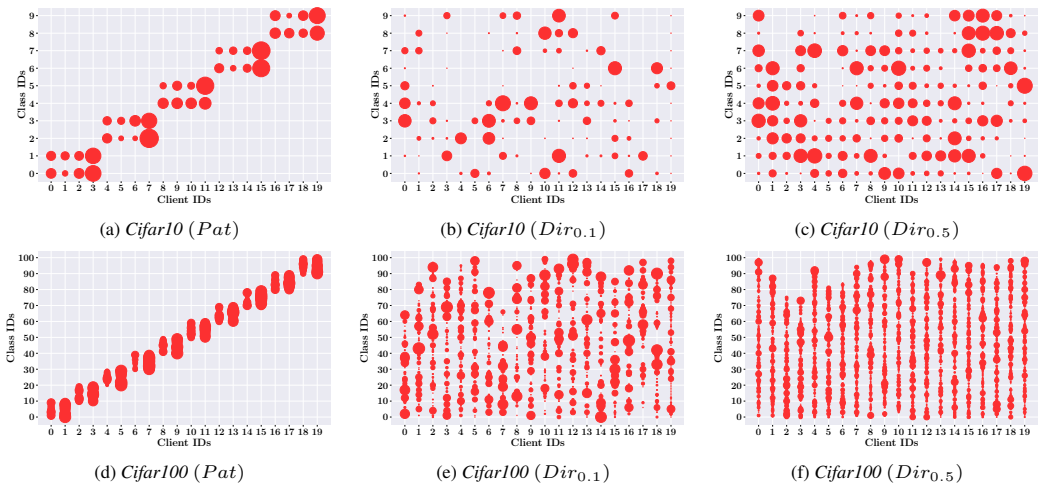
Assume that the learning rates are sufficiently small such that $\mathrm{Q}_s \le \eta_s K/4$ and $\mathrm{Q}_f \le \eta_f K/4$. Substituting $\eta_s = a_s/(L_s K\sqrt{T})$ and $\eta_f = a_f/(L_f K\sqrt{T})$ into Eq. (33), we have:

$$\begin{aligned}
&\frac{1}{T}\sum_{t=0}^{T-1}\Big(\frac{a_s}{L_s}\mathbb{E}[\Delta^{s,t}] + \frac{a_f}{L_f}\mathbb{E}[\Delta^{f,t}]\Big) \\
&\le \mathcal{O}\Big(\frac{F(\bar{\Theta}^{s,0}, \Theta^{f,0}) - F^\star}{\sqrt{T}} + \frac{1+\chi^2}{\sqrt{T}}\Big[\frac{a_s^2\sigma_s^2}{L_s K M} + \frac{a_f^2\sigma_f^2}{L_f K}\Big] \\
&\quad + \frac{1}{T}\Big[\Big(\frac{a_s^3}{L_s} + \frac{\chi^2 a_f a_s^2}{L_s}\Big)(\sigma_s^2 + \sigma_g^2) + \Big(\frac{a_f^3}{L_f} + \frac{\chi^2 a_s a_f^2}{L_f}\Big)\sigma_f^2\Big] \\
&\quad + \frac{1}{(1-\lambda)^2 T}\Big(\frac{a_s^3}{L_s} + \frac{\chi^2 a_f a_s^2}{L_s}\Big)(\sigma_s^2 + \sigma_g^2)\Big).
\end{aligned} \tag{35}$$

Terms containing four or more learning-rate factors decay at least as fast as $T^{-3/2}$ and are absorbed into the displayed T^{-1} terms since $T \ge 1$ and a_s, a_f are fixed sufficiently small constants. □

D.6 Interpretation

Remark 2. *The dominant descent contributions are $\eta_s K\mathbb{E}[\Delta^{s,t}] + \eta_f K\mathbb{E}[\Delta^{f,t}]$. With $\eta_s = a_s/(L_s K\sqrt{T})$ and $\eta_f = a_f/(L_f K\sqrt{T})$, the constants a_s and a_f determine the effective update rates of the slow/shared and fast/personalized variables, respectively. In particular, $a_s/L_s \ll a_f/L_f$ ensures $\eta_s \ll \eta_f$, thereby inducing a two-timescale update scheme. Since both learning rates are of*



