Towards Asynchronous Client Collaboration in Personalized Federated Learning

Boyi Liu[1,2,3], Zimu Zhou[1,2], Pengfei Gao[3], Shuo Kang[3], Yongxin Tong[3]
[1] Department of Data Science, City University of Hong Kong, Hong Kong, China
[2] City University of Hong Kong Shenzhen Research Institute, Shenzhen, China
[3] State Key Laboratory of Complex & Critical Software Environment Lab, Beihang University, Beijing, China
[1,2]boy.liu@my.cityu.edu.hk, zimuzhou@cityu.edu.hk, [3]{gpfgpf, kangshuo, yxtong}@buaa.edu.cn,

***Abstract*—Collaboration-based personalized federated learning (Co-PFL) achieves superior accuracy on non-IID data by leveraging a collaboration graph that captures pairwise similarities among client data distributions for personalized model aggregation. However, its synchronous operation suffers from substantial delays caused by slow clients, limiting its applicability in latency-sensitive IoT applications. A natural alternative is asynchronous Co-PFL, where the server processes model updates as they arrive. Yet, asynchrony introduces staleness that distorts both collaboration graph estimation and personalized model aggregation, two tightly coupled components in Co-PFL. Existing asynchronous federated learning methods only handle staleness in global model aggregation and fail to maintain this coupling. To bridge this gap, we propose PACE, the first asynchronous Co-PFL framework that jointly manages staleness across collaboration estimation and model aggregation. PACE employs two lightweight, theoretically grounded server-side mechanisms: a collaboration-aware buffer update that adaptively refreshes buffered models based on staleness and collaboration relevance, and a staleness-triggered model multicast that selectively disseminates fresh aggregates when collaboration estimation error exceeds a threshold. Extensive evaluations show that PACE matches the accuracy of state-of-the-art synchronous Co-PFL methods, while reducing convergence time by up to $8.58\times$, enabling practical Co-PFL with IoT devices.**

***Index Terms*—Personalized Federated Learning, Asynchronous Federated Learning, Collaboration**

I. Introduction

Collaboration-based personalized federated learning (Co-PFL) [1]–[3] is an emerging paradigm for addressing *data heterogeneity* in federated learning by explicitly modeling *client-to-client* cooperation. It maintains a collaboration graph whose weighted edges quantify the similarity between clients' data distributions, inferred from their model parameters without sharing raw data. In each communication round, clients upload their local models to the server, which estimates the collaboration graph, performs personalized model aggregation based on the graph weights, and returns the updated models to clients for subsequent local training. Such fine-grained aggregation generalizes clustering-based PFL [4]–[9], which imposes hard partitions with uniform intra-cluster weights. By aligning aggregation weights with the underlying structure of inter-client data distributions, Co-PFL achieves high accuracy in non-IID environments [2], [3], and is fit for federated learning in IoT applications such as mobile healthcare [4], smart homes [5], and intelligent transportation [10].

Although Co-PFL delivers superior accuracy, its *synchronous* workflow incurs substantial wall-clock delays in IoT deployments. The server must wait for all clients to complete local training before estimating the collaboration graph and aggregating model updates. This idle waiting becomes particularly severe in the presence of high device heterogeneity [11]–[15]. Empirically, when 30% of clients are $5\times$ slower than the rest, a synchronous round can take $3-12\times$ longer [7], [16], significantly lengthening convergence time. A natural remedy is *asynchronous* Co-PFL, where the server processes model updates as they arrive and immediately returns aggregated models, thereby eliminating idle time caused by stragglers.

To support asynchronous Co-PFL, the server maintains a *buffer* that stores the latest model update from each client. These buffers decouple model arrivals from dispatch but introduce *staleness*, *i.e.*, buffered models may originate from different rounds. Staleness undermines Co-PFL on two fronts. *(i)* The collaboration graph, estimated from pairwise similarities between buffered models, can be distorted when those models reflect different training states. *(ii)* Personalized aggregation performed with a mixed-age buffer may send outdated or inconsistent parameters back to clients, hindering convergence. Existing staleness management strategies for asynchronous federated learning of a single global model address only aggregation [17]–[19]. In contrast, Co-PFL must control staleness jointly in collaboration graph estimation and personalized model aggregation, keeping the two steps synergetic. Designing mechanisms that coordinate these coupled operations while preserving the latency advantage of asynchrony is therefore essential to realizing effective asynchronous Co-PFL.

In this paper, we propose PACE (Personalization under Asynchronous Collaboration Estimation), the first Co-PFL framework designed for fully asynchronous clients. PACE preserves coherence between collaboration graph estimation and personalized model aggregation under unbounded staleness through two lightweight, theoretically grounded server-side mechanisms. *(i) Collaboration-Aware Buffer Update.* Upon receiving a new model update, the server refreshes the buffers of all other clients using a fine-grained decay factor that reflects both the update's staleness and the collaboration weights. It aligns the freshness of each buffered model with its relevance during asynchronous model aggregation, so pairwise similarities remain meaningful even when updates arrive

out-of-sync. *(ii) Staleness-Triggered Model Multicast.* When the staleness-induced error in the estimated collaboration graph exceeds a threshold, the server multicasts a small set of freshly aggregated models to selected clients. This targeted intervention improves accuracy without altering the estimation algorithm or incurring excessive communication overhead. Together, these passive-active staleness management mechanisms enable PACE to operate asynchronously, while preserving personalization quality and convergence guarantees.

Our main contributions are as follows.

- We present PACE, the first asynchronous Co-PFL framework. It eliminates straggler-induced delays and makes collaboration-based personalization practical in heterogeneous IoT environments.
- We introduce two synergistic staleness-control mechanisms and prove that their integration enables convergent asynchronous Co-PFL at rate of $O(1/\sqrt{T})$, where T is the number of communication rounds.
- Extensive evaluations show that PACE matches the accuracy of state-of-the-art Co-PFL baselines [1]–[3] while converging 2.33-8.58$\times$ faster in terms of wall-clock time.

II. Related Work

Personalized Federated Learning. PFL combats data heterogeneity by collaboratively training models tailored to each client's local data distribution. Existing studies fall into two categories. *(i) Model-based personalization.* These methods partition model parameters into global and local components, or design client-specific architectures to capture individual data characteristics [20]–[23]. *(ii) Similarity-based personalization.* These approaches aggregate updates selectively based on inter-client data similarity, and are increasingly popular in IoT applications [4], [5], [8]. There are two subcategories: *Clustering-based PFL* partitions clients into disjoint groups and shares models within each cluster [4]–[9]; *Collaboration-based PFL* (Co-PFL) generalizes this by learning a weighted graph that enables *fine-grained* pairwise aggregation [1]–[3].

While Co-PFL offers higher flexibility, its operation remains *synchronous* and vulnerable to straggler delays. We fill in this gap by introducing asynchrony into Co-PFL, allowing adaptive pairwise aggregation without waiting for slow clients.

Asynchronous Federated Learning. AFL reduces straggler-induced latency by aggregating server updates as soon as they arrive, instead of waiting for all clients. The primary challenge in AFL is *staleness*, where outdated updates may destabilize convergence. Most AFL algorithms incorporate staleness-aware aggregation via decay mechanisms. For instance, FedAsync [17] introduces a freshness-based decay function. FedBuff [24] averages a buffer of recent updates. PORT [18] jointly considers update divergence and staleness. FedAC [19] incorporates weighted momentum into global model update. ASAFL [25] adaptively weights aggregation using model divergence.

However, these methods are designed for a *single* global model. Conversely, we bring asynchrony to Co-PFL, where a learned collaboration graph guides personalized model aggregation. We analyze how staleness affects collaboration estimation and model aggregation, and propose effective staleness control strategies to improve the accuracy of Co-PFL in asynchronous environments.

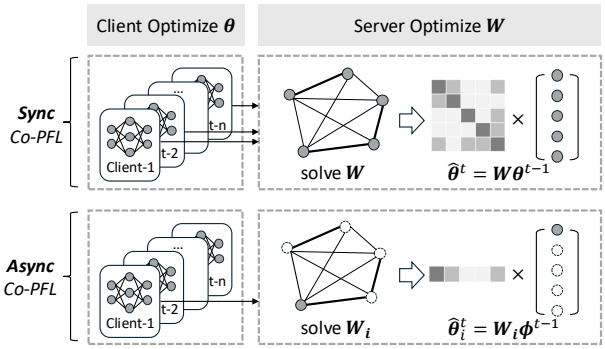


Fig. 1: Co-PFL overview.

