ACCELERATING FEDERATED LEARNING CONVERGENCE VIA OPPORTUNISTIC MOBILE RELAYING

Anonymous authors
Paper under double-blind review

ABSTRACT

This paper studies asynchronous Federated Learning (FL) subject to clients' individual arbitrary communication patterns with the parameter server. We propose FedMobile, a new asynchronous FL algorithm that exploits the mobility attribute of the mobile FL system to improve the learning performance. The key idea is to leverage the random client-client communication in a mobile network to create additional indirect communication opportunities with the server via upload and download relaying. We prove that FedMobile achieves a convergence rate $O(\frac{1}{\sqrt{NT}})$, where N is the number of clients and T is the number of communication slots, and show that the optimal design involves an interesting trade-off on the best timing of relaying. Our analysis suggests that with an increased rate of client-client communication opportunities, asynchronous FL converges faster using FedMobile. Experiment results on a synthetic dataset and two real-world datasets verify our theoretical findings.

1 INTRODUCTION

Federated learning (FL) McMahan et al. (2017) is a distributed machine learning paradigm where a number of clients with decentralized data work collaboratively to learn a common model under the coordination of a centralized server. The majority of FL algorithms McMahan et al. (2017); Li et al. (2019); Khaled et al. (2020); Yang et al. (2020); Yu et al. (2019) study the synchronous communication setting where clients can synchronize and communicate with the server periodically and simultaneously. The communication frequency is thus a key design parameter of the FL algorithm under the implicit assumption that the communication channel between the clients and the server is always and universally available. This assumption, however, hardly holds in many real-world systems where clients have only sporadic communication opportunities with the server, and the communication patterns vary among the clients. For example, mobile clients (e.g., mobile devices, sensors, vehicles) can communicate with the server (e.g., base stations, sensing hubs, road side units) only when they meet the server (i.e., entering the communication range of the server). In such mobile systems, FL must be executed asynchronously obeying clients' individual communication patterns with the server.

The literature on asynchronous FL is much smaller than that on its synchronous counterpart. Although some insights have been derived (e.g., Avdiukhin & Kasiviswanathan (2021) proves that the convergence rate of asynchronous FL can match that of synchronous FL under the same communication interval), the performance of asynchronous FL is significantly limited by the communication patterns of individual clients as they are arbitrary and *not* an algorithm parameter. With only occasional client-server meetings, asynchronous FL can converge slowly or even fail to converge.

This paper studies asynchronous FL in a mobile network setting and demonstrates that the communication patterns of the mobile clients can be "virtually" reshaped to improve the FL convergence performance by exploiting a key attribute of the mobile FL system, namely *mobility*. Mobility causes the sporadic client-server meetings but at the same time also creates (the largely neglected) communication opportunities *among the clients*. This latter consequence of mobility enables a client to *indirectly* communicate with the server by using another client as the relay, thereby creating additional communication opportunities (both upload and download) with the server. In particular, the client's local model updates can be uploaded to the server sooner if the relay client meets the server before the sending client does. Likewise, the client can receive a fresher global model from a relay

client that more recently met the server. We present a motivating example, namely FL in Vehicular Ad Hoc Networks (VANET), in the supplementary material where the considered setting and the assumptions made are justified.

Main Contributions. Our study starts with a simplified yet practical setting where a client is allowed to use at most one upload relay communication and at most one download relay communication between any of its two consecutive meetings with the server (considering the additional cost due to client-to-client communication). We design a new asynchronous FL algorithm with opportunistic relaying, called **FedMobile**, that addresses the following key questions: (1) when to upload (download) via relaying and who should be the relay? (2) how to relay the local model updates to (the global model from) the server? We prove the convergence of FedMobile and reveal an interesting trade-off of the best timing of relaying for both the upload and download cases, and that the mobility improves the convergence speed of asynchronous FL. Furthermore, we extend FedMobile to allow multiple relay communications between two consecutive server meetings. Extensive experiments on a synthetic dataset and two real-world datasets verify our theoretical findings.

We note that this paper is focused on the theoretical understanding of the convergence speed of FL under opportunistic mobile relays. Relaying incentives and the security issues associated with relaying are orthogonal to the research topic of this paper. Nevertheless, in the supplementary material, we also present an extension where clients manipulate data before relaying it to the server. Such data manipulation can be used to reduce the relaying cost as well as enhance privacy during relaying.

2 Related Work

FedAvg McMahan et al. (2017) uses local stochastic gradient descent (SGD) to reduce the number of communications between the server and clients. Convergence has been analyzed for FedAvg Li et al. (2019); Khaled et al. (2020); Yang et al. (2020); Yu et al. (2019) and its variants (e.g., momentum variants Wang et al. (2019); Liu et al. (2020) and variance-reducing variants Konečnỳ et al. (2016); Karimireddy et al. (2020)) in both iid and non-iid data settings. However, the majority of works on FL study the synchronous setting and treat the communication frequency as a tunable parameter of the algorithm.

Asynchronous distributed optimization/learning was studied in the past several years. These works Mitliagkas et al. (2016); Hadjis et al. (2016); Chen et al. (2016); Zheng et al. (2017); Dai et al. (2018) consider a single SGD step by the distributed nodes with iid data distributions, which are not typical settings of FL. The literature on asynchronous FL is much smaller but different works have vastly different focuses. For example, some existing works Chen et al. (2018); Smith et al. (2017); Nishio & Yonetani (2019) still assume universal communication at all times and the asynchronicity is the result of an algorithm decision rather than a constraint. Some other works van Dijk et al. (2020); Chai et al. (2021); Chen et al. (2020); Li et al. (2021) use asynchronous model aggregation to address the "straggler" problem encountered in synchronous FL. The asynchronous setting closest to ours is studied in Avdiukhin & Kasiviswanathan (2021); Basu et al. (2019), which considered arbitrary communication patterns. However, these works only considered the interaction between the server and the clients, neglecting the interaction among clients that may be leveraged to improve the FL performance.

Our work is remotely related to decentralized FL Lalitha et al. (2019); Zhang et al. (2021); Xing et al. (2020) (and some hybrid FL works Guo et al. (2021)) where clients can also communicate among themselves during the training process, typically by using a type of gossip algorithm to exchange local model information. However, these works assume a fixed topology of clients and the communication among the clients is still synchronous and periodic.

3 Model and Preliminaries

We consider a mobile FL system with one server and N mobile clients. The mobile clients work together to train a machine learning model by solving a distributed optimization problem as follows:

$$\min_x f(x) = \frac{1}{N}\sum_{i=1}^{N} f_i(x) = \frac{1}{N}\sum_{i=1}^{N} \mathbb{E}_{\zeta_i}[F_i(x,\zeta_i)] \tag{1}$$

where $f_i : \mathbb{R}^d \to \mathbb{R}$ is a non-convex loss function for client i and F_i is the estimated loss function based on a mini-batch data sample ζ_i drawn from client i's own dataset.

Mobility Model. We consider a discrete time system where time is divided into slots of equal length. Clients move independently in a network and make contact with the server only at certain time slots. At each time t and for each client i, let $\tau_i^{\text{last}}(t)$ be the last time when client i meets the server by t (including t), and $\tau_i^{\text{next}}(t)$ be the next time when client i will meet the server (excluding t). At any t, client i knows $\tau_i^{\text{next}}(t)$ but may not know the future meeting times. The communication pattern is arbitrary and different for different clients but we assume that the time interval between any two consecutive server meetings for any client is bounded by Δ, i.e., $\tau_i^{\text{next}}(t) - \tau_i^{\text{last}}(t) \leq \Delta, \forall t, \forall i$.

Clients can also meet among themselves due to mobility. When two clients meet, they can communicate with each other via, e.g., device-to-device (D2D) communication protocols Asadi et al. (2014). For the ease of analysis, we assume that at each time t, a client can meet at most one other client (the extension to multiple clients is straightforward). Let $\rho \in [0, 1]$ be the probability of a client meeting another client at a time slot. When $\rho = 0$, clients do not meet with each other, thus degenerating to the conventional case. We assume that the client-server meetings remain unchanged regardless of the value of ρ. Please see the use case in the supplementary material for the justification of the system assumptions.

Asynchronous Federated Learning. Because FL cannot be executed synchronously in our setting, we consider an asynchronous FL algorithm similar to Avdiukhin & Kasiviswanathan (2021); Basu et al. (2019) under arbitrary communication patterns. Note, however, that client meetings due to mobility have not come into the picture yet.

Local Model Update. For any client i, when it meets the server at t, it downloads the current global model x^t. The client then uses $x_i^t = x^t$ as the initial model to train a new local model using its own local dataset until it meets the server again. This is done by using a mini-batch SGD method:

$$x_i^{s+1} = x_i^s - \eta g_i^s, \forall s = t, ..., \tau_i^{\text{next}}(t) - 1 \tag{2}$$

where $g_i^s = \nabla F_i(x_i^s, \zeta_i^s)$ is the stochastic gradient on a randomly drawn mini-batch ζ_i^s and η is the learning rate. Here, we assume that a client performs one step SGD at each time slot to keep the notations simple. Let $m_i^t \in \mathbb{R}^d$ be the cumulative local updates (CLU) of client i at time t since its last meeting with the server, which is updated recursively as follows:

$$m_i^t = \eta g_i^{t-1}, \text{ if } t = \tau_i^{\text{last}}(t) + 1; \quad m_i^t = m_i^{t-1} + \eta g_i^{t-1}, \ \forall t = \tau_i^{\text{last}}(t) + 2, ..., \tau_i^{\text{next}}(t) \tag{3}$$

Global Model Update. At any t, let $\mathcal{S}^t$ denote the set of clients who meet the server ($\mathcal{S}^t$ may be empty). These clients upload their CLUs to the server who then updates the global model as

$$x^t = x^{t-1} - \frac{1}{N} \sum_{i \in \mathcal{S}^t} m_i^t \tag{4}$$

The updated global model x^t is then downloaded to each client i in $\mathcal{S}^t$, and the client starts its local training with a new initial model $x_i^t = x^t$.

4 FEDMOBILE

In the vanilla asynchronous FL described in the above section, a client can upload its CLU and download the global model only when it meets the server. When the meeting intervals are long, however, this information cannot be exchanged in a timely manner, thus hindering the global training process. To overcome this issue, we take advantage of *mobility* and propose FedMobile that creates indirect client-server communication opportunities, thereby improving the FL convergence speed. In a nutshell, a client i can use another client j as a relay to upload its CLU to the server (or download the global model from the server) if client j's next (or last) server meeting is earlier (or later) than client i's. Thus, the CLU can be passed to the server sooner (or the client can train based on a fresher global model). We call such client j an upload (or download) *relay* for client i.