(a) *Cifar10* (Pat) (b) *Cifar10* $(Dir_{0.1})$ (c) *Cifar10* $(Dir_{0.5})$

(d) *Cifar100* (Pat) (e) *Cifar100* $(Dir_{0.1})$ (f) *Cifar100* $(Dir_{0.5})$

Figure 12: Data distribution of each client on *Cifar10* and *Cifar100* under different settings. The size of the circle indicates the number of samples.

order $\mathcal{O}(1/\sqrt{T})$, this asymmetric choice changes the relative optimization speed of the two blocks without altering the overall $\mathcal{O}(1/\sqrt{T})$ convergence rate.

Remark 3. *The client-number-dependent variance term in Theorem 2 is $\mathcal{O}\big(a_s^2\sigma_s^2/(MKL_s\sqrt{T})\big)$. Thus, increasing the number of participating clients reduces the stochastic variance associated with the aggregated slow/shared update at the rate $1/M$. This variance reduction does not directly apply to the fast/personalized update, which is maintained locally, nor to the slow-gradient diversity term σ_g^2. Hence, increasing M primarily improves the statistical efficiency of the shared component rather than uniformly reducing all sources of optimization error.*

Remark 4. *The slow-gradient diversity σ_g^2 appears in the higher-order heterogeneity-dependent terms of the convergence bound. Therefore, for fixed a_s, a_f, and other problem parameters, a smaller σ_g^2 leads to a tighter bound. This provides a theoretical motivation for assigning relatively consistent parameters to the slow/shared block: parameters with stronger cross-client variation can be retained in the personalized fast block, thereby reducing the gradient diversity induced by the shared component.*

E Reproducibility Statement

To facilitate reproducibility, we detail the datasets, model architectures, and hyperparameter settings in Sec. B. We provide Python pseudocode for the key functions in Sec. H and include the implementation of our method in the supplementary materials.

F AI Use Statement

In preparing this manuscript, we used generative AI tools solely to polish the language and improve the clarity and fluency of the text. The research ideas, experimental design, and data analysis were developed and conducted by the human authors. Generative AI was not used to generate research ideas or analyze experimental results. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

G Discussion of Limitations and Future Work

Our method improves performance under data heterogeneity by decomposing model parameters into slow and fast components, then tailoring their optimization and aggregation to their distinct characteristics. However, its performance may be affected by poorly chosen hyperparameter settings. In future work, we aim to develop a more robust hyperparameter selection strategy to reduce the

impact of inappropriate settings on performance. We also plan to develop the slow–fast learning approach into a plug-and-play module that can be integrated into centralized, personalized, and decentralized federated learning frameworks.