III. Preliminary and Problem

A. Co-PFL

Consider a federation of n clients $\mathcal{C} = \{c_1, \ldots, c_n\}$. Each client c_i holds a private dataset D_i and maintains a personalized model $\theta_i \in \mathbb{R}^d$. *Collaboration-based personalized federated learning (Co-PFL)* [1]–[3] aims to learn personalized models $\boldsymbol{\theta} = \{\theta_1, \ldots, \theta_n\}$ guided by a *collaboration graph* $G(V, W)$, where $V = \mathcal{C}$ and $W \in \mathbb{R}^{n \times n}$ encodes pairwise collaboration intensities. Ideally, W_{ij} quantifies the similarity between the data distributions of clients i and j. Yet due to inaccessibility of raw data in federated learning, W_{ij} is often estimated via model similarity.

Formally, Co-PFL jointly optimizes the personalized models and the collaboration graph as follows.

$$\begin{aligned} &\min_{\boldsymbol{\theta}, W} \sum_{i=1}^{n} p_i \left(F_i \left(\sum_{j=1}^{n} W_{ij}\theta_j; D_i \right) + \mathcal{S}(W, \boldsymbol{\theta}) \right) \\ &\text{s.t.} \sum_{j=1}^{n} W_{ij} = 1, \forall i; \quad W_{ij} \geq 0, \forall i, j \end{aligned} \tag{1}$$

where $p_i = |D_i| / \sum_{j=1}^{n} |D_j|$ is the relative sample size, $F_i(\cdot; D_i)$ is the local empirical risk of client i, and $\mathcal{S}(W, \boldsymbol{\theta})$ encourages W to reflect model similarities measured by L_1, L_2-based kernels [1] or cosine similarity [2], [3]. Finally, each client receives a personalized model $\widehat{\theta}_i = \sum_{j=1}^{n} W_{ij}\theta_j$.

Such formulation enables *fine-grained* client collaboration, generalizing clustering-based PFL [4]–[9], which imposes hard partitions with uniform intra-cluster weights.

B. Synchronous Optimization Framework

In standard Co-PFL, Eq. (1) is solved over T *synchronous* rounds by alternately updating W and $\boldsymbol{\theta}$ (upper part of Fig. 1).

- *Server-Side Collaboration Estimation and Model Aggregation.* At round t, the server infers client i's collaboration intensities $W_i = \{W_{ij}\}_{j=1}^n$ via:

$$\begin{aligned} &\min_{W_i} \sum_{j=1}^{n} (W_{ij} - p_j)^2 + \gamma \sum_{j=1}^{n} W_{ij} s(\theta_i^{t-1}, \theta_j^{t-1}) \\ &\text{s.t.} \sum_{j=1}^{n} W_{ij} = 1, \quad W_{ij} \geq 0 \quad \forall i, j \end{aligned} \tag{2}$$

  where the first term favors clients with larger data sizes and the second penalizes those with dissimilar model parameters. $s(\cdot, \cdot)$ is a similarity metric (*e.g.* cosine or L_1, L_2). Let $\boldsymbol{W} = [W_1, \ldots, W_n]^\top \in \mathbb{R}^{n \times n}$ be a matrix stacking all the collaboration intensities $\{W_1, \ldots W_n\}$. The server aggregates the personalized models $\boldsymbol{\theta}^{t-1} = [\theta_1^{t-1}, \ldots, \theta_n^{t-1}]^\top$ from the previous round as:

$$\widehat{\boldsymbol{\theta}}^t = \boldsymbol{W} \boldsymbol{\theta}^{t-1} \tag{3}$$

  and sends the aggregated models $\{\widehat{\theta}_1^t, \ldots, \widehat{\theta}_n^t\}$ to clients.
- *Client-Side Local Training.* Upon receiving $\widehat{\theta}_i^t$, client i updates its personalized model by minimizing:

$$\mathcal{F}_i(\theta; D_i) = F_i(\theta; D_i) + \lambda s(\theta, \widehat{\theta}_i^t) \tag{4}$$

  where the first term is the local empirical risk and the second promotes consistency with the aggregated model. The local model is updated via SGD $\theta_i^t \leftarrow \widehat{\theta}_i^t - \nabla \mathcal{F}_i(\widehat{\theta}_i^t; D_i)$, which is then uploaded to the server for the next round.

C. Asynchronous Co-PFL

We extend the above framework to *asynchronous* Co-PFL, where clients communicate with the server independently. In this setting, client updates arrive at *different times*, making synchronous collaboration estimation (*i.e.*, Eq. (2)) and model aggregation (*i.e.*, Eq. (3)) infeasible (lower part of Fig. 1).

To address this, the server maintains a *buffer* of the *latest received* models $\{\phi_1, \ldots, \phi_n\}$ from clients. Let τ_i be the staleness of client i's model update, defined as the gap between the current server round t and the round $t - \tau_i$ at which client i last received a model from the server. Upon receiving a delayed model $\theta_i^{t-\tau_i}$ from client i, the server *(i)* immediately sends ϕ_i back to client i as the latest personalized model for subsequent local training; and *(ii)* uses $\theta_i^{t-\tau_i}$ to update both the collaboration graph and the buffers for other clients.

The buffers enable asynchronous model *reception* and *dispatch*, yet introduce unique challenges to Co-PFL.

- **Inaccurate Collaboration Estimation.** The server infers collaboration vector W_i with $s(\theta_i^{t-\tau_i}, \phi_j)$ instead of $s(\theta_i^{t-1}, \theta_j^{t-1})$ in Eq. (2). The stale models lead to inaccurate collaboration estimation, potentially overestimating similarity between unrelated clients.
- **Biased Model Aggregation.** The server constructs personalized models $\widehat{\boldsymbol{\theta}}^t$ with buffered models $\{\phi_1, \ldots, \phi_n\}$, which are not aligned in version. Accordingly, it becomes non-trivial to assign meaningful collaboration weights, leading to aggregation bias and model divergence.

Table. I summarizes the major notations used in this paper.

TABLE I: Summary of major notations.

Notation	Definition
n	number of clients
$D_i; \lvert D_i \rvert$	local dataset of client i; size of D_i
$D; \lvert D \rvert$	dataset of all clients; size of $\lvert D \rvert$
τ_i	staleness of client i
$\theta_i^{t-\tau_i}$	model of client i starting at round t
ϕ_i	model of client i cached at server
$\widehat{\theta}_i^t$	fresh model of client i aggregated at round t
F_i	main task of client i
E	epoch number
T	communication round
$s(\cdot, \cdot)$	similarity metric
$W; W_{ij}$	collaboration graph; collaboration of client i and j
α	default decay coefficient for aggregation
α_{ij}	decay coefficient for client i updating client j

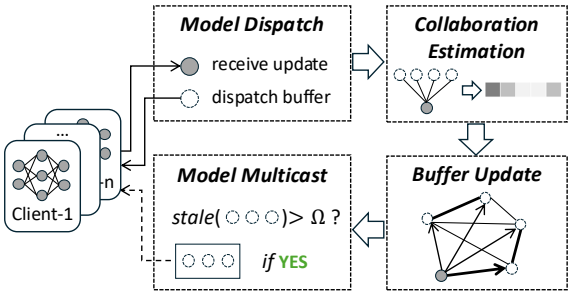


Fig. 2: PACE workflow.

IV. Method

A. PACE Overview

We propose PACE (Personalization under Asynchronous Collaboration Estimation), which effectively manages staleness for asynchronous Co-PFL (Fig. 2).

Workflow. PACE follows the client and server operations in Sec. III-B, with major changes at the *server* to accommodate asynchronous clients (Algorithm 1). Assume the server receives a delayed model update $\theta_i^{t-\tau_i}$ from client i in asynchronous round t. The server executes the following steps.

- *Model Dispatch*. Upon receiving $\theta_i^{t-\tau_i}$, the server immediately sends the current buffered model ϕ_i to client i, allowing it to begin a new local training round without waiting for other clients.
- *Collaboration Estimation*. The server estimates the collaboration intensities using Eq. (2) but calculates the similarity as $s(\theta_i^{t-\tau_i}, \phi_j)$ rather than $s(\theta_i^{t-1}, \theta_j^{t-1})$.
- *Buffer Update*. The server updates the buffers of *all other* clients with $\theta_i^{t-\tau_i}$, considering both staleness and inferred collaboration intensities. Specifically, buffer ϕ_j for client j $(j \neq i)$ is updated as:

$$\phi_j = \alpha_{ij} \theta_i^{t-\tau_i} + (1 - \alpha_{ij}) \phi_j \tag{5}$$

  where α_{ij} is a decay that decides how $\theta_i^{t-\tau_i}$ is aggregated into the buffer ϕ_j, and correspondingly, the personalized

Algorithm 1: PACE workflow.