For now, we assume that a client can only use *at most one* upload relay and *at most one* download relay between any two consecutive server meetings, considering the extra cost incurred due to the client-client communication. However, FedMobile can be easily extended (we discuss it in The General Case section). Next, we describe separately how FedMobile handles uploading and downloading, which are essentially the dual cases to each other.

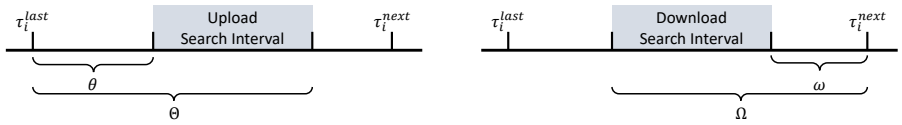


Figure 1: The upload/download search intervals.

4.1 Uploading CLU via Relaying

When to upload via relaying and who should be the relay? Since a client can use the upload relay at most once between its two consecutive server meetings, the timing of uploading via relaying is crucial. If client i uploads too early (i.e., close to τ_i^{last}), then the CLU has little new information since the client has run just a few mini-batch SGD steps. If client i uploads too late (i.e., close to τ_i^{next}), then the CLU will be uploaded to the server late even if a relay is used. To make this balance, FedMobile introduces a notion called *upload search interval* defined by two parameters θ and Θ, where $0 \leq \theta \leq \Theta \leq \Delta$ (See Fig. 1). Client i will upload its CLU via a relay only during the interval $[\tau_i^{\text{last}} + \theta, \tau_i^{\text{last}} + \Theta]$. In addition, not every other client that client i meets during the search interval is qualified. We say that client j is a *semi-qualified* upload relay if $\tau_j^{\text{next}} \leq \tau_i^{\text{last}} + \Theta$, i.e., client j is able to relay client i's CLU to the server before the end of the search interval. We further say that client j is a *qualified* upload relay if in addition $\tau_j^{\text{next}} < \tau_i^{\text{next}}$, i.e., client j can indeed deliver the CLU earlier than client i's own server meeting [1]. FedMobile picks the *first* qualified upload relay during the search interval. Note that it is possible that no qualified upload relay is met, in which case no uploading via relaying is performed.

How to relay the local updates to the server? FedMobile ensures that the same piece of information is delivered to the server *exactly once* by devising a simple yet storage-efficient mechanism. Upon a CLU exchange event at time t between a sender client i and a relay client j:

RESET (by sender): After sending its current CLU m_i^t, sender client i resets its CLU to $m_i^t := 0$

COMBINE (by relay): After receiving m_i^t from client i, client j combines m_j^t and m_i^t to produce a new m_j^t, i.e., $m_j^t := m_j^t + m_i^t$.

In this way, FedMobile essentially offloads the uploading task of m_i^t from client i to client j, who, by our design, has a sooner server meeting time than client i.

Remark: (1) The subsequent local training is not affected at all for both the sender and the relay. That is, for client i (client j), g_i^s (g_j^s) will be the same for all s up to $\tau_i^{\text{next}}(t)$ ($\tau_j^{\text{next}}(t)$) regardless of the CLU exchange. (2) The relay client's storage requirement remains the same because COMBINE produces a CLU of the same size as before. In fact, the relay can combine multiple sender clients' CLUs between two consecutive server meeting times (these exchanges occur at different time slots), yet the storage requirement is the same. (3) A client can be both a sender and a relay between two consecutive server meeting times. If a client i has already received the CLU(s) from some other client(s) before it sends m_i^t to client j, then m_i^t actually also contains CLU(s) of other clients.

Remark: For higher communication efficiency and/or better privacy protection, the clients may send an altered CLU to the relay for uploading. We discuss this extension and provide its convergence analysis in the supplementary material.

4.2 Downloading Global Model via Relaying

When to download via relaying and who should be the relay? Similar to the uploading CLU case, when to download a global model via relaying also involves a trade-off. FedMobile introduces a *download search interval* to make this balance, which is defined by two parameters ω and Ω, where $0 \leq \omega \leq \Omega \leq \Delta$ (see Fig. 1). Given client i's next server meeting time τ_i^{next}, client i will only download a new global model via relaying during the search interval $[\tau_i^{\text{next}} - \Omega, \tau_i^{\text{next}} - \omega]$. Similarly, we say that client j is a *semi-qualified* download relay if $\tau_j^{\text{last}} \geq \tau_i^{\text{next}} - \Omega$, i.e., client j's global model is less than Ω time slots older than client i's next global model directly from the server.

[1] It is possible that $\tau_i^{\text{last}} + \Theta \geq \tau_i^{\text{next}}$ due to the fixed value of Θ, so a semi-qualified upload relay is not necessarily qualified.

We further say that client j is a *qualified* download relay if in addition $\tau_j^{\text{last}} > \tau_i^{\text{last}}$, i.e., client j's global model is indeed fresher than client i's current global model that it received directly from the server. FedMobile picks the *first* qualified download relay during the search interval. Again, it is possible that no qualified download relay is met, in which case no downloading via relaying occurs.

How to relay the global model to the client? To be able to relay a global model to other clients, every client keeps a copy of the most recent global model that it received (from either the server or another client). We denote this copy for client j by $x^{\psi_j(t)}$, where $\psi_j(t)$ is the time version (or timestamp) of the global model. Upon a global model exchange at time t between a receiver client i and a relay client j:

REPLACE (by receiver): After receiving $x^{\psi_j(t)}$ from client j, client i replaces its local model with $x^{\psi_i(t)}$, i.e., $x_i^t := x^{\psi_j(t)}$, and resumes the local training steps.

Remark: (1) Client i also replaces its global model copy with $x^{\psi_j(t)}$ since it is a fresher version by our design. Thus, $\psi_i(t)$ is updated to $\psi_j(t)$. (2) An alternative to the current downloading scheme is that relay client j simply sends its current local model x_j^t to client i, who then replaces its current local model x_i^t with x_j^t, i.e., $x_i^t := x_j^t$. In this way, the clients do not have to keep a copy of the most recent global model, thereby reducing the stored data. The convergence analysis is not affected by this change and the same convergence bound can be proved (see the proof of Theorem 1).

5 Convergence Analysis

Our convergence analysis will utilize the following standard assumptions.

Assumption 1 (Lipschitz Smoothness)**.** *There exists a constant $L > 0$ such that $\|\nabla f_i(x) - \nabla f_i(y)\| \leq L\|x - y\|$, $\forall x, y \in \mathbb{R}^d$ and $\forall i = 1, ..., N$.*

Assumption 2 (Unbiased Local Gradient Estimate)**.** *The local gradient estimate is unbiased, i.e., $\mathbb{E}_\zeta F_i(x, \zeta) = \nabla f_i(x)$, $\forall x$ and $\forall i = 1, \cdots, N$.*

Assumption 3 (Bounded Variance)**.** *There exists a constant $\sigma > 0$ such that $\mathbb{E}[\|\nabla F_i(x, \zeta_i) - \nabla f_i(x)\|^2] \leq \sigma^2$, $\forall x \in \mathbb{R}^d$ and $\forall i = 1, ..., N$.*

Assumption 4 (Bounded Second Moment)**.** *There exists a constant $G > 0$ such that $\mathbb{E}[\|\nabla F_i(x, \zeta_i)\|^2] \leq G^2$, $\forall x \in \mathbb{R}^d$ and $\forall i = 1, ..., N$.*

The *real sequence* of the global model is calculated as

$$x^t = x^0 - \frac{1}{N} \sum_{i=1}^{N} \sum_{s=0}^{\phi_i(t)} \eta g_i^s, \quad \forall t \tag{5}$$

where we define $\phi_i(t)$ to be the time slot up to when all corresponding gradients of client i have been received at time t. In the vanilla asynchronous FL case, $\phi_i(t)$ is simply $\tau_i^{\text{last}}(t) - 1$. In FedMobile, typically $\phi_i(t) > \tau_i^{\text{last}}(t) - 1$ because more information can be uploaded earlier than t due to relaying.

We also define the *virtual sequence* of the global model, which is achieved in the imaginary ideal case where all local gradients are uploaded to the server instantly at every slot,

$$v^t = x^0 - \frac{1}{N} \sum_{i=1}^{N} \sum_{s=0}^{t-1} \eta g_i^s, \quad \forall t \tag{6}$$

First, we bound the difference $(t-1) - \phi_i(t)$, which characterizes how much CLU information of client i is missing compared to the virtual sequence.

Lemma 1. *Assuming at least one semi-qualified upload relay client exists in every upload search interval, then we have $(t-1) - \phi_i(t) \leq \max\{\Delta - \theta, \Theta\} \triangleq C(\theta, \Theta; \Delta), \forall i = 1, ..., N, \forall t$.*

Next, we bound the difference $t - \psi_i(t)$, which characterizes the version difference between the current global model and client i's copy of the global model.

Lemma 2. *Assuming at least one semi-qualified download relay exists in every download search interval, we have $t - \psi_i(t) \leq \max\{\Delta - \omega, \Omega\} \triangleq D(\omega, \Omega; \Delta), \forall i = 1, ..., N, \forall t$.*

The following lemma then bounds the model differences in the real sequence and the virtual sequence.

Lemma 3. *The difference of the real global model and the virtual global model is bounded as follows*

$$\mathbb{E}\left[\|v^t - x^t\|^2\right] \leq C^2(\delta, \Theta; \Delta)\eta^2 G^2 \tag{7}$$

For each client i, the difference of its local model and the virtual global model is bounded as follows

$$\mathbb{E}\left[\|v^t - x_i^t\|^2\right] \leq 3(2D^2(\omega, \Omega; \Delta) + C^2(\theta, \Theta; \Delta))\eta^2 G^2 \tag{8}$$

Now, we are ready to bound the convergence of the real model sequence achieved in FedMobile.