H PSEUDOCODE OF GeoSF

Algorithm 2 Pseudocode of **GeoSF**

```
import copy
import torch
import numpy as np

class GeoSFAPI(API):
  def __init__(self, dataset, device, args, model_trainer, logger, time):
    super().__init__(dataset, device, args, model_trainer, logger, time)
    self._setup_clients(train_data, test_data, model_trainer)

  def _aggregate_func(self, round_idx, client_idx, client_indexes, w_pers):
    w_i = w_pers[client_idx]
    w_tmp = copy.deepcopy(w_i)
    mask_dict = copy.deepcopy(w_i)

    # Obtain the current neighboring clients
    cur_client_indexes = np.where(client_indexes[client_idx] > 0)[0]

    # Construct the slow-parameter mask
    for name in w_i.keys():
      flat = w_tmp[name].flatten()
      mask = self._get_selected_mask(cur_client_indexes, w_pers, name)
      mask_dict[name] = mask.view_as(w_tmp[name])

    # Construct the multi-gossip mixing matrix
    graph = torch.as_tensor(client_indexes, dtype=torch.bool)
    graph = graph | graph.t()
    graph.fill_diagonal_(False)
    A = torch.zeros((self.args.client_num_in_total, self.args.client_num_in_total))

    for m, n in torch.nonzero(torch.triu(graph, diagonal=1), as_tuple=False):
      distance = self._pair_distance(round_idx, w_pers[m], w_pers[n], mask_dict)
      similarity = torch.exp(-distance / self.args.tau)
      A[m, n] = similarity
      A[n, m] = similarity

    degree = A.sum(dim=1)
    alpha = 1.0 / (degree.max().item() + 1.0)
    W = torch.diag(1.0 - alpha * degree) + alpha * A
    W = torch.linalg.matrix_power(W, self.args.c_n)
    weights = W[client_idx]
    cur_client_indexes = torch.where(weights > 0)[0].cpu().numpy()

    # Aggregate neighboring models
    for name in w_i.keys():
      w_agg = torch.zeros_like(w_tmp[name], dtype=torch.float32)
      for index, j in enumerate(cur_client_indexes):
        weight = weights[j]
        w_agg += weight.to(w_agg.device) * w_pers[j][name].float()
      # Periodically aggregate all parameters; otherwise, only slow ones
      if round_idx % self.args.gap == 0:
        w_tmp[name] = w_agg
      else:
        w_tmp[name] = torch.where(mask_dict[name], w_agg, w_i[name].float())

  return w_tmp, mask_dict

  def _pair_distance(self, round_idx, w_a, w_b, mask_dict):
    distance = 0.0

    for name in self.params.keys():
      mask = mask_dict[name]
```

```
            difference = w_a[name].float() - w_b[name].float()
            distance += torch.sum(difference[mask] ** 2)

        return distance

    def _get_selected_mask(self, client_indexes, w_pers, name, eps=1e-12):
        # Stack the parameters received from neighboring clients
        layer_params = torch.stack([w_pers[j][name].float() for j in client_indexes], dim=0)

        # Measure sign consistency and parameter variance
        sign_consistency = torch.sign(layer_params).mean(dim=0).abs()
        variance = layer_params.var(dim=0, unbiased=False)

        # Assign higher scores to consistent, low-variance parameters
        score = sign_consistency / (torch.sqrt(variance) + eps)

        # Select the parameters with the largest scores as slow parameters
        flat_score = score.flatten()
        mask = torch.zeros_like(flat_score, dtype=torch.bool)
        selected_num = int(flat_score.numel() * self.args.top)
        mask[torch.topk(flat_score, selected_num).indices] = True

        return mask.view_as(score)

class ConsensusGradientDescent(Optimizer):
    def __init__(self, named_params, lr=0.05, lr2=0.2, mu=0.0, momentum=0.0, weight_decay=0.0):
        self.named_params = list(named_params)
        params = [param for _, param in self.named_params]
        defaults = dict(lr=lr, lr2=lr2, mu=mu, momentum=momentum, weight_decay=weight_decay)
        super().__init__(params, defaults)

    @torch.no_grad()
    def step(self, current_w_agg, w_per_mask):
        for group in self.param_groups:
            lr = group["lr"]
            lr2 = group["lr2"]
            mu = group["mu"]
            momentum = group["momentum"]
            weight_decay = group["weight_decay"]

            for name, param in self.named_params:
                if param.grad is None:
                    continue

                # Obtain the stochastic gradient
                direction = param.grad.detach().clone()
                # Apply weight decay
                if weight_decay != 0:
                    direction.add_(param, alpha=weight_decay)
                # Add the consensus regularization to shared parameters
                direction.add_((param - current_w_agg[name]) * w_per_mask[name], alpha=mu)

                # Apply momentum
                if momentum != 0:
                    state = self.state[param]
                    if "momentum_buffer" not in state:
                        state["momentum_buffer"] = direction.clone().detach()
                    else:
                        state["momentum_buffer"].mul_(momentum).add_(direction)
                    direction = state["momentum_buffer"]

                # Use different learning rates for the two parameter blocks
                step_size = torch.where(w_per_mask[name], lr, lr2)
                # Update the model parameters
                param.add_(-step_size * direction)
```