Input: Clients' model parameters $\theta_1, ..., \theta_n$
Output: Clients' model parameters $\theta_1, ..., \theta_n$

Server Process:
for *asynchronous round* t **do**
  Receive update $\theta_i^{t-\tau_i}$ from client i
  // Model Dispatch
  Downlink $\theta_i^t \leftarrow \phi_i$
  // Collaboration Estimation
  Update buffer $\phi_i \leftarrow \theta_i^{t-\tau_i}$
  $W_i \leftarrow$ solve Eq. (2)
  // Buffer Update
  for *client* j **do**
    Weight decay $\alpha_{ij} \leftarrow$ Eq. (11)
    Update $\phi_j \leftarrow \alpha_{ij}\theta_i^{t-\tau_i} + (1-\alpha_{ij})\phi_j$
  // Model Multicast
  $\mathcal{G}^t \leftarrow$ solve Eq. (16)
  for *client* $i \in \mathcal{G}^t$ **do**
    Update local model $\theta_i^t \leftarrow \phi_i$ via multicast

Client Process:
while *True* **do**
  Download model ϕ_i
  Local training via Eq. (4)
  Upload $\theta_i^{t-\tau_i}$ to server

model to be dispatched to client j. This buffer update step *replaces* the synchronous model aggregation in Eq. (3).

- *Model Multicast*. The server sends updated buffers to a selection of clients to mitigate the impact of staleness on collaboration estimation at a tunable communication cost.

The client-side operations remain unchanged from Sec. III-B.

Principles. Our modifications to the server-side operations are inspired by prior research on asynchronous federated learning (AFL), which uses *(i)* passive staleness-aware decay for aggregating stale updates [17]–[19], and *(ii)* active model broadcasting to bound staleness [8], [16]. However, existing schemes fail for Co-PFL due to two distinctions. *(i)* AFL aggregates updates into a *single global* model, whereas Co-PFL maintains *multiple personalized* models. *(ii)* Co-PFL requires *collaboration estimation*, an extra step than AFL that is sensitive to staleness.

PACE integrates both passive and active staleness control strategies and organizes them in two key operations.

- **Collaboration-Aware Buffer Update** (Sec. IV-B). This module implements passive staleness control. We design a fine-grained decay mechanism that modulates the influence of $\theta_i^{t-\tau_i}$ on buffer ϕ_j, based on both its staleness τ_i and the collaboration intensity W_{ij}. Crucially, we ensure that model updates from different versions are aggregated with aligned collaboration intensities.
- **Staleness-Triggered Model Multicast** (Sec. IV-C). This module implements active staleness control. We analyze how staleness degrades the accuracy of collaboration estimation and develop a strategy that bounds this degradation by multicasting updated buffer models to selected clients. This strategy improves accuracy without modifying the collaboration estimation algorithm itself.

These components ensure convergent training (Sec. IV-D) in asynchronous Co-PFL.

B. Collaboration-Aware Buffer Update

This subsection explains how PACE computes decay α_{ij} in Eq. (5) to incorporate both *collaboration* and *staleness*.

Collaboration-Based Decay. To ensure that the asynchronous buffer update as in Eq. (5) reflects the correct collaboration, *i.e.*, client i's model is aggregated into client j's buffer ϕ_j *in proportion to* W_{ij}, we define a collaboration-based decay as:

$$\alpha_{ij} = \frac{W_{ji}}{\sum_{\pi \in \boldsymbol{\pi}_j \cup \{j\}} W_{j\pi}}, \tag{6}$$

where $\boldsymbol{\pi}_j$ is the *sequence* of clients that have updated ϕ_j.

Proof. Suppose buffer ϕ_j is updated by a sequence of clients $\boldsymbol{\pi}_j = \{\pi_j^1, \ldots, \pi_j^L\}$, and the corresponding models aggregated into ϕ_j are $\boldsymbol{\psi}_j = \{\psi_j^1, \ldots, \psi_j^L\}$. The buffer ϕ_j is updated as:

$$\begin{cases} \phi_j^1 = \alpha_j^1 \psi_j^1 + (1-\alpha_j^1)\phi_j^0, \\ \vdots \\ \phi_j^L = \alpha_j^L \psi_j^L + (1-\alpha_j^L)\phi_j^{L-1}, \end{cases} \tag{7}$$

with decay $\boldsymbol{\alpha}_j = \{\alpha_j^1, \ldots, \alpha_j^L\}$.

Unfolding the recurrence yields:

$$\begin{aligned} \phi_j^L = {} & \alpha_j^L \psi_j^L + (1-\alpha_j^L)\alpha_j^{L-1}\psi_j^{L-1} + \cdots \\ & + \left(\prod_{l=2}^{L}(1-\alpha_j^l)\alpha_j^1\psi_j^1\right) + \left(\prod_{l=1}^{L}(1-\alpha_j^l)\phi_j^0\right). \end{aligned} \tag{8}$$

To preserve the collaboration intensities as if these models were aggregated synchronously as in Eq. (3), decay $\boldsymbol{\alpha}_j$ should match their normalized collaboration intensities:

$$\begin{cases} \frac{W_{jj}}{W_{jj}+\sum_{\pi\in\boldsymbol{\pi}_j} W_{j\pi}} = \prod_{l=1}^{L}(1-\alpha_j^l), \\ \frac{W_{j\pi_j^1}}{W_{jj}+\sum_{\pi\in\boldsymbol{\pi}_j} W_{j\pi}} = \prod_{l=2}^{L}(1-\alpha_j^l)\alpha_j^1, \\ \vdots \\ \frac{W_{j\pi_j^L}}{W_{jj}+\sum_{\pi\in\boldsymbol{\pi}_j} W_{j\pi}} = \alpha_j^L. \end{cases} \tag{9}$$

Solving Eq. (9) gives:

$$\alpha_j^l = \frac{W_{j\pi_j^l}}{W_{jj} + \sum_{q=1}^{l} W_{j\pi_j^q}}, \quad \forall\ 1 \le l \le L. \tag{10}$$

Hence, upon receiving a model update from client i, the decay to buffer ϕ_j becomes $\alpha_{ij} = \frac{W_{ji}}{\sum_{\pi\in\boldsymbol{\pi}_j\cup\{j\}} W_{j\pi}}$. □

Incorporating Staleness-Based Decay. To further account for the delay in received updates $\theta_i^{t-\tau_i}$, incorporate a staleness-based decay term, following standard AFL practice [17]–[19].

We adopt a *polynomial* staleness-based penalty $(1+\tau_i)^{-a}$ as [17] for convergent asynchronous Co-PFL.

Combining this with the collaboration-based decay, the final decay becomes

$$\alpha_{ij} = \frac{W_{ji}}{\sum_{\pi \in \boldsymbol{\pi}_j \cup \{j\}} W_{j\pi}} \cdot (1+\tau_i)^{-a}. \tag{11}$$

The client sequence $\boldsymbol{\pi}_j$ is reset to $\emptyset$ if the server receives a new update from client j. This is because past models that have already been aggregated should no longer influence future updates from the same client.

C. Staleness-Triggered Model Multicast

This subsection analyzes the impact of staleness on collaboration estimation in Eq. (2) and presents a staleness-aware multicast scheme for accurate collaboration estimation without excessive communication cost.

Staleness-Induced Collaboration Error. Claim 1 quantifies the collaboration estimation error via Eq. (2) with delayed model $\theta_i^{t-\tau_i}$ from client i and buffers $\{\phi_j\}$.

Claim 1. *(Collaboration estimation error). The divergence between the estimated collaboration vector $\tilde{W}_i$ and the ground-truth W_i for client i is:*

$$\|W_i - \tilde{W}_i\| \leq \sum_{j=1}^{n} \frac{\tau_i \delta \eta E G}{2\sum_{\pi \in \boldsymbol{\pi}_j \cup \{j\}} W_{j\pi} G_j}, \tag{12}$$

where δ is defined in Definition 1 of Appendix A, η is the learning rate, E is the number of local epochs, G is the upper bound of gradient, and $G_j = \|\phi_j\|$.

Claim 1 can be proved by first bounding the divergence between the estimated pairwise similarity s_{ij} and the ground-truth $\tilde{s}_{ij}$ and then taking it as noise when solving Eq. (2). Below is a proof sketch assuming the widely adopted cosine similarity [2], [3]. The proof also applies to other similarity metrics such as L_1, L_2 and inner product.

Proof. We first bound the divergence between s_{ij} and $\tilde{s}_{ij}$ as:

$$\begin{aligned} s_{ij} - \tilde{s}_{ij} &= \cos\left(\theta_i^{t-\tau_i}, \theta_j^{t-\tau_i}\right) - \cos\left(\theta_i^{t-\tau_i}, \phi_j^t\right) \\ &= \frac{\left\langle \theta_i^{t-\tau_i}, \theta_j^{t-\tau_i} \right\rangle}{\|\theta_i^{t-\tau_i}\| \cdot \|\theta_j^{t-\tau_i}\|} - \frac{\left\langle \theta_i^{t-\tau_i}, \phi_j^t \right\rangle}{\|\theta_i^{t-\tau_i}\| \cdot \|\phi_j^t\|} \overset{(a)}{\approx} \frac{\left\langle \theta_i^{t-\tau_i}, \theta_j^{t-\tau_i} - \phi_j^t \right\rangle}{\|\theta_i^{t-\tau_i}\| \cdot \|\phi_j^t\|} \\ &\overset{(b)}{\leq} \frac{\|\theta_j^{t-\tau_i} - \phi_j^t\|}{\|\phi_j^t\|} \overset{(c)}{\leq} \frac{\tau_i \delta \eta E G}{\sum_{\pi \in \boldsymbol{\pi}_j \cup \{j\}} W_{j\pi} G_j} \end{aligned} \tag{13}$$

where (a) is by the approximation that the gradient norms of a single client between close rounds are similar $\|\phi_j\| \approx \|\theta_j^{t-\tau_i}\|$, (b) is by properties of inner product $\langle a, b\rangle \leq \|a\| \cdot \|b\|$, (c) is by Lemma 2 of the convergence analysis (Appendix A).