Theorem 1. *Assuming at least one semi-qualified upload (download) relay client exists in every upload (download) search interval, by setting $\eta \leq 1/L$, after T time slots, we have*

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left[\|\nabla f(x^t)\|^2\right] \leq \frac{4}{\eta T}\left(f(x^0) - f^*\right) + 4(3D^2(\omega, \Omega; \Delta) + 2C^2(\delta, \Theta; \Delta))L^2\eta^2 G^2 + \frac{2L\eta\sigma^2}{N} \tag{9}$$

Remark: The convergence bound established in Theorem 1 contains three parts. The first part diminishes as T approaches infinity. Both the second and third terms are constants for a constant η, with the second term depending on our algorithm parameters δ, Θ, ω and Ω. For a given final step T, one can set $\eta = \frac{\sqrt{N}}{L\sqrt{T}}$ so that the bound becomes

$$\frac{4L}{\sqrt{NT}}\left(f(x^0) - f^*\right) + \frac{4N}{T}(3D^2(\omega, \Omega; \Delta) + 2C^2(\delta, \Theta; \Delta))G^2 + \frac{2\sigma^2}{\sqrt{NT}} \tag{10}$$

Furthermore, if $T \geq N^3$, the above bound recovers the same $O(\frac{1}{\sqrt{NT}})$ convergence rate of the classic synchronous FL Yu et al. (2019) [2]. Next, we discuss the above convergence result in more detail and investigate how the mobility affects the convergence. To this end, we consider for now a fixed client-client meeting rate ρ, and investigate how the convergence result depends on our algorithm parameters.

Bound Optimization. Since the convergence bound can be improved by lowering $C(\theta, \Theta; \Delta)$ and $D(\omega, \Omega; \Delta)$, we first investigate them as functions of the algorithm parameters.

Proposition 1. *(1) $C(\theta, \Theta; \Delta)$ is non-decreasing in Θ and non-increasing in θ. Moreover, $\forall\theta, \Theta$, $C(\theta, \Theta; \Delta) \geq \frac{\Delta}{2}$, and the lower bound is attained by choosing $\theta = \Theta = \frac{\Delta}{2}$. (2) $D(\omega, \Omega; \Delta)$ is non-decreasing in Ω and non-increasing in ω. Moreover, $\forall\omega, \Omega$, $D(\omega, \Omega; \Delta) \geq \frac{\Delta}{2}$, and the lower bound is attained by choosing $\omega = \Omega = \frac{\Delta}{2}$.*

Proposition 1 implies that one should use shorter search intervals, namely $\Theta - \theta$ and $\Omega - \omega$, to lower $C(\theta, \Theta; \Delta)$ and $D(\omega, \Omega; \Delta)$. However, the best C and D are no smaller than $\frac{\Delta}{2}$, which are achieved by choosing $\theta = \Theta = \frac{\Delta}{2}$ and $\omega = \Omega = \frac{\Delta}{2}$. In other words, a short upload search interval around the time $\tau_i^{\text{last}} + \frac{\Delta}{2}$ and a short download search interval around the time $\tau_i^{\text{next}} - \frac{\Delta}{2}$ improve the FL convergence. This suggests that the best timing for uploading is at exactly $\tau_i^{\text{last}} + \frac{\Delta}{2}$ and the best timing for downloading is at exactly $\tau_i^{\text{next}} - \frac{\Delta}{2}$, which are neither too early or too late in both cases.

Probability of Meeting a Semi-Qualified Relay. The convergence bound in Theorem 1, however, is obtained under the assumption that a client can meet at least one semi-qualified upload (download) relay client in each upload (download) search interval, which may not always hold. How easily a client can find a semi-qualified relay client depends on the client-client meeting rate. In the VANET example, this rate depends on the D2D communication range.

Let $Q_u(\theta, \Theta)$ (and $Q_d(\omega, \Omega)$) be the probability that at least one semi-qualified uploading (download) relay is met in an uploading (download) search interval with length $\Theta - \theta$ (and $\Omega - \omega$). They are characterized as follows

[2] There is a subtle difference because T in the asynchronous setting is the number of time slots while in the synchronous setting T is the number of rounds. However, they differ by at most a factor of Δ.

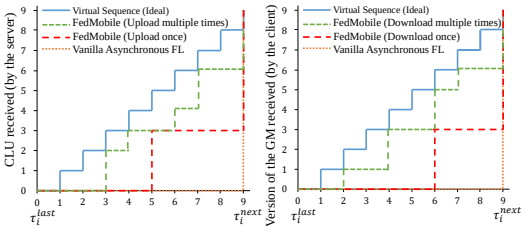
Proposition 2. *Assuming sufficiently many clients in the system and that clients meet each other uniformly randomly. Let $P_{int}(\cdot)$ be the distribution of server meeting intervals. Then $Q_u(\theta, \Theta) = 1 - \prod_{t=0}^{\Theta-\theta}(1 - \rho q_u(\Theta - \theta - t))$ and $Q_d(\omega, \Omega) = 1 - \prod_{t=0}^{\Omega-\omega}(1 - \rho q_d(t))$, where $q_u(\cdot)$ and $q_d(\cdot)$ are distributions computed based on $P_{int}(\cdot)$.*

Proposition 2 states an intuitive result that one should use a larger search interval to increase the probability of meeting a semi-qualified relay. This, however, is not desirable for lowering the convergence bound according to Proposition 1. This is exactly where mobility can help improving the FL convergence: by increasing the client-client meeting rate ρ, a shorter search interval $\Theta - \theta$ (or $\Omega - \omega$) can be used to achieve the same Q_u (or Q_d), but a smaller C (or D) is obtained. In fact, by relaxing the constraint that a client can only meet one other client at a time slot, with sufficiently many clients in the system, both Q_u and Q_d can approach 1 even if the search interval is just one slot.

6 The General Case

FedMobile can be easily extended to allow multiple upload (download) relay communications between any two consecutive server meetings. We will still have the upload (download) search interval, and FedMobile will simply pick the first K qualified upload (download) relays to perform the upload (download) relay communications. The exact mechanisms of how uploading (downloading) is performed will be slightly changed to handle out-of-order information and avoid redundant information. Our convergence analysis still holds correctly for this generalized case, although the bound may become looser compared to the actual performance.

Figure 2 illustrates the key ideas behind FedMobile. Figure 2(a) plots hypothetical sequences of CLUs received by the server from a representative client between two consecutive server meetings. The virtual sequence is the ideal case where each local stochastic gradient g_i^t is uploaded to the server instantly after it is computed. Therefore, the received CLU curve is a steady staircase. In the vanilla asynchronous FL, the client uploads the CLU only when it meets the server. Therefore there is a big jump at the next server meeting time but it is all 0 before. FedMobile with at most one upload relaying allows some local stochastic gradient information to be received by the server earlier. The generalized FedMobile allows more CLUs to be relayed and received by the server at earlier time slots. One can imagine that when ρ is large enough, it becomes much easier for the client to find qualified relays that can quickly upload CLUs to the server at every time slot. Therefore, the received CLU curve approaches that in the ideal case. Similarly, Figure 2(b) plots the hypothetical sequence of the global models received by the client.



(a) Uploading (b) Downloading

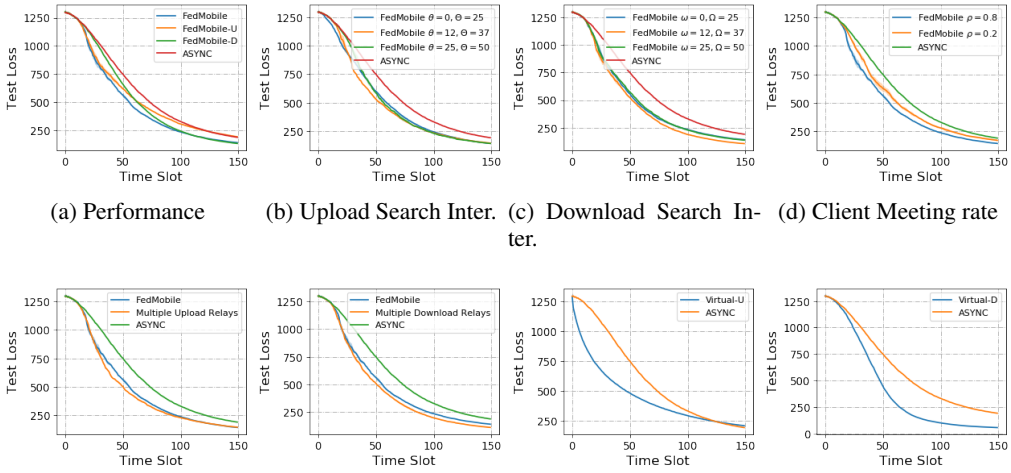
Figure 2: Hypothetical sequences of (a) CLUs received by the server and (b) global model received by the client.

7 Experiments

Datasets and Models. We conduct experiments on a synthetic dataset and two real-world datasets, i.e., FMNIST Xiao et al. (2017) and CIFAR10 Krizhevsky et al. (2009). **Synthetic dataset**: The synthetic dataset is generated on a least-squares linear regression problem. Each data sample has a 200 dimensional feature vector and its real-valued label is calculated as the vector product of the feature and an underlying linear weight plus a 0-mean Gaussian noise. The FL system has 50 clients with each client having 40 data samples. **FMNIST**: The FL system has 50 clients with each client having 400 data samples. The data samples are randomly allocated to the clients according to a widely used method to generate non-i.i.d. datasets for FL Chen & Chao (2020). We use LeNet LeCun et al. (1998) as the backbone model. **CIFAR10**: The FL system has 50 clients with each client having 600 data samples. The data allocation method is the same as that used for FMNIST. We use ResNet-9 He et al. (2016) as the backbone model.

Benchmarks. The following benchmarks are considered in our experiments. (1) **ASYNC**: This is the state-of-the-art asynchronous FL method proposed in Avdiukhin & Kasiviswanathan (2021) to handle arbitrary communication patterns. (2) **Virtual-U**: This method assumes that each client can upload its local updates immediately to the server at every slot via an imaginary channel but can only download the global model at their actual communication slots. (3) **Virtual-D**: This method assumes that each client can download the global model from the server at every slot via an imaginary channel but can only upload their CLUs at their actual communication slots. We also consider several variations of FedMobile. (1) **FedMobile**: This is the proposed algorithm combining both upload and download relay communications. (2) **FedMobile-U**: This is the proposed algorithm with only upload relaying. (3) **FedMobile-D**: This is the proposed algorithm with only download relaying.