From Eq. (13), we have:

$$\|s_i - \tilde{s}_i\| \leq \sum_{j=1}^{n} \|s_{ij} - \tilde{s}_{ij}\| \leq \sum_{j=1}^{n} \frac{\tau_i \delta \eta E G}{\sum_{\pi \in \boldsymbol{\pi}_j \cup \{j\}} W_{j\pi} G_j} \tag{14}$$

Following [2], [3], we solve Eq. (2) as quadratic program, yet with noise in s_i bounded by Eq. (14). Formally, Eq. (2) can be rewritten into:

$$\min_{W_i} W_i^\top W_i + (-2\boldsymbol{p} + \gamma s_i)^\top W_i \quad \text{s.t.} \sum_{j=1}^{n} W_{ij} = 1,\ W_{ij} \geq 0\ \forall j \tag{15}$$

where $\boldsymbol{p} = [p_1, p_2, \ldots, p_n]^\top$ is the relative dataset size vector. For a function $f(W_i) = W_i^\top W_i$, there is $\nabla f(W_i) = 2W_i$ and $\nabla^2 f(W_i) = 2I \succeq 2I$. Consequently, $f(W_i)$ is a 2-convex function. From perturbation theory [26], there is $\|W_i - \tilde{W}_i\| \leq \|s_i - \tilde{s}_i\|/\mu$, where μ is a strong convexity parameter. For $f(W_i)$ we have $\mu = 2$. Then we have: $\|W_i - \tilde{W}_i\| \leq \sum_{j=1}^{n} \frac{\tau_i \delta \eta E G}{2\sum_{\pi \in \boldsymbol{\pi}_j \cup \{j\}} W_{j\pi} G_j}$ □

Model Multicast. Claim 1 shows that the error in collaboration estimation is bounded by the staleness of client models $\theta_i^{t-\tau_i}$. Accordingly, we actively control the collaboration estimation error by *selectively* disseminating the latest models (in the buffers) to clients to refresh their local models.

Formally, we check whether there is a group of clients $\mathcal{G}^t$ actively participating in training, which satisfies:

$$\sum_{i \in \mathcal{G}^t} \tau_i^2 > \Omega, \quad |\mathcal{G}^t| \times \theta.size() \leq \mathcal{P} \tag{16}$$

where Ω is a threshold for the maximum tolerable staleness, and $\mathcal{P}$ is the communication budget. Following [27], we assume a maximum downlink bandwidth of $B = 10$ MHz, a tolerable latency of $\overline{\tau} = 300$ ms, and an SNR of 10 dB, and thus a budget of $\mathcal{P} = \overline{\tau} B \log(1 + \text{SNR}) = 1.297$MB. When the conditions in Eq. (16) are satisfied, multicast is triggered and the buffered models are sent to client sets $\mathcal{G}^t$.

D. Analysis

Convergence Analysis. The effectiveness of our asynchronous Co-PFL framework is guaranteed by the theorem below. The complete proof is in Appendix A.

Theorem 1. *(Convergence of PACE) Algorithm 1 converges to an arbitrary constant ϵ with a convergence rate of $\mathcal{O}(1/\sqrt{T})$:*

$$\frac{1}{T}\sum_{t=0}^{T-1} p_i \sum_{e \in U_i^t} \|\nabla F_i^{t,e}\|^2 \leq \frac{\frac{\Delta}{T} + \delta\eta^2 EGW' + LE\eta^2\sigma^2/n^2}{\eta - L\eta^2/2} \tag{17}$$

where n is the number of clients, E is the number of local epochs, T is the total communication rounds, η is the learning rate, L is defined in Assumption 1, G is defined in Assumption 3, and σ is defined in Assumption 4.

Communication Cost. PACE only exchanges model parameters as standard federated learning. However, the staleness-triggered model multicast (Sec. IV-C) introduces extra *downlink* communication. Our empirical evaluation shows that such overhead is marginal. For example, PACE incurs 1.01-1.05× communication rounds of standard AFL baselines (*e.g.* FedAsync [17]) in five setups (Sec. V-D2). Since the bandwidth is typically much larger at downlink than uplink [8], [28], these extra communication rounds lead to negligible latency measured in wall-clock time.

V. Experiments

A. Experimental Setup

Baselines. We compare PACE with the following FL methods.

- *Synchronous* FL: FedAvg [29] and FedProx [30].
- *Asynchronous* FL: FedAsync [17], FedBuff [24], PORT [18], FedAC [19], and ASAFL [25].
- *Synchronous* Co-PFL: FedAMP [1], pFedGraph [2], and FedSaC [3].
- *Asynchronous variants* of Co-PFL: Since there are no prior asynchronous Co-PFL methods available, we extend the synchronous Co-PFL baselines above into asynchronous versions, which are denoted as FedAMP-Async, pFedGraph-Async, and FedSaC-Async.

Datasets and Models. We experiment with three tasks on five datasets: *(i)* Image classification on EMNIST[1] [31] and CIFAR-10/CIFAR-100 [32]; *(ii)* Human activity recognition on HARBox [4]; *(iii)* Text classification on AGNews [33]. We adopt the Dirichlet-based partitioning [34] to simulate non-IID data distribution. Specifically, we use $Dir(0.1)$ for EMNIST, CIFAR-10, and CIFAR-100, and $Dir(0.3)$ for AGNews. We also test the pathological non-IID setting [29], where each client holds data from a fixed number of labels: $\#C = 10$ for EMNIST, $\#C = 2$ for CIFAR-10, and $\#C = 20$ for CIFAR-100. HARBox contains sensor data collected from 120 participants, naturally reflecting real-world data heterogeneity.

For CIFAR-10, CIFAR-100, we apply a CNN as [2]. For AgNews, we apply a TextCNN as [35]. For HARBox, we apply a two-layer MLP as [7].

Configurations. We simulate a federation of 100 clients. For synchronous FL methods, the client sampling rate is 0.1. For asynchronous FL methods, the maximum parallelism is 10%. We set 30% clients as slow devices, whose local training latency is five times that of normal devices [7], [16], [36].

The total number of communication rounds T is set to 200 for synchronous FL methods and $2,000$ for asynchronous FL methods, with $E = 5$ local training epochs per round. Wall-clock time is simulated with a priority queue mechanism [18].

The learning rate η is set to 0.01 for CNN and TextCNN, and 0.001 for MLP. We use SGD as the optimizer with batch size 64, momentum 0.9, and a weight decay of 1×10^{-4}.

Environment. The experiments are conducted on a machine equipped with an Intel Xeon Gold 6230R CPU and NVIDIA A100 GPUs (40GB memory).

Hyperparameters. For FedProx, $\mu = 0.5$. For FedAsync, we use the *hinge* setup with $a = 1$, $b = 4$. For FedBuff, $\eta_g = 10$, $k = 10$. For PORT, $\alpha = 1$, $\beta = 1$, $\Omega = 10$, and the minimal client scale for aggregation is 5. For FedAC, $\beta = 0.9$, $\eta_g = 0.01$, $\eta_l = 0.01$, and the buffer size is 5. For ASAFL, $\eta_s = 0.01$, $\eta_c = 0.01$. For FedAMP-Async, $\lambda = 1$, $\sigma = 10$, $\xi_{ii} = 0.7$. For pFedGraph-Async, $\alpha = 1.5$, $\lambda = 0.01$. For FedSaC-Async, $\alpha = 1.5$, $\beta = 1.5$, $\lambda = 0.01$. For PACE, $\gamma = 1.5$, $\lambda = 0.01$, $a = 1.5$, $\Omega = 2500$.

[1]For EMNIST, we use the *Balanced* subset, which contains 47 classes.

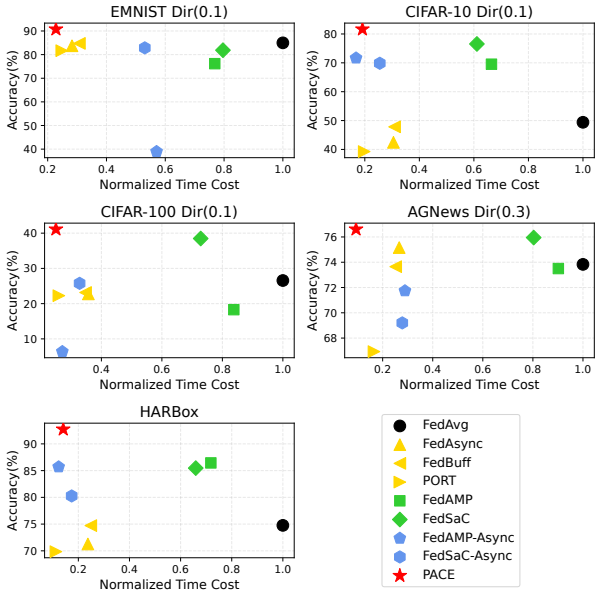


Fig. 3: Accuracy v.s. normalized convergence time.

Metrics. We consider two metrics. *(i) Accuracy*: test accuracy of the personalized models on local datasets; and *(ii) Convergence Time*: wall-clock time of training till convergence.

B. Main Results

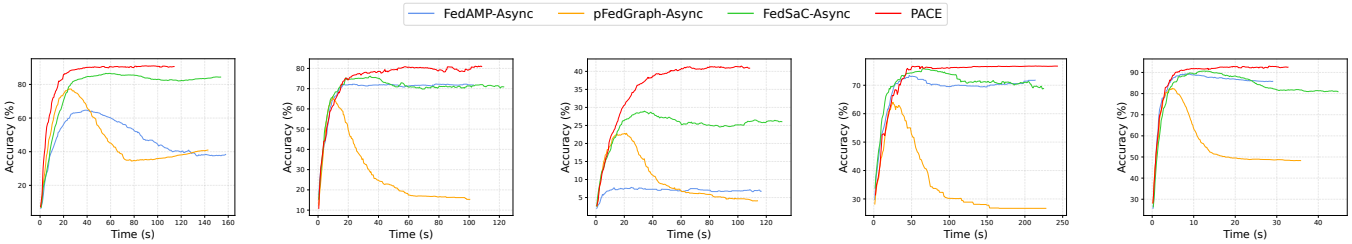
1) Accuracy: Table. II lists the accuracy (averaged over clients) on five datasets. PACE achieves the highest accuracy across all settings and datasets. Compared to synchronous (FedAvg, FedProx) and asynchronous (FedAsync, FedBuff, PORT, FedAC, ASAFL) FL baselines *without* personalization, PACE improves the accuracy by up to 9.52% on EMNIST, 47.5% on CIFAR-10, 27.32% on CIFAR-100, 15.13% on AGNews, and 22.88% on HARBox, which validates the need for training personalized models in non-IID settings. The asynchronous Co-PFL baselines (FedAMP-Async, pFedGraph-Async, FedSaC-Async) underperform their synchronous counterparts (FedAMP, pFedGraph, FedSaC), implying the non-trivial challenges to incorporate asynchrony into Co-PFL (our motivation). Our PACE outperforms these Co-PFL baselines by at least 7.90%, 9.98%, 12.84%, 5.09% and 7.03% on the five datasets, respectively, which shows that PACE enables accurate, fine-grained collaboration with asynchronous clients.

2) Convergence Time vs. Accuracy: To further understand the tradeoff between accuracy and convergence (measured in wall-clock time), Fig. 3 plots the accuracy and normalized convergence time (w.r.t FedAvg) of PACE and eight baselines, including FedAvg (for normalization), three asynchronous FL methods, two synchronous Co-PFL methods and their asynchronous variants that yield the top accuracies. Other baselines are omitted for clear visualization.

As expected, synchronous Co-PFL methods converge slower than their asynchronous counterparts because they need to wait for slow clients. As the first asynchronous Co-PFL scheme, our PACE only requires 43.00%, 31.30%, 34.28%, 11.66%,

TABLE II: Accuracy of all methods (highest accuracy marked in bold).

Method	EMNIST		CIFAR-10		CIFAR-100		AGNews	HARBox
	$\#C = 10$	$Dir(0.1)$	$\#C = 2$	$Dir(0.1)$	$\#C = 20$	$Dir(0.1)$	$Dir(0.3)$	*Real-world*
FedAvg [29]	$82.38_{\pm 0.35}$	$84.98_{\pm 0.23}$	$45.32_{\pm 0.07}$	$49.40_{\pm 0.07}$	$23.77_{\pm 0.14}$	$26.56_{\pm 0.50}$	$72.27_{\pm 1.56}$	$74.76_{\pm 0.14}$
FedProx [30]	$82.65_{\pm 1.17}$	$84.43_{\pm 0.16}$	$46.08_{\pm 0.45}$	$51.07_{\pm 1.56}$	$23.34_{\pm 0.63}$	$26.55_{\pm 0.27}$	$67.97_{\pm 0.55}$	$74.90_{\pm 0.09}$
FedAsync [17]	$80.62_{\pm 0.08}$	$83.74_{\pm 0.20}$	$40.76_{\pm 0.91}$	$42.45_{\pm 0.44}$	$20.12_{\pm 0.28}$	$22.76_{\pm 0.19}$	$74.20_{\pm 0.96}$	$71.28_{\pm 1.42}$
FedBuff [24]	$81.24_{\pm 1.19}$	$84.47_{\pm 0.17}$	$43.89_{\pm 0.17}$	$47.83_{\pm 1.92}$	$17.04_{\pm 0.34}$	$23.16_{\pm 0.45}$	$70.62_{\pm 3.03}$	$74.72_{\pm 0.70}$
PORT [18]	$68.43_{\pm 0.01}$	$81.67_{\pm 0.19}$	$32.48_{\pm 0.75}$	$39.26_{\pm 0.09}$	$17.43_{\pm 0.25}$	$22.26_{\pm 0.25}$	$64.90_{\pm 2.04}$	$69.85_{\pm 1.16}$
FedAC [19]	$83.36_{\pm 0.42}$	$84.65_{\pm 0.17}$	$45.48_{\pm 0.19}$	$52.81_{\pm 0.47}$	$20.61_{\pm 0.07}$	$24.09_{\pm 0.38}$	$75.77_{\pm 0.01}$	$76.14_{\pm 0.15}$
ASAFL [25]	$80.71_{\pm 0.28}$	$81.23_{\pm 0.03}$	$42.81_{\pm 0.70}$	$41.12_{\pm 1.54}$	$10.14_{\pm 0.13}$	$13.79_{\pm 0.30}$	$61.56_{\pm 0.82}$	$71.93_{\pm 0.85}$
FedAMP [1]	$80.70_{\pm 0.31}$	$76.17_{\pm 0.68}$	$56.71_{\pm 0.31}$	$69.53_{\pm 0.09}$	$23.58_{\pm 0.37}$	$18.29_{\pm 0.18}$	$73.50_{\pm 0.24}$	$86.44_{\pm 0.06}$
pFedGraph [2]	$85.32_{\pm 0.01}$	$81.71_{\pm 0.00}$	$77.43_{\pm 0.09}$	$76.80_{\pm 0.03}$	$22.82_{\pm 0.06}$	$38.05_{\pm 0.01}$	$75.85_{\pm 0.12}$	$85.52_{\pm 0.10}$
FedSaC [3]	$85.54_{\pm 0.31}$	$81.71_{\pm 0.09}$	$77.63_{\pm 0.20}$	$76.54_{\pm 0.17}$	$22.69_{\pm 0.04}$	$38.48_{\pm 0.09}$	$75.95_{\pm 0.07}$	$85.47_{\pm 0.19}$
FedAMP-Async	$16.77_{\pm 0.41}$	$38.97_{\pm 0.92}$	$50.33_{\pm 0.11}$	$71.67_{\pm 0.15}$	$1.07_{\pm 0.18}$	$6.39_{\pm 0.52}$	$71.60_{\pm 0.14}$	$85.70_{\pm 0.10}$
pFedGraph-Async	$72.64_{\pm 1.55}$	$42.98_{\pm 2.20}$	$59.90_{\pm 0.33}$	$16.70_{\pm 1.38}$	$4.80_{\pm 0.25}$	$3.91_{\pm 0.18}$	$26.83_{\pm 0.14}$	$45.84_{\pm 2.48}$
FedSaC-Async	$86.59_{\pm 0.26}$	$82.85_{\pm 1.53}$	$68.89_{\pm 0.27}$	$69.85_{\pm 0.93}$	$12.19_{\pm 0.38}$	$25.76_{\pm 0.33}$	$67.49_{\pm 1.72}$	$80.26_{\pm 0.82}$
PACE (Ours)	$\mathbf{91.97_{\pm 0.13}}$	$\mathbf{90.75_{\pm 0.21}}$	$\mathbf{79.98_{\pm 0.05}}$	$\mathbf{81.65_{\pm 0.62}}$	$\mathbf{25.03_{\pm 0.22}}$	$\mathbf{41.11_{\pm 0.11}}$	$\mathbf{76.69_{\pm 0.04}}$	$\mathbf{92.73_{\pm 0.20}}$



(a) EMNIST $Dir(0.1)$ (b) CIFAR-10 $Dir(0.1)$ (c) CIFAR-100 $Dir(0.1)$ (d) AGNews $Dir(0.3)$ (e) HARBox

Fig. 4: Accuracy vs. convergence time of asynchronous Co-PFL methods.

and 21.37% of the convergence time compared to the fastest Co-PFL baseline on the five datasets. PACE also converges faster than FedAvg, the synchronous FL baseline, reducing its convergence time to 22.81%, 19.12%, 24.97%, 9.37% and 14.09% on the five datasets. Among the asynchronous methods, our PACE is still the fastest on EMNIST, CIFAR-100, and AGNews, and incurs a comparable convergence time to FedAMP-Async and PORT on CIFAR-10 and HARBox.