Communication Patterns. We simulate two asynchronous communication patterns with the server. **Fixed-Interval**. Client $i \in \{1, ..., 50\}$ communicates with the server at slots $i + 50n, \forall n = 0, 1, 2, ...$ **Random-Interval**. Client $i \in \{1, ..., 50\}$ has the first communication with the server at slot i, and then communicates with the server with a random interval between 30 and 50 slots. For communications among the clients, clients randomly meet with each other. Details can be found in the supplementary material.



(a) Performance (b) Upload Search Inter. (c) Download Search Inter. (d) Client Meeting rate

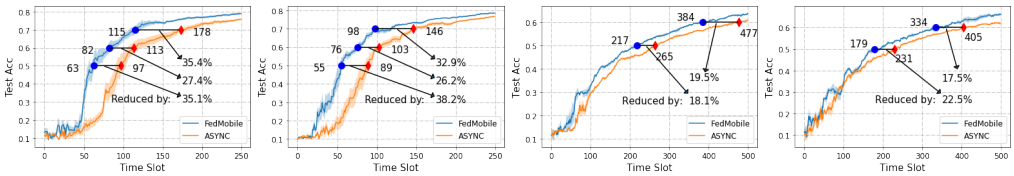
(e) Multiple Uploads (f) Multiple Downloads (g) Virtual Upload (h) Virtual Downloads

Figure 3: Results on the synthetic data.

Results on Synthetic Data: Fig. 3 reports the results (averaged over three repeats) on the synthetic data with a fixed-interval communication pattern. The detailed configurations can be found in the supplementary material. Fig. 3(a) compares **FedMobile** and its variations with **ASYNC** in terms of the test loss. As can been seen, incorporating either upload or download relaying into asynchronous FL improves the FL performance, and a further improvement can be achieved when uploading and downloading are combined. Fig. 3(b)/(c) illustrates the impact of upload/download search interval on the FL convergence. In both cases, the best search timing is around the middle point of the two consecutive server meeting times, confirming our theoretical analysis in Theorem 1 and Proposition 1. Fig. 3(d) shows the impact of ρ on the FL convergence. As predicted by our analysis, a higher ρ in the system improves FL convergence since more timely relay communication opportunities are created. In Figs. 3(e)(f), we allow clients to use multiple relays to create more communication opportunities with the server whenever possible. The results show that further improvement can indeed be achieved. We also conduct experiments on the ideal relaying scenarios, namely **Virtual-U** and **Virtual-D**, to illustrate what can be achieved in the ideal case. The results verify our hypothesis that more communication opportunities with the server benefits convergence. It is also interesting to note that virtual uploading/downloading has a more significant impact on the early/late FL slots, suggesting that an adaptive design may better balance the FL performance and the resource cost.

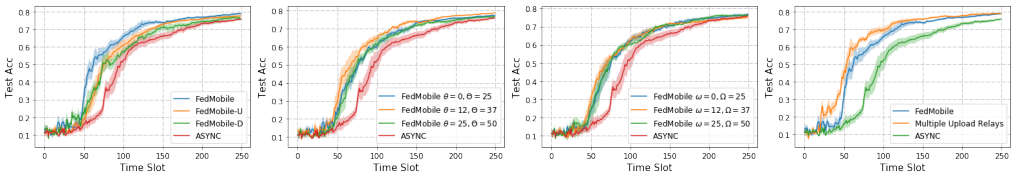
Results on Real-World Datasets: We now report the results (averaged over three repeats) on real-world datasets. Fig. 4 shows that **FedMobile** achieves a better convergence speed than **ASYNC** on FMNIST and CIFAR10 under both the Fixed-Interval and Random-Interval settings. For a specific

example, to achieve 70% accuracy on FMNIST under the fixed-interval setting, **ASYNC** needs about 178 slots while **FedMobile** only needs 115 slots. The required time slot decreases by 35.4%, which demonstrates the significant improvement of **FedMobile**. Fig. 5 and Fig. 6 further show the experiment results on FMNIST under the fixed-interval setting and the random-interval setting, respectively. The results largely resemble those in the synthetic data case. In particular, Fig. 5(a) and Fig. 6(a) demonstrate that while upload and download relay communications individually can improve the FL convergence performance, combining them results in an additional benefit. Fig. 5 (b)(c) show that the best timing of upload and download relaying should be neither too early nor too late. However, the difference between the three download search intervals in Fig. 5(c) is less distinguishable. Fig. 5(d) and Fig. 6(b) show that using multiple upload relay communications further improves the FL convergence performance. However, we observed in our experiments that this is not the case with using multiple download relay communications. This is likely due to the complexity of real-world datasets, which causes the assumptions for our theoretical analysis to be violated. This represents a limitation of our current analysis and requires further investigation. We defer related experiment results to the supplemental material. Finally, Fig. 6(c) confirms that a higher client-client meeting rate (e.g., larger D2D communication range of mobile devices) of the system improves the convergence of asynchronous FL with the help of **FedMobile**.



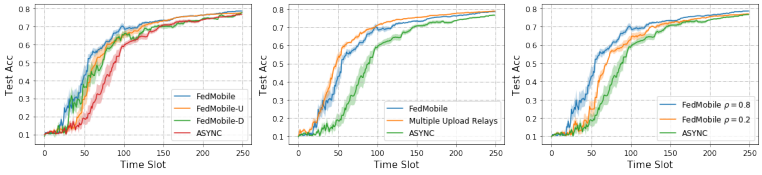
(a) FMNIST (Fixed) (b) FMNIST (Random) (c) CIFAR10 (Fixed) (d) CIFAR10 (Random)

Figure 4: Performance Comparison on FMNIST and CIFAR10.



(a) Upload/Download (b) Upload Search Inter. (c) Download Search Inter. (d) Multiple Upload

Figure 5: Experiments on FMNIST under the Fixed-Interval setting.



(a) Upload/Download (b) Multiple Upload (c) Client Meeting Rate

Figure 6: Experiments on FMNIST under the Random-Interval setting.

8 Conclusion

This paper offers a new perspective on how to design asynchronous FL algorithms in practical systems where client-server communication is not always available. We advocate the role of client mobility, and hence the resulting random client-client communication opportunities, in the timely information exchange for the model training in asynchronous FL. The experiment results are mostly consistent with our theoretical results and demonstrate that mobility indeed improves the convergence of asynchronous FL. We believe that FedMobile can promote distributed machine learning, especially FL, in many real-world systems such as mobile gaming, mobile sensing and smart vehicular networks.

REFERENCES

Dan Alistarh, Demjan Grubic, Jerry Li, Ryota Tomioka, and Milan Vojnovic. Qsgd: Communication-efficient sgd via gradient quantization and encoding. *Advances in neural information processing systems*, 30, 2017.

Arash Asadi, Qing Wang, and Vincenzo Mancuso. A survey on device-to-device communication in cellular networks. *IEEE Communications Surveys & Tutorials*, 16(4):1801–1819, 2014.

Dmitrii Avdiukhin and Shiva Kasiviswanathan. Federated learning under arbitrary communication patterns. In *International Conference on Machine Learning*, pp. 425–435. PMLR, 2021.

Debraj Basu, Deepesh Data, Can Karakus, and Suhas Diggavi. Qsparse-local-sgd: Distributed sgd with quantization, sparsification and local computations. *Advances in Neural Information Processing Systems*, 32, 2019.

Zheng Chai, Yujing Chen, Ali Anwar, Liang Zhao, Yue Cheng, and Huzefa Rangwala. Fedat: a high-performance and communication-efficient federated learning system with asynchronous tiers. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 1–16, 2021.

Hong-You Chen and Wei-Lun Chao. Fedbe: Making bayesian model ensemble applicable to federated learning. In *International Conference on Learning Representations*, 2020.

Jianmin Chen, Xinghao Pan, Rajat Monga, Samy Bengio, and Rafal Jozefowicz. Revisiting distributed synchronous sgd. *arXiv preprint arXiv:1604.00981*, 2016.

Tianyi Chen, Georgios Giannakis, Tao Sun, and Wotao Yin. Lag: Lazily aggregated gradient for communication-efficient distributed learning. *Advances in Neural Information Processing Systems*, 31, 2018.

Yujing Chen, Yue Ning, Martin Slawski, and Huzefa Rangwala. Asynchronous online federated learning for edge devices with non-iid data. In *2020 IEEE International Conference on Big Data (Big Data)*, pp. 15–24. IEEE, 2020.

Wei Dai, Yi Zhou, Nanqing Dong, Hao Zhang, and Eric Xing. Toward understanding the impact of staleness in distributed machine learning. In *International Conference on Learning Representations*, 2018.

Abderrahim Guerna, Salim Bitam, and Carlos T Calafate. Roadside unit deployment in internet of vehicles systems: a survey. *Sensors*, 22(9):3190, 2022.

Yuanxiong Guo, Ying Sun, Rui Hu, and Yanmin Gong. Hybrid local sgd for federated learning with heterogeneous communications. In *International Conference on Learning Representations*, 2021.

Stefan Hadjis, Ce Zhang, Ioannis Mitliagkas, Dan Iter, and Christopher Ré. Omnivore: An optimizer for multi-device deep learning on cpus and gpus. *arXiv preprint arXiv:1606.04487*, 2016.

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.

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.

Ahmed Khaled, Konstantin Mishchenko, and Peter Richtárik. Tighter theory for local sgd on identical and heterogeneous data. In *International Conference on Artificial Intelligence and Statistics*, pp. 4519–4529. PMLR, 2020.

Jakub Konečnỳ, H Brendan McMahan, Daniel Ramage, and Peter Richtárik. Federated optimization: Distributed machine learning for on-device intelligence. *arXiv preprint arXiv:1610.02527*, 2016.

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

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

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

Xiang Li, Kaixuan Huang, Wenhao Yang, Shusen Wang, and Zhihua Zhang. On the convergence of fedavg on non-iid data. In *International Conference on Learning Representations*, 2019.

Xingyu Li, Zhe Qu, Bo Tang, and Zhuo Lu. Stragglers are not disaster: A hybrid federated learning algorithm with delayed gradients. *arXiv preprint arXiv:2102.06329*, 2021.

Wei Liu, Li Chen, Yunfei Chen, and Wenyi Zhang. Accelerating federated learning via momentum gradient descent. *IEEE Transactions on Parallel and Distributed Systems*, 31(8):1754–1766, 2020.

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.

Ioannis Mitliagkas, Ce Zhang, Stefan Hadjis, and Christopher Ré. Asynchrony begets momentum, with an application to deep learning. In *2016 54th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pp. 997–1004. IEEE, 2016.