In summary, when jointly comparing the convergence time and accuracy against FedAvg on non-IID datasets, synchronous Co-PFL methods improve accuracy and decrease convergence time, but the reduction in convergence time is marginal. Asynchronous FL baselines substantially shorten the convergence time by allowing asynchronous model aggregation, yet suffer from low accuracy. Naively extending synchronous Co-PFL methods to incorporate asynchrony does not necessarily improve the accuracy. In contrast, our PACE explicitly orchestrates collaboration and asynchrony, thus notably advancing the Pareto front between convergence time and model accuracy.

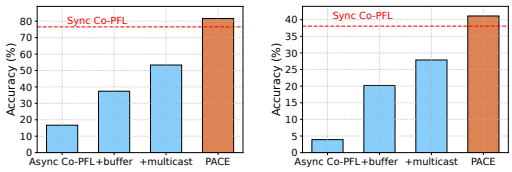
C. Detailed Comparison of Asynchronous Co-PFL Methods

Fig. 4 presents the accuracy-convergence training curves of PACE and three asynchronous Co-PFL methods. The accuracy of the three asynchronous Co-PFL baselines show a noticeable drop after the initial increase. This is because they do not explicitly manage staleness during client collaboration. The biased global model, coupled with misaligned update rounds, leads to an inaccurate collaboration graph, and thus degraded accuracy. Among these baselines, pFedGraph-Async suffers the most severe accuracy drop. Although our PACE adopts the same collaboration graph formulation as pFedGraph-Async, it consistently achieves the highest accuracy and fastest convergence, verifying the effectiveness of our design.

TABLE III: Accuracy comparison when extending Co-PFL methods with PACE and their naive asynchronous variants.

Method	CIFAR-10 $Dir(0.1)$	CIFAR-100 $Dir(0.1)$	AGNews $Dir(0.3)$
FedAMP-Async	71.67	6.39	71.60
FedAMP-PACE	**79.87**(+8.20)	**37.21**(+30.82)	**77.39**(+5.79)
FedSaC-Async	69.85	25.76	67.49
FedSaC-PACE	**81.58**(+11.73)	**40.97**(+15.21)	**76.87**(+9.38)

To show that our collaboration-aware staleness management scheme generalizes to other Co-PFL formulations, we further integrate PACE into FedAMP and FedSaC. As shown in Table. III, when extending synchronous Co-PFL methods with PACE, the resulting asynchronous versions drastically outperform the naive extentions (*i.e.*, FedAMP-Async and FedSaC-Async). On CIFAR-10, CIFAR-100, and AGNews, the gains in accuracy are 8.20%, 30.82%, and 5.79% for FedAMP; and 11.73%, 15.21%, and 9.38% for FedSaC. These results show the potential of PACE as a generic plug-in strategy to enhance various Co-PFL schemes.



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

Fig. 5: Contributions of individual components.

TABLE IV: Impact of similarity metrics on accuracy of PACE.

	Cosine	Inner product	L_1	L_2
EMNIST	$90.75_{\pm 0.21}$	$90.78_{\pm 0.01}$	$\mathbf{90.86_{\pm 0.06}}$	$90.83_{\pm 0.01}$
CIFAR-10	$81.65_{\pm 0.62}$	$\mathbf{82.61_{\pm 0.11}}$	$82.33_{\pm 0.14}$	$82.40_{\pm 0.30}$
CIFAR-100	$41.11_{\pm 0.11}$	$\mathbf{41.58_{\pm 0.10}}$	$41.38_{\pm 0.09}$	$41.40_{\pm 0.08}$
AGNews	$76.69_{\pm 0.04}$	$76.97_{\pm 0.06}$	$76.92_{\pm 0.07}$	$\mathbf{77.02_{\pm 0.13}}$
HARBox	$92.73_{\pm 0.20}$	$\mathbf{93.13_{\pm 0.20}}$	$93.02_{\pm 0.20}$	$93.11_{\pm 0.02}$

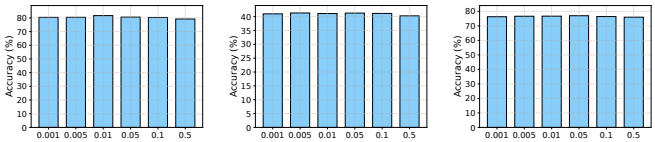
D. Ablation Studies

1) Contributions of Individual Components: Fig. 5 zooms into the contributions of individual component in PACE, by gradually adding modules into pFedGraph-Async, which shares the same collaboration graph formulation with PACE. The first bar is the accuracy of the naive asynchronous Co-PFL. The second bar adds the buffer module (Sec. IV-A). The third bar further adds the multicast module (Sec. IV-C). The final bar shows the results after including staleness-aware decay, which makes the full PACE (Sec. IV-B).

The buffer mechanism actively updates models and improves the accuracy by 20.69%, 16.30%. The multicast mechanism further calibrates the collaboration graph, with another accuracy improvement of 15.93%, 7.66%. With staleness-aware decay, the received models provide accurate and up-to-date updates to the buffers, enabling PACE to further increase the accuracy by 28.33%, 13.24%.

2) Communication Overhead: The effectiveness of PACE comes with extra communication due to its staleness-triggered multicast (Sec. IV-C). As a quantitative comparison, PACE incurs $1.03\times$, $1.01\times$, $1.01\times$, $1.04\times$, and $1.05\times$ the communication frequency, including the count of model dispatch and model multicast, of standard asynchronous FL baselines (*i.e.*, FedAsync) on EMNIST, CIFAR-10, CIFAR-100, AGNews, and HARBox, respectively. The marginally increased communication frequency is acceptable compared its contributions in accuracy (see Sec. V-D1).

3) Impact of Similarity Metric: We adopt the *cosine* similarity, which is widely used in Co-PFL, for the main experiments (Sec. V-B). Yet the effectivenss of PACE is agnostic to the similarity metric $s(\cdot,\cdot)$. As an empirical verification, we test the accuracy of PACE using L_1, L_2, and inner product, three other similarity metrics in Co-PFL. As shown in Table. IV, all the similarity metrics yield comparable accuracy. PACE even achieves a slightly higher accuracy using other similarity metrics.



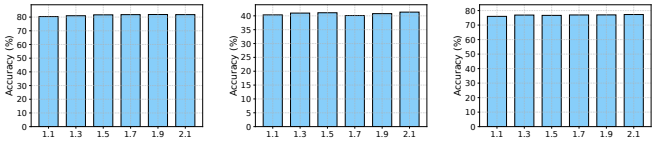
(a) CIFAR-10 (b) CIFAR-100 (c) AGNews

Fig. 6: Impact of hyperparmater λ.

E. Hyperparameter Sensitivity

1) Impact of λ: Fig. 6 studies hyperparameter sensitivity of λ, which balances the regularization on local training (see Eq. (4)). In PACE, we set $\lambda = 0.01$, and tune λ across $\{0.001, 0.005, \mathbf{0.01}, 0.05, 0.1, 0.5\}$ on CIFAR-10, CIFAR-100 and AGNews. PACE performs similarly across the six hyperparameter configurations.

2) Impact of γ: Fig. 7 presents the hyperparameter sensitivity of γ, which determines the solution of collaboration graph (see Eq. (2)). In PACE, we set $\gamma = 1.5$, and tune γ across $\{1.1, 1.3, \mathbf{1.5}, 1.7, 1.9, 2.1\}$ on CIFAR-10, CIFAR-100 and AGNews. As is shown, PACE is robust to γ.



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

Fig. 7: Impact of hyperparmater γ.

VI. Conclusion

We proposed PACE, the first asynchronous framework for collaboration-based personalized federated learning. PACE jointly manages staleness in collaboration graph estimation and personalized model aggregation through a collaboration-aware buffer update and staleness-triggered model multicast. Extensive experiments show that PACE achieves state-of-the-art accuracy while significantly reducing convergence time, making it a practical solution for efficient, personalized federated learning in real-world asynchronous environments.

Acknowledgments

This work is partially supported by the RGC of Hong Kong SAR, China (Project No. CityU 11206425), National Science Foundation of China (NSFC) (Grant Nos. 62572412, 62425202, 62336003), the Beijing Natural Science Foundation (Z230001), the Fundamental Research Funds for the Central Universities No. JK2024-03, the Didi Collaborative Research Program, the City University of Hong Kong Shenzhen Research Institute (CityUHKSRI), and the State Key Laboratory of Complex & Critical Software Environment (SKLCCSE). Zimu Zhou and Yongxin Tong are the corresponding authors.

APPENDIX

A. Proof of Convergence Analysis

We analyze the convergence of PACE following the same proof framework as FedAsync [17], which monitors the change in the objective function $F(\cdot)$ per asynchronous communication round. However, FedAsync [17] only applies a constant staleness-based decay α, whereas our PACE adopts a fine-grained decay related to both staleness and collaboration.

Proof. Let $F^{t,A}$ be the objective F of all clients at round t before local training, and $F^{t,L}$ be the objective F at round t before server aggregation. Our goal is to derive bounds on $\mathbb{E}[F^{t,L}] - F^{t,A}$ and $\mathbb{E}[F^{t+1,A}] - F^{t,L}$.

We first make the following assumptions.

Assumption 1. *(Smoothness) The local objective F_i of client i is L-smooth, satisfying: $\|\nabla F_i(x) - \nabla F_i(y)\| \leq L\|x-y\|$, which can be reformulated as: $F_i(y) \leq F_i(x) + \langle \nabla F_i(x), y - x\rangle + \frac{L}{2}\|y-x\|^2$.*

Assumption 2. *(Unbiased gradient) The expectation of stochastic gradient $g_i(\theta_i; z)$ is an unbiased estimator of the local gradient, satisfying: $\mathbb{E}_{z\sim D_i} g_i(\theta_i; z) = \nabla F_i(\theta_i)$.*

Assumption 3. *(Bounded gradient) The expectation square norm of stochastic gradient $g_i(\theta_i; z)$ is bounded, satisfying: $\|g_i(\theta_i; z)\|^2 \leq G^2$.*

Assumption 4. *(Bounded gradient variance) The variance of stochastic gradient $g_i(\theta_i; z)$ is bounded, satisfying: $\mathbb{E}_{z\sim D_i}\|g_i(\theta_i; z) - \nabla F_i(\theta_i)\|^2 \leq \sigma^2$.*

Assumptions 1 to 4 are commonly adopted in federated learning [17], [37]. Unlike most asynchronous FL schemes [17], [18], [24], PACE does not assume bounded staleness, as its collaboration-based decay naturally diminishes as updates accumulate (Sec. IV-B). For simplicity, let $\nabla F_i(\theta_i)$ be ∇F_i, and $g_i(\theta_i; z)$ be g_i.

Lemma 1. *(Divergence during local training) We provide the divergence of objective F during local training at round t:*

$$\mathbb{E}[F^{t,L}] - F^{t,A} \leq \sum_{i=1}^{n} p_i \sum_{e\in U_i^t}((\frac{L\eta^2}{2} - \eta)\|\nabla F_i^{t,e}\|^2 + \frac{L\eta^2\sigma^2}{2}) \tag{18}$$

Proof. Assume that at asynchronous round t, client i starts from local epoch u_i^t, and ends at local epoch v_i^t. Let U_i^t be $[u_i^t, v_i^t]$, for client i at round t, there is:

$$\begin{aligned}
\mathbb{E}[F_i^{t,L}] - F_i^{t,A} &\overset{(a)}{\leq} \sum_{e\in U_i^t}(-\eta\mathbb{E}[\langle \nabla F_i^{t,e}, g_i^{t,e}\rangle] + \frac{L\eta^2}{2}\mathbb{E}[\|g_i^{t,e}\|^2]) \\
&\overset{(b)}{\leq} \sum_{e\in U_i^t}(-\eta\mathbb{E}[\langle \nabla F_i^{t,e}, g_i^{t,e}\rangle] + \frac{L\eta^2}{2}(\mathbb{E}[\|\nabla F_i^{t,e}\|^2] + \sigma^2)) \\
&\overset{(c)}{=} \sum_{e\in U_i^t}(-\eta\|\nabla F_i^{t,e}\|^2 + \frac{L\eta^2}{2}(\mathbb{E}[\|\nabla F_i^{t,e}\|^2] + \sigma^2))
\end{aligned} \tag{19}$$

where (a) is by smoothness of $F_i(\cdot)$ as in Assumption 1, (b) is by $Var(x) = \mathbb{E}[x^2] - (\mathbb{E}[x])^2$, (c) is by Assumption 2. Then the divergence of the overall objective F is:

$$\mathbb{E}[F^{t,L}] - F^{t,A} \leq \sum_{i=1}^{n} p_i \sum_{e\in U_i^t}((\frac{L\eta^2}{2} - \eta)\|\nabla F_i^{t,e}\|^2 + \frac{L\eta^2\sigma^2}{2}) \tag{20}$$

□

Definition 1. *(Model update heterogeneity) For arbitrary i, j, given client model updates θ_i and θ_j, there exists $\delta > 0$, satisfying $\|\theta_i - \theta_j\| \leq \delta\eta EG/W_{ij}$.*

Lemma 2. *(Bounded local-global divergence) At round t, the divergence between global model $\theta_{g,i}^t$ and received local model $\theta_i^{t-\tau_i}$ satisfies:*

$$\|\theta_i^{t-\tau_i} - \phi_i^t\| \leq \textstyle\sum_{\pi\in\boldsymbol{\pi}_i\cup\{i\}} \frac{\delta\eta EG}{\sum_{\pi\in\boldsymbol{\pi}_i\cup\{i\}} W_{i\pi}} \tag{21}$$

Proof. Note that collaboration-based decay α_{ij} is proportional to the collaboration intensity W_{ij}. Consequently, we can expand ϕ_i^t into combinations of received model updates $\{\psi_\pi\}_{\pi\in\boldsymbol{\pi}_i\cup\{i\}}$. Then we have:

$$\begin{aligned}
\|\theta_i^{t-\tau_i} - \phi_i^t\| &\overset{(a)}{\leq} \|\theta_i^{t-\tau_i} - \textstyle\sum_{\pi\in\boldsymbol{\pi}_i\cup\{i\}} \frac{W_{i\pi}}{\sum_{\pi\in\boldsymbol{\pi}_i\cup\{i\}} W_{i\pi}}\psi_\pi\| \\
&\overset{(b)}{\leq} \textstyle\sum_{\pi\in\boldsymbol{\pi}_i\cup\{i\}} \frac{W_{i\pi}}{\sum_{\pi\in\boldsymbol{\pi}_i\cup\{i\}} W_{i\pi}}\|\theta_i^{t-\tau_i} - \psi_\pi\| \\
&\overset{(c)}{\leq} \textstyle\sum_{\pi\in\boldsymbol{\pi}_i\cup\{i\}} \frac{\delta\eta EG}{\sum_{\pi\in\boldsymbol{\pi}_i\cup\{i\}} W_{i\pi}}
\end{aligned} \tag{22}$$

where (a) is by expanding ϕ_i^t, (b) is by $\|x+y\| \leq \|x\| + \|y\|$, (c) is by Definition 1. □

Lemma 3. *(Divergence during aggregation) We provide the divergence of objective F during aggregation at round t:*

$$\mathbb{E}[F^{t+1,A}] - F^{t,L} \leq \delta\eta^2 EG/\overline{W}_i \tag{23}$$

where $\overline{W}_i = \sum_{\pi\in\boldsymbol{\pi}_i\cup\{i\}} W_{i\pi}/|\boldsymbol{\pi}_i \cup \{i\}|$ is the averaged collaboration relationship of models updating client i.

Proof. Given that only the buffer of client i is updated, we have: $\mathbb{E}[F^{t+1,A}] - F^{t,L} = \mathbb{E}[F_i^{t+1,A}] - F_i^{t,L}$. By taking Lemma 2 into Lemma 2 of [38], we complete the proof.