Takayuki Nishio and Ryo Yonetani. Client selection for federated learning with heterogeneous resources in mobile edge. In *ICC 2019-2019 IEEE international conference on communications (ICC)*, pp. 1–7. IEEE, 2019.

Virginia Smith, Chao-Kai Chiang, Maziar Sanjabi, and Ameet S Talwalkar. Federated multi-task learning. *Advances in neural information processing systems*, 30, 2017.

Marten van Dijk, Nhuong V Nguyen, Toan N Nguyen, Lam M Nguyen, Quoc Tran-Dinh, and Phuong Ha Nguyen. Asynchronous federated learning with reduced number of rounds and with differential privacy from less aggregated gaussian noise. *arXiv preprint arXiv:2007.09208*, 2020.

Jianyu Wang, Vinayak Tantia, Nicolas Ballas, and Michael Rabbat. Slowmo: Improving communication-efficient distributed sgd with slow momentum. In *International Conference on Learning Representations*, 2019.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

Hong Xing, Osvaldo Simeone, and Suzhi Bi. Decentralized federated learning via sgd over wireless d2d networks. In *2020 IEEE 21st International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, pp. 1–5. IEEE, 2020.

Haibo Yang, Minghong Fang, and Jia Liu. Achieving linear speedup with partial worker participation in non-iid federated learning. In *International Conference on Learning Representations*, 2020.

Hao Yu, Sen Yang, and Shenghuo Zhu. Parallel restarted sgd with faster convergence and less communication: Demystifying why model averaging works for deep learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 5693–5700, 2019.

Xueqing Zhang, Yanwei Liu, Jinxia Liu, Antonios Argyriou, and Yanni Han. D2d-assisted federated learning in mobile edge computing networks. In *2021 IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1–7. IEEE, 2021.

Shuxin Zheng, Qi Meng, Taifeng Wang, Wei Chen, Nenghai Yu, Zhi-Ming Ma, and Tie-Yan Liu. Asynchronous stochastic gradient descent with delay compensation. In *International Conference on Machine Learning*, pp. 4120–4129. PMLR, 2017.

A USE CASE OF FEDMOBILE

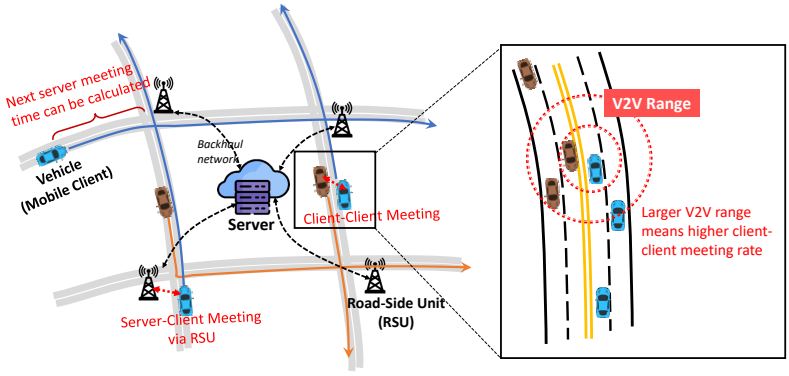


Figure 7: Illustration of FedMobile in VANET.

In this section, we provide a motivating use case of FedMobile, which is the Vehicular Ad Hoc Network (VANET) (See Fig. 7 for an illustration). Consider a road network where many vehicles (as mobile clients) are moving and performing FL (e.g., for traffic prediction) using on-board computing power. A number of road-side units (RSUs) are deployed in the road network which the vehicles can connect to via wireless. All relevant RSUs are connected to a wired backhaul network where the parameter server resides. In a typical deployment, the RSUs are only deployed in certain parts of the road network (e.g., some road intersections) and do not cover the entire road network Guerna et al. (2022). Therefore, the vehicles (clients) can communicate with the server only when they enter the communication range of a RSU. Because vehicles (clients) have different speeds and routes, their meeting times with servers (via the RSUs) are naturally asynchronous.

In the VANET use case, vehicles (clients) often have a pre-determined route and move at a roughly constant speed. Thus, it is easy to calculate the next server meeting given the next RSU location and the vehicle trajectory. If a vehicle's moving speed is fixed, then the client-server meetings do not change (because distances between RSUs cannot be changed). A client-client meeting occurs if two clients enter the device-to-device communication (e.g., Wi-Fi Direct or LTE Direct) range of each other. Thus, by increasing the communication range (e.g., by using a larger transmission power), then more client-client meetings can occur while the client-server meetings stay the same. Thus, ρ is a parameter that models the client-client meeting rates in this case. A smaller client-client communication range results in a smaller ρ while a larger client-client communication range results in a larger ρ.

Another example use case is FL for post-disaster assessment and response. Consider a post-disaster network where the communication infrastructure is largely disrupted. To assist quick disaster assessment and response, FL can be used to collect data over the disaster area and perform spatial and temporal analysis. However, because of the disrupted communication infrastructure, mobile clients can sedomly communicate with the server via the limited number of functioning base stations/access points. FedMobile can be used in this scenario to improve the FL convergence speed leveraging client mobility and client-client communications.

B EXPERIMENT DETAILS

We implement FL simulation on the Pytorch framework and perform the model training on one Geforce RTX 3080 GPU. The code files of data generation and the proposed method are attached in the supplemental material. All experiment results are averaged over 3 repeats.

The real datasets (FMNIST and CIFAR10) which we utilized in experiment section are both public. Both of them are licensed under MIT License.

To simulate communications among the clients, at each time slot, we uniformly sample ρN clients over total N clients and randomly construct $\frac{\rho N}{2}$ client pairs. Any two clients in the same pair are simulated to communicate.

Synthetic dataset: The synthetic dataset is generated on a least-squares linear regression problem. Each data sample has a 200 dimensional feature vector and its real-valued target is calculated as the vector product of the feature and an underlying linear weight plus a 0-mean Gaussian noise. The FL system has 50 clients with each client having 40 data samples. The experiment configuration details are shown in Table 1.

Table 1: Synthetic Dataset Setup

Hyper-parameters	Values
Learning rate	0.01 (initially)
Decay rate	0.99 (until lr = 0.0001)
Train batch size	128
Test batch size	128
Number of clients	50
Each client train data	40
Communication Patterns	Fixed-Interval
Number of training time slots	150
Default Upload Parameters	$\theta = 10$, $\Theta = 40$
Default Download Parameters	$\omega = 5$. $\Omega = 25$

FMNIST(Fashion-MNIST): The FL system has 50 clients with each client having 400 data samples. We utilize the Dirichlet function ($\alpha = 0.3$) which is typically utilized to simulate the level of non-IID in FL. We use LeNet as the backbone model. The training configuration details are summarized in Table 2.

Table 2: Fashion-MNIST Setup

Hyper-parameters	Values
Learning rate	0.1 (initially)
Decay rate	0.99 (until lr = 0.001)
Train batch size	128
Test batch size	128
Number of clients	50
Each client train data	400
Communication Patterns	Fixed/Random-Interval
Number of training time slots	250
Default Upload Parameters	$\theta = 10$, $\Theta = 40$
Default Download Parameters	$\omega = 5$. $\Omega = 25$

CIFAR10: The FL system has 50 clients with each client having 600 data samples. The data allocation method is the same as that used for FMNIST. We use ResNet-9 as the backbone model. The architecture of ResNet-9 is given in Table 4.The training configuration details are summarized in Table 3.

Table 3: CIFAR10 Setup

Hyper-parameters	Values
Learning rate	0.01
Decay rate	N/A
Train batch size	128
Test batch size	128
Number of clients	50
Each client train data	600
Communication Patterns	Fixed/Random-Interval
Number of training time slots	500
Default Upload Parameters	$\theta = 10$, $\Theta = 40$
Default Download Parameters	$\omega = 5$. $\Omega = 25$

Table 4: Network Architecture of ResNet-9

Layer	Filter Shape	Stride	Output
Input	N/A	N/A	$32 \times 32 \times 3$
Conv 1	$3 \times 3 \times 3 \times 64$	1	$32 \times 32 \times 64$
Conv 2	$3 \times 3 \times 64 \times 128$	1	$32 \times 32 \times 128$
Pool 1	2×2	2	$16 \times 16 \times 128$
Conv 3	$3 \times 3 \times 128 \times 128$	1	$16 \times 16 \times 128$
Conv 4	$3 \times 3 \times 128 \times 128$	1	$16 \times 16 \times 128$
Conv 5	$3 \times 3 \times 128 \times 256$	1	$16 \times 16 \times 256$
Pool 2	2×2	2	$8 \times 8 \times 256$
Conv 6	$3 \times 3 \times 256 \times 512$	1	$8 \times 8 \times 512$
Pool 3	2×2	2	$4 \times 4 \times 512$
Conv 7	$3 \times 3 \times 512 \times 512$	1	$4 \times 4 \times 512$
Conv 8	$3 \times 3 \times 512 \times 512$	1	$4 \times 4 \times 512$
Pool 4	4×4	4	$1 \times 1 \times 512$
Softmax	512×10	N/A	$1 \times 1 \times 10$

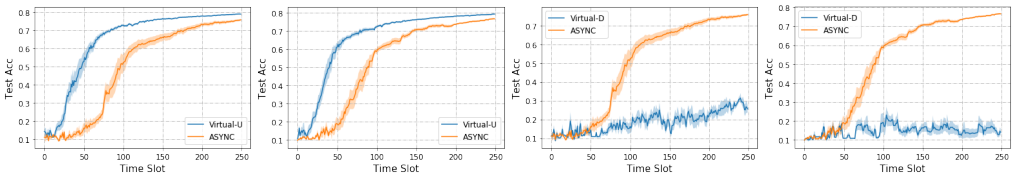
C Additional Experiments

C.1 Virtual-U and Virtual-D

Virtual-U and **Virtual-D** work as the ideal cases for upload relaying and download relaying and hence we conduct experiments to investigate their performance on the real-world dataset. Fig. 8 reports the performance of **Virtual-U** and **Virtual-D** on FMNIST with both fixed and random interval communication patterns. Similar to Fig. 3(g) for the synthetic data, Figs. 8 (a)(b) show that **Virtual-U** can greatly improve the convergence performance. However, different from Fig. 3(h) for the synthetic data, Figs. 8 (c)(d) show that **Virtual-D** fail to converge on the real-world dataset. We conjecture that this is likely due to the complexity of the real-world dataset where some of the assumptions needed for our theoretical analysis do not strictly hold. However, as Figs. 5 and 6 show, using one-time download relaying still has benefits on the convergence even in the real-world dataset.

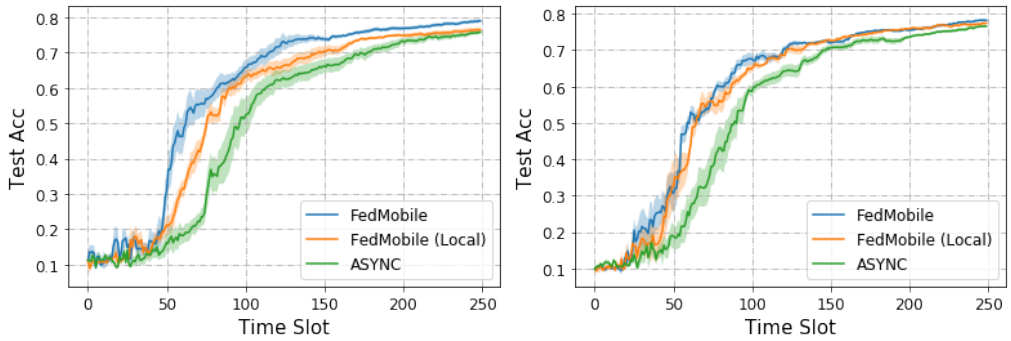
C.2 Alternative Download Relaying Scheme

In Section 4.2, we discussed how to relay the global model to the client. In particular, each client has to keep a copy of the most recent global model that it received (from either the server or another client). Therefore, when needed, the client can relay this global model to another client. We mentioned an alternative download relaying scheme where clients do not have to keep this copy. Instead, the client simply performs the local training on the received global model (from either the server or another client). When needed, the client relays this local model to another client. This way saves the clients storage space. We also proved in Theorem 1 that this alternative download relaying scheme achieves the same convergence bound as the original one. Here, we illustrate its performance via



(a) Virtual-U (Fixed) (b) Virtual-U (Random) (c) Virtual-D (Fixed) (d) Virtual-D (Random)

Figure 8: Virtual-U and Virtual-D on FMNIST.



(a) Alternative download (Fixed) (b) Alternative download (Random)

Figure 9: Alternative download relay scheme.

experiments in Fig. 9. As we can see, the alternative download relaying scheme still outperforms **ASYNC**.

C.3 Download Time Effect

In FMNIST experiments, the difference between three download search interval is not obvious. To better show the impact of download relaying timing, we consider an ideal scenario where each client can virtually download the global model 1/30/49 time slot(s) after its last communication with server. The result in Fig. 10 clearly shows that the best timing of download relaying should be neither too early nor too late.

C.4 Client-Client Meeting Rate

Fig. 6(d) shows the effect of client-client meeting rate under the random-interval communication pattern. Here, we provide the experiment results under the fixed-interval pattern. Fig. 11 shows that the higher client-client meeting rate improves the convergence performance.

C.5 Different Interval Length and Number of Clients

To further validate the proposed method FedMobile, we run additional experiments with different interval length/number of clients. In Fig. 12, holding other training settings unchanged, we extend the fixed-interval to 70 time slots and update FedMobile's parameters ($\theta = 20, \Theta = 60, \omega = 10, \Omega = 30$). In Fig. 13, we increase the number of clients to 100. Both results validate that our proposed method can outperform the baseline method.

C.6 Exponentially Distributed Server Meeting Intervals

We consider the client communication pattern where the server meeting intervals are exponentially distributed with mean being 30 time slots. In addition, we assume that the distribution is truncated

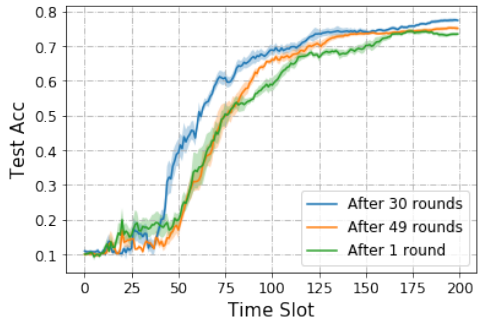


Figure 10: Download Time Effect

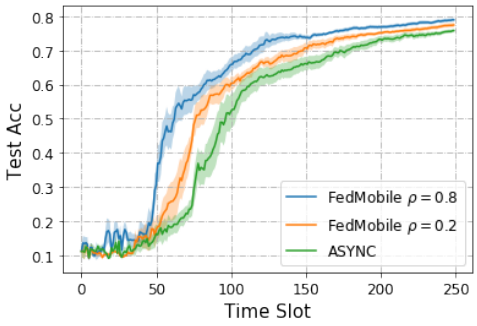


Figure 11: Client-client Meeting Rates (Fixed)

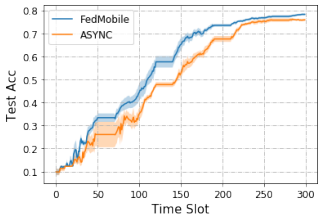


Figure 12: Fixed-interval = 70

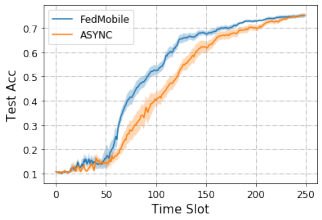


Figure 13: With 100 clients

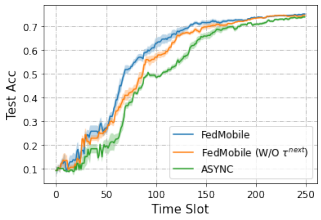


Figure 14: Exponentially distributed interval (Mean = 30, Truncated at 80)

(maximum is 80 time slots) so that the intervals between consecutive server meetings are bounded. We consider both the case where the next server meeting time is known to the clients and the case where the next server meeting time is unknown. In the latter case, the expected value of the exponential distribution is used to estimate their next server meeting time. Fig. 14 shows that FedMobile can still outperform the baseline under exponentially distributed server meeting intervals.

D PROOFS

D.1 PROOF OF LEMMA 1

It is obvious that if the server meeting time interval $\tau_i^{\text{next}}(t) - \tau_i^{\text{last}}(t) \leq \Theta$, then $(t-1) - \phi_i(t) = t - \tau_i^{\text{last}}(t) \leq \Theta$ already holds. Otherwise, for all $t \leq \tau_i^{\text{last}}(t) + \Theta$, then $(t-1) - \phi_i(t) = t - \tau_i^{\text{last}}(t) \leq \Theta$ also holds. Thus, we only need to consider the case $\tau_i^{\text{next}}(t) - \tau_i^{\text{last}}(t) > \Theta$ and for time slot $t > \tau_i^{\text{last}}(t) + \Theta$. In this case, a semi-qualified relay client is also a qualified relay client because

$$\tau_i^{\text{last}}(t) + \Theta < \tau_i^{\text{next}}(t) \tag{11}$$

By the assumption that at least one semi-qualified relay exists in the search interval, at least one qualified relay must exist. This further implies that the qualified relay client is able to upload a CLU before t. Because this CLU contains gradients of client i for at least θ steps since $\tau_i^{\text{last}}(t)$, we have $\phi_i(t) \geq \tau_i^{\text{last}}(t) + \theta - 1$. Therefore,

$$(t-1) - \phi_i(t) = \left(t - \tau_i^{\text{last}}(t)\right) + \left(\tau_i^{\text{last}}(t) - \phi_i(t) - 1\right) \leq \Delta - \theta \tag{12}$$

To summarize the above cases, $(t-1) - \phi_i(t) \leq \max\{\Delta - \theta, \Theta\}$ is established.

D.2 PROOF OF LEMMA 2

Let t' be the meeting time between receiver client i and relay client j. Clearly, $\tau_i^{\text{next}} - \Omega \leq \tau_j^{\text{last}}(t) \leq t' \leq \tau_i^{\text{next}} - \omega$ by the definition of the search interval.

For $t \leq t'$, client i has not met client j yet, so $\psi_i(t) = \tau_i^{\text{last}}(t)$. Therefore,

$$t - \psi_i(t) = t - \tau_i^{\text{last}}(t) \leq t' - \tau_i^{\text{last}}(t) \leq \tau_i^{\text{next}}(t) - \omega - \tau_i^{\text{last}}(t) \leq \Delta - \omega \tag{13}$$

For $t > t'$, client i has met client j, so $\psi_i(t) \geq \tau_j^{\text{last}}(t)$. Therefore,

$$t - \psi_i(t) \leq t - \tau_j^{\text{last}}(t) \leq t - (\tau_i^{\text{next}}(t) - \Omega) \leq \Omega \tag{14}$$

To sum up, $t - \psi_i(t) \leq \max\{\Delta - \omega, \Omega\}$

D.3 Proof of Lemma 3

Consider any time t, by the definition of virtual and real sequences, we have

$$\mathbb{E}\left[\|v^t - x^t\|^2\right] = \mathbb{E}\left[\left\|\frac{1}{N}\sum_{i=1}^{N}\sum_{s=\phi_i(t)}^{t-1}\eta g_i^s\right\|^2\right] \tag{15}$$

$$\leq \frac{\eta^2}{N}\sum_{i=1}^{N}\mathbb{E}\left[\left\|\sum_{s=\phi_i(t)}^{t-1} g_i^s\right\|^2\right] \leq \frac{\eta^2}{N}\sum_{i=1}^{N}((t-1) - \phi_i(t))^2 G^2 \tag{16}$$

$$\leq C^2(\theta, \Theta; \Delta)\eta^2 G^2 \tag{17}$$

where the first $\leq$ is due to the Cauchy-Schwarz inequality, the second $\leq$ is due to both the Cauchy-Schwarz inequality and Assumption 4, and the last $\leq$ is due to Lemma 1.