□

Convergence of PACE. By putting Lemma 1 and Lemma 3 together, we have:

$$\begin{aligned}
\mathbb{E}[F^{t+1,A}] - F^{t,A} &\leq \delta\eta^2 EG/\overline{W}_i \\
&+ \textstyle\sum_{i=1}^{n} p_i \sum_{e\in U_i^t}((\frac{L\eta^2}{2} - \eta)\|\nabla F_i^{t,e}\|^2 + \frac{L\eta^2\sigma^2}{2})
\end{aligned} \tag{24}$$

We telescope Eq. (24) from round 0 to $T-1$, and let $\Delta = F^{0,A} - \mathbb{E}[F^{T,A}]$, and $W' = \frac{1}{T}\sum_{t=0}^{T-1}\frac{1}{\overline{W}_{i(t)}}$, where $i(t)$ represents the clients received at round t. We get:

$$\textstyle\frac{1}{T}\sum_{t=0}^{T-1} p_i \sum_{e\in U_i^t}\|\nabla F_i^{t,e}\|^2 \leq \frac{\frac{\Delta}{T} + \delta\eta^2 EGW' + LE\eta^2\sigma^2/n^2}{\eta - L\eta^2/2} \tag{25}$$

Therefore, we prove that for an arbitrary constant $\epsilon > 0$, as $T > 0$, $\Delta > 0$, Eq. (25) converges to ϵ when: $\eta \leq \min(\frac{2}{L}, \frac{2n^2\epsilon}{n^2L\epsilon + 2n^2\delta W' EG + 2LE\sigma^2})$. Consequently, when the learning rate η satisfies the condition above, PACE converges. If the learning rate η is set $\mathcal{O}(1/\sqrt{T})$, the convergence rate of PACE is $\mathcal{O}(1/\sqrt{T})$. □

REFERENCES

[1] Y. Huang, L. Chu, Z. Zhou, L. Wang, J. Liu, J. Pei, and Y. Zhang, "Personalized cross-silo federated learning on non-iid data," in *AAAI*, vol. 35, no. 9, 2021, pp. 7865–7873.

[2] R. Ye, Z. Ni, F. Wu, S. Chen, and Y. Wang, "Personalized federated learning with inferred collaboration graphs," in *ICML*, 2023, pp. 39 801–39 817.

[3] K. Yan, S. Cui, A. Wuerkaixi, J. Zhang, B. Han, G. Niu, M. Sugiyama, and C. Zhang, "Balancing similarity and complementarity for federated learning," in *ICML*, 2024.

[4] X. Ouyang, Z. Xie, J. Zhou, J. Huang, and G. Xing, "Clusterfl: a similarity-aware federated learning system for human activity recognition," in *MobiSys*, 2021, pp. 54–66.

[5] X. Ouyang, Z. Xie, H. Fu, S. Cheng, L. Pan, N. Ling, G. Xing, J. Zhou, and J. Huang, "Harmony: Heterogeneous multi-modal federated learning through disentangled model training," in *MobiSys*, 2023, pp. 530–543.

[6] R. Zhou, J. Yu, R. Wang, B. Li, J. Jiang, and L. Wu, "A reinforcement learning approach for minimizing job completion time in clustered federated learning," in *INFOCOM*, 2023, pp. 1–10.

[7] B. Liu, Y. Ma, Z. Zhou, Y. Shi, S. Li, and Y. Tong, "Casa: Clustered federated learning with asynchronous clients," in *SIGKDD*, 2024, pp. 1851–1862.

[8] X. Li, S. Liu, Z. Zhou, B. Guo, Y. Xu, and Z. Yu, "Echopfl: Asynchronous personalized federated learning on mobile devices with on-demand staleness control," *IMWUT*, vol. 8, no. 1, pp. 1–22, 2024.

[9] X. Li, S. Liu, Z. Zhou, Y. Xu, B. Guo, and Z. Yu, "Classter: Mobile shift-robust personalized federated learning via class-wise clustering," *IEEE Transactions on Mobile Computing*, vol. 24, no. 03, pp. 2014–2028, 2025.

[10] T. Zheng, A. Li, Z. Chen, H. Wang, and J. Luo, "Autofed: Heterogeneity-aware federated multimodal learning for robust autonomous driving," in *MobiCom*, 2023, pp. 1–15.

[11] C. Li, X. Zeng, M. Zhang, and Z. Cao, "Pyramidfl: A fine-grained client selection framework for efficient federated learning," in *MobiCom*, 2022, pp. 158–171.

[12] L. Shen, Q. Yang, K. Cui, Y. Zheng, X.-Y. Wei, J. Liu, and J. Han, "Fedconv: A learning-on-model paradigm for heterogeneous federated clients," in *MobiSys*, 2024, pp. 398–411.

[13] K. Wang, Z. Zhou, and Z. Li, "Latte: Layer algorithm-aware training time estimation for heterogeneous federated learning," in *MobiCom*, 2024, pp. 1470–1484.

[14] M. Gan, L. Li, S. Alam, L. Liu, L. Liu, M. Zhang, and Z. Cao, "Geofl: A framework for efficient geo-distributed cross-device federated learning," in *INFOCOM*, 2025, pp. 1–10.

[15] L. Qu, S. Li, Z. Zhou, B. Liu, Y. Xu, and Y. Tong, "Darkdistill: Difficulty-aligned federated early-exit network training on heterogeneous devices," in *SIGKDD*, 2025, pp. 2374–2385.

[16] J. Liu, J. Jia, T. Che, C. Huo, J. Ren, Y. Zhou, H. Dai, and D. Dou, "Fedasmu: Efficient asynchronous federated learning with dynamic staleness-aware model update," in *AAAI*, vol. 38, no. 12, 2024, pp. 13 900–13 908.

[17] C. Xie, S. Koyejo, and I. Gupta, "Asynchronous federated optimization," *arXiv preprint arXiv:1903.03934*, 2019.

[18] N. Su and B. Li, "How asynchronous can federated learning be?" in *IWQoS*, 2022, pp. 1–11.

[19] Y. Zang, Z. Xue, S. Ou, L. Chu, J. Du, and Y. Long, "Efficient asynchronous federated learning with prospective momentum aggregation and fine-grained correction," in *AAAI*, vol. 38, no. 15, 2024, pp. 16 642–16 650.

[20] J. Zhang, S. Guo, X. Ma, W. Xu, Q. Zhou, J. Guo, Z. Hong, and J. Shan, "Model decomposition and reassembly for purified knowledge transfer in personalized federated learning," *IEEE Transactions on Mobile Computing*, 2024.

[21] A. Li, J. Sun, P. Li, Y. Pu, H. Li, and Y. Chen, "Hermes: an efficient federated learning framework for heterogeneous mobile clients," in *MobiCom*, 2021, pp. 420–437.

[22] A. Li, J. Sun, X. Zeng, M. Zhang, H. Li, and Y. Chen, "Fedmask: Joint computation and communication-efficient personalized federated learning via heterogeneous masking," in *SenSys*, 2021, pp. 42–55.

[23] S. Li, B. Liu, Z. Zhou, and J. Dong, "Fedaims: Adaptive intermediate supervision for personalized federated learning," *Frontiers of Computer Science*, 2025.

[24] J. Nguyen, K. Malik, H. Zhan, A. Yousefpour, M. Rabbat, M. Malek, and D. Huba, "Federated learning with buffered asynchronous aggregation," in *AISTATS*, 2022, pp. 3581–3607.

[25] W. Zhang, D. Deng, X. Wu, W. Zhao, Z. Liu, T. Zhang, J. Kang, and D. Niyato, "An adaptive asynchronous federated learning framework for heterogeneous internet of things," *Information Sciences*, vol. 689, p. 121458, 2025.

[26] J. F. Bonnans and A. Shapiro, *Perturbation analysis of optimization problems*. Springer Science & Business Media, 2013.

[27] J. Xu and H. Wang, "Client selection and bandwidth allocation in wireless federated learning networks: A long-term perspective," *IEEE Transactions on Wireless Communications*, vol. 20, no. 2, pp. 1188–1200, 2020.

[28] S. Chen and J. Zhao, "The requirements, challenges, and technologies for 5g of terrestrial mobile telecommunication," *IEEE Communications Magazine*, vol. 52, no. 5, pp. 36–43, 2014.

[29] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *AISTATS*, 2017, pp. 1273–1282.

[30] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," *MLSys*, vol. 2, pp. 429–450, 2020.

[31] G. Cohen, S. Afshar, J. Tapson, and A. Van Schaik, "Emnist: Extending mnist to handwritten letters," in *IJCNN*, 2017, pp. 2921–2926.

[32] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009.

[33] X. Zhang, J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," *NeurIPS*, vol. 28, 2015.

[34] T.-M. H. Hsu, H. Qi, and M. Brown, "Measuring the effects of non-identical data distribution for federated visual classification," *arXiv preprint arXiv:1909.06335*, 2019.

[35] J. Zhang, Y. Hua, H. Wang, T. Song, Z. Xue, R. Ma, and H. Guan, "Fedala: Adaptive local aggregation for personalized federated learning," in *AAAI*, vol. 37, no. 9, 2023, pp. 11 237–11 244.

[36] B. Liu, S. Li, Z. Zhou, S. Kang, Y. Ma, and Y. Tong, "Poster: Asynchronous federated learning library and benchmark with afl-lib," in *MobiCom*, 2025, pp. 1281–1283.

[37] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of fedavg on non-iid data," in *ICLR*, 2020.

[38] L. Yi, H. Yu, C. Ren, G. Wang, X. Li *et al.*, "Federated model heterogeneous matryoshka representation learning," *NeurIPS*, vol. 37, pp. 66 431–66 454, 2024.