Consider client i at any time t with the global model version $\psi_i(t)$. Therefore client i is doing the local training steps using the global model $x^{\psi_i(t)}$ as the initial model. Thus, we use $x^{\psi_i(t)}$ as the anchor model to investigate the difference between v^t and x_i^t.

$$\mathbb{E}\left[\left\|v^t - x_i^t\right\|^2\right] = \mathbb{E}\left[\left\|v^t - v^{\psi_i(t)} + v^{\psi_i(t)} - x^{\psi_i(t)} + x^{\psi_i(t)} - x_i^t\right\|^2\right] \tag{18}$$

$$\leq 3\mathbb{E}\left[\left\|v^t - v^{\psi_i(t)}\right\|^2 + \left\|v^{\psi_i(t)} - x^{\psi_i(t)}\right\|^2 + \left\|x^{\psi_i(t)} - x_i^t\right\|^2\right] \tag{19}$$

$$\leq 3\left(\mathbb{E}\left[\left\|\frac{1}{N}\sum_{i=1}^{N}\sum_{s=\psi_i(t)}^{t-1}\eta g_i^s\right\|^2\right] + \mathbb{E}\left[\left\|v^{\psi_i(t)} - x^{\psi_i(t)}\right\|^2\right] + \mathbb{E}\left[\left\|x_i^{t'} - x_i^t\right\|^2\right]\right) \tag{20}$$

$$\leq 3\left(\mathbb{E}\left[\left\|\frac{1}{N}\sum_{i=1}^{N}\sum_{s=\psi_i(t)}^{t-1}\eta g_i^s\right\|^2\right] + \mathbb{E}\left[\left\|v^{\psi_i(t)} - x^{\psi_i(t)}\right\|^2\right] + \mathbb{E}\left[\left\|x_i^{\psi_i(t)} - x_i^t\right\|^2\right]\right) \tag{21}$$

$$\leq 3\left(D^2(\omega, \Omega; \Delta)\eta^2 G^2 + C^2(\theta, \Theta; \Delta)\eta^2 G^2 + D^2(\omega, \Omega; \Delta)\eta^2 G^2\right) \tag{22}$$

$$= 3(2D^2(\omega, \Omega; \Delta) + C^2(\theta, \Theta; \Delta))\eta^2 G^2 \tag{23}$$

Note that $t - t' \leq t - \psi_i(t)$ always holds. If the client i has not met its download relay, $t' = \psi_i(t)$. Otherwise (the client i has met its download relay), t' is the meeting time and therefore, $t' > \psi_i(t)$. Note that this bound holds even if a relay client j simply sends its local model x_j^s to the client i at some time $t \leq \psi(t) \leq s \leq t$.

D.4 Proof of Theorem 1

We first analyze the convergence of the virtual global model sequence. By the smoothness of $f(x)$ (Assumption 1), we have

$$\mathbb{E}[\|f(v^{t+1})\|] \leq \mathbb{E}[\|f(v^t)\|] + \mathbb{E}[\langle \nabla f(v^t), v^{t+1} - v^t\rangle] + \frac{L}{2}\mathbb{E}[\|v^{t+1} - v^t\|^2] \tag{24}$$

The second term on the right-hand side of equation 24 can be expressed as follows,

$$\mathbb{E}[\langle \nabla f(v^t), v^{t+1} - v^t \rangle] \tag{25}$$

$$= -\eta \mathbb{E}[\langle \nabla f(v^t), \frac{1}{N}\sum_{i=1}^{N} g_i^t \rangle] = -\eta \mathbb{E}[\langle \nabla f(v^t), \frac{1}{N}\sum_{i=1}^{N} \nabla f_i(x_i^t) \rangle] \tag{26}$$

$$= -\frac{\eta}{2}\mathbb{E}\left[\|\nabla f(v^t)\|^2 + \|\frac{1}{N}\sum_{i=1}^{N}\nabla f_i(x_i^t)\|^2 - \|\nabla f(v^t) - \frac{1}{N}\sum_{i=1}^{N}\nabla f_i(x_i^t)\|^2\right] \tag{27}$$

$$= -\frac{\eta}{2}\mathbb{E}\left[\|\nabla f(v^t)\|^2\right] - \frac{\eta}{2}\mathbb{E}\left[\|\frac{1}{N}\sum_{i=1}^{N}\nabla f_i(x_i^t)\|^2\right] + \frac{\eta}{2}\mathbb{E}\left[\|\nabla f(v^t) - \frac{1}{N}\sum_{i=1}^{N}\nabla f_i(x_i^t)\|^2\right] \tag{28}$$

The third term on the right-hand side of equation 24 can be bounded as follows,

$$\frac{L}{2}\mathbb{E}[\|v^{t+1} - v^t\|^2] = \frac{L\eta^2}{2}\mathbb{E}\left[\|\frac{1}{N}\sum_{i=1}^{N} g_i^t\|^2\right] \tag{29}$$

$$= \frac{L\eta^2}{2}\mathbb{E}\left[\|\frac{1}{N}\sum_{i=1}^{N}(g_i^t - \nabla f_i(x_i^t))\|^2\right] + \frac{L\eta^2}{2}\mathbb{E}\left[\|\frac{1}{N}\sum_{i=1}^{N}\nabla f_i(x_i^t)\|^2\right] \tag{30}$$

$$= \frac{L\eta^2}{2N^2}\sum_{i=1}^{N}\mathbb{E}\left[\|g_i^t - \nabla f_i(x_i^t)\|^2\right] + \frac{L\eta^2}{2}\mathbb{E}\left[\|\frac{1}{N}\sum_{i=1}^{N}\nabla f_i(x_i^t)\|^2\right] \tag{31}$$

$$\leq \frac{L\eta^2\sigma^2}{2N} + \frac{L\eta^2}{2}\mathbb{E}\left[\|\frac{1}{N}\sum_{i=1}^{N}\nabla f_i(x_i^t)\|^2\right] \tag{32}$$

Substituting these into equation 24 yields

$$\mathbb{E}[\|f(v^{t+1}\|] \tag{33}$$

$$\leq \mathbb{E}[\|f(v^t)\|] - \frac{\eta}{2}\mathbb{E}\left[\|\nabla f(v^t)\|^2\right] - \frac{\eta - L\eta^2}{2}\mathbb{E}\left[\|\frac{1}{N}\sum_{i=1}^{N}\nabla f_i(x_i^t)\|^2\right]$$
$$+ \frac{\eta}{2}\mathbb{E}\left[\|\nabla f(v^t) - \frac{1}{N}\sum_{i=1}^{N}\nabla f_i(x_i^t)\|^2\right] + \frac{L\eta^2\sigma^2}{2N} \tag{34}$$

$$\leq \mathbb{E}[\|f(v^t)\|] - \frac{\eta}{2}\mathbb{E}\left[\|\nabla f(v^t)\|^2\right] + \frac{\eta}{2}\mathbb{E}\left[\|\nabla f(v^t) - \frac{1}{N}\sum_{i=1}^{N}\nabla f_i(x_i^t)\|^2\right] + \frac{L\eta^2\sigma^2}{2N} \tag{35}$$

$$\leq \mathbb{E}[\|f(v^t)\|] - \frac{\eta}{2}\mathbb{E}\left[\|\nabla f(v^t)\|^2\right] + \frac{\eta}{2N}\sum_{i=1}^{N}\mathbb{E}\left[\|\nabla f(v^t) - \nabla f_i(x_i^t)\|^2\right] + \frac{L\eta^2\sigma^2}{2N} \tag{36}$$

$$\leq \mathbb{E}[\|f(v^t)\|] - \frac{\eta}{2}\mathbb{E}\left[\|\nabla f(v^t)\|^2\right] + \frac{\eta L^2}{2N}\sum_{i=1}^{N}\mathbb{E}\left[\|v^t - x_i^t\|^2\right] + \frac{L\eta^2\sigma^2}{2N} \tag{37}$$

$$\leq \mathbb{E}[\|f(v^t)\|] - \frac{\eta}{2}\mathbb{E}\left[\|\nabla f(v^t)\|^2\right] + (3D^2(\omega, \Omega; \Delta) + 1.5C^2(\theta, \Theta; \Delta))L^2\eta^3G^2 + \frac{L\eta^2\sigma^2}{2N} \tag{38}$$

Dividing both sides by $\frac{\eta}{2}$ and rearranging the terms, we have

$$\mathbb{E}\left[\|\nabla f(v^t)\|^2\right] \leq \frac{2}{\eta}\left(\mathbb{E}[\|f(v^t)\|] - \mathbb{E}[\|f(v^{t+1}\|]\right)$$
$$+ 3(2D^2(\omega, \Omega; \Delta) + C^2(\delta, \Theta; \Delta))L^2\eta^2G^2 + \frac{L\eta\sigma^2}{N} \tag{39}$$

Taking the sum over $t = 0, 1, ..., T-1$ and dividing both sides by T, we have

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left[\|\nabla f(v^t)\|^2\right] \tag{40}$$

$$\leq \frac{2}{\eta T}\left(\mathbb{E}[\|f(v^0)\|] - \mathbb{E}[\|f(v^T\|]\right) + 3(2D^2(\omega,\Omega;\Delta) + C^2(\delta,\Theta;\Delta))L^2\eta^2G^2 + \frac{L\eta\sigma^2}{N} \tag{41}$$

$$\leq \frac{2}{\eta T}\left(f(x^0) - f^*\right) + 3(2D^2(\omega,\Omega;\Delta) + C^2(\delta,\Theta;\Delta))L^2\eta^2G^2 + \frac{L\eta\sigma^2}{N} \tag{42}$$

where $f^* = \min_x f(x)$. Now, for the real sequence, we have

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left[\|\nabla f(x^t)\|^2\right] \tag{43}$$

$$\leq \frac{1}{T}\sum_{t=0}^{T-1}(2\mathbb{E}\left[\|\nabla f(v^t)\|^2\right] + 2\mathbb{E}\left[\|\nabla f(v^t) - \nabla f(x^t)\|^2\right]) \tag{44}$$

$$\leq \frac{1}{T}\sum_{t=0}^{T-1}(2\mathbb{E}\left[\|\nabla f(v^t)\|^2\right] + 2L^2\mathbb{E}\left[\|v^t - x^t\|^2\right]) \tag{45}$$

$$\leq \frac{2}{T}\sum_{t=0}^{T-1}\mathbb{E}\left[\|\nabla f(v^t)\|^2\right] + 2C^2(\delta,\Theta;\Delta)L^2\eta^2G^2 \tag{46}$$

Substituting the bound on the virtual sequence into the above equation, we have,

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left[\|\nabla f(x^t)\|^2\right] \leq \frac{4}{\eta T}\left(f(x^0) - f^*\right) + 4(3D^2(\omega,\Omega;\Delta) + 2C^2(\delta,\Theta;\Delta))L^2\eta^2G^2 + \frac{2L\eta\sigma^2}{N} \tag{47}$$

This completes the proof.

D.5 Proof of Proposition 2

Let $\pi_u(s)$ be the fraction of clients whose next server meeting time is in s time slots. For sufficiently many clients, the distribution $\pi_u = [\pi_u(0)\ \pi_u(1)\ ...\ \pi_u(\Delta)]$ is time-invariant and can be calculated by solving the stationary distribution of a Markov chain that describes the client state in terms of the remaining server meeting time. Specifically, π_u is the solution to

$$\pi_u \begin{bmatrix} 0 & P_{\text{int}}(1) & P_{\text{int}}(2) & ... & P_{\text{int}}(\Delta) \\ 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 1 & 0 \end{bmatrix} = \pi_u \tag{48}$$

Furthermore, we let $q_u(t) = \sum_{s=0}^{t}\pi_u(s)$ be the cumulative distribution function. Suppose in the $(t+1)$-th slot in the upload search interval, a client is met. The probability that this client is a semi-qualified upload relay is $q_u(\Theta - \theta - t)$. Therefore, the probability that no semi-qualified upload client is met in the $(t+1)$-th slot is $1 - \rho q_u(\Theta - \theta - t)$. The probability that no semi-qualified upload relay client is met during the entire upload search interval is thus $\prod_{t=0}^{\Theta-\theta}(1 - \rho q(\Theta - \theta - t))$.

Similarly, let $\pi_d(s)$ be the fraction of clients whose last server meeting time was s time slots ago. The invariant distribution can be solved according to another Markov chain as

$$\pi_d \begin{bmatrix} 0 & 1 & 0 & ... & 0 \\ P'_{\text{int}}(1) & 0 & 1 - P'_{\text{int}}(1) & \cdots & 0 \\ P'_{\text{int}}(2) & 0 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ P'_{\text{int}}(\Delta) & \cdots & 0 & 0 & 1 - P'_{\text{int}}(\Delta) \end{bmatrix} = \pi_d \tag{49}$$

where $P'_{\text{int}}(t) = \frac{P_{\text{int}}(t)}{\sum_{s=t}^{\Delta} P_{\text{int}}(s)}$. Let $q_d(t) = \sum_{s=0}^{t} \pi_d(s)$ be the cumulative distribution function. Suppose in the t-th slot in the download search interval, a client is met. The probability that this client is a semi-qualified download relay is $q_d(t)$. Therefore, the probability that no semi-qualified download client is met in the t-th slot is $1-\rho q_d(t)$. The probability that no semi-qualified download relay client is met during the entire download search interval is thus $\prod_{t=0}^{\Omega-\omega}(1-\rho q_d(t))$.

Finally, the probabilities of meeting at least one semi-qualified relay are obtained.

E RELAYING MANIPULATED CLU

In this section, we present an extension of FedMobile where clients upload manipulated CLUs via relaying. The manipulation operations can be either quantization/compression (in order to reduce the relayed data size) or perturbation (in order to add privacy protection). To avoid confusions, we let n_i^t denote the cumulative gradient update of client i's own, which is not combined with any CLUs received from other clients. We call n_i^t the private-CLU of client i.

In general, there is a difference between the CLUs before and after manipulation. We denote this difference by ϵ_i^t. In the case of quantization, a quantized private-CLU is used for relaying. Therefore, the difference is $\epsilon_i^t = Q(n_i^t) - n_i^t$ where $Q(\cdot)$ is the quantizer operator. In the case of perturbation, a noise ϵ_i^t is directly added to n_i^t to create a noisy CLU $n_i^t + \epsilon_i^t$ for relaying. To facilitate presentation, we call ϵ_i^t noise in both cases.

When to upload via relaying and who should be the relay? This is the same as the original FedMobile strategy.

How to relay the local updates to the server? Upon a CLU exchange event at time t between a sender client i and a relay client j:

RESET (by sender): Before sending the CLU to the relay client j, the sender client first records the noise ϵ_i^t. Then the manipulated CLU, i.e., $\tilde{m}_i^t = m_i^t + \epsilon_i^t$, is sent to the relay. The sender then resets its CLU to $m_i^t := -\epsilon_i^t$ and its private-CLU to $n_i^t := 0$.

COMBINE (by relay): After receiving $\tilde{m}_i^t$ from client i, client j combines m_j^t and $\widetilde{m}_i^t$ to produce a new m_j^t, i.e., $m_j^t := m_j^t + \widetilde{m}_i^t$.

In this way, the server could get the client i's manipulated CLU sooner. Then the manipulated CLU will later be corrected when the client i meets the server.

E.1 CONVERGENCE ANALYSIS

In this subsection, we prove the convergence of FedMobile with manipulated CLU uploading. We first state an additional assumption on the noise term ϵ_i^t.

Assumption 5 (Bounded Error)**.** *The noise term ϵ_i is bounded, i.e., $\mathbb{E}[\|\epsilon_i^t\|^2] \leq q\mathbb{E}[\|n_i^t\|^2]$, $\forall i = 1, ..., N$, $\forall t$ for some positive real constant q.*

The *corrupted real sequence* of the global model can be written as

$$\widetilde{x}^t = x^t + \frac{1}{N}\sum_{i\in U_t}\epsilon_i, \quad \forall t \tag{50}$$

where we define U_t as the set of clients for whom only corrupted CLUs have been received by the server via the relay, and $|U_t|$ is the size of U_t. The real sequence x^t is the imaginary real sequence where the noise is not added.

Lemma 4. *The difference of the corrupted real global model and the virtual global model is bounded as follows*

$$\mathbb{E}\left[\|v^t - \widetilde{x}^t\|^2\right] \leq 2C^2(\delta, \Theta; \Delta)\eta^2 G^2 + 2q\Theta^2\eta^2 G^2 \tag{51}$$

For each client i, the difference of its local model and the virtual global model is bounded as follows

$$\mathbb{E}\left[\|v^t - x_i^t\|^2\right] \leq 6(D^2(\omega, \Omega; \Delta) + C^2(\theta, \Theta; \Delta) + q\Theta^2)\eta^2 G^2 \tag{52}$$

Proof: Consider any time t, let t_i' be the meeting time of client i and its relay, by the definition of corrupted real sequence, we have

$$\mathbb{E}\left[\|v^t - \widetilde{x}^t\|^2\right] = \mathbb{E}\left[\|v^t - x^t + x^t - \widetilde{x}^t\|^2\right] \tag{53}$$

$$\leq 2\mathbb{E}\left[\|v^t - x^t\|^2\right] + 2\mathbb{E}\left[\|x^t - \widetilde{x}^t\|^2\right] \tag{54}$$

$$\leq 2C^2(\delta, \Theta; \Delta)\eta^2 G^2 + 2\mathbb{E}\left[\|\frac{1}{N}\sum_{i\in U_t}\epsilon_i\|^2\right] \tag{55}$$

$$\leq 2C^2(\delta, \Theta; \Delta)\eta^2 G^2 + \frac{2}{N^2}|U_t|\sum_{i\in U_t}\mathbb{E}\left[\|\epsilon_i\|^2\right] \tag{56}$$

$$\leq 2C^2(\delta, \Theta; \Delta)\eta^2 G^2 + \frac{2q}{N^2}|U_t|\sum_{i\in U_t}\mathbb{E}[\|n_i^t\|^2] \tag{57}$$

$$\leq 2C^2(\delta, \Theta; \Delta)\eta^2 G^2 + \frac{2q}{N^2}|U_t|\sum_{i\in U_t}\mathbb{E}[\|\sum_{s=\tau_i^{\text{last}}(t)}^{t_i'-1}\eta g_i^s\|^2] \tag{58}$$

$$\leq 2C^2(\delta, \Theta; \Delta)\eta^2 G^2 + \frac{2q\Theta\eta^2}{N^2}|U_t|\sum_{i\in U_t}\sum_{s=\tau_i^{\text{last}}(t)}^{t_i'-1}\mathbb{E}[\|g_i^s\|^2] \tag{59}$$

$$\leq 2C^2(\delta, \Theta; \Delta)\eta^2 G^2 + \frac{2q\Theta\eta^2}{N^2}|U_t|\sum_{i\in U_t}\sum_{s=\tau_i^{\text{last}}(t)}^{t_i'-1}G^2 \tag{60}$$

$$\leq 2C^2(\delta, \Theta; \Delta)\eta^2 G^2 + \frac{2q\Theta^2\eta^2}{N^2}|U_t|^2 G^2 \tag{61}$$

$$\leq 2C^2(\delta, \Theta; \Delta)\eta^2 G^2 + 2q\Theta^2\eta^2 G^2 \tag{62}$$

Similar to the proof of Lemma 3, consider client i at any time t with the global model version $\psi_i(t)$. Therefore client i is doing the local training steps using the global model $x^{\psi_i(t)}$ as the initial model. Thus, we use $x^{\psi_i(t)}$ as the anchor model to investigate the difference between v^t and x_i^t.

$$\mathbb{E}\left[\left\|v^t - x_i^t\right\|^2\right] \tag{63}$$

$$= \mathbb{E}\left[\left\|v^t - v^{\psi_i(t)} + v^{\psi_i(t)} - \widetilde{x}^{\psi_i(t)} + \widetilde{x}^{\psi_i(t)} - x_i^t\right\|^2\right] \tag{64}$$

$$\leq 3\mathbb{E}\left[\left\|v^t - v^{\psi_i(t)}\right\|^2 + \left\|v^{\psi_i(t)} - \widetilde{x}^{\psi_i(t)}\right\|^2 + \left\|\widetilde{x}^{\psi_i(t)} - x_i^t\right\|^2\right] \tag{65}$$

$$\leq 3\mathbb{E}\left[\left\|\frac{1}{N}\sum_{i=1}^{N}\sum_{s=\psi_i(t)}^{t-1}\eta g_i^s\right\|^2\right] + 3\mathbb{E}\left[\left\|v^{\psi_i(t)} - \widetilde{x}^{\psi_i(t)}\right\|^2\right] + 3\mathbb{E}\left[\left\|x_i^{t'} - x_i^t\right\|^2\right] \tag{66}$$

$$\leq 3\mathbb{E}\left[\left\|\frac{1}{N}\sum_{i=1}^{N}\sum_{s=\psi_i(t)}^{t-1}\eta g_i^s\right\|^2\right] + 3\mathbb{E}\left[\left\|v^{\psi_i(t)} - \widetilde{x}^{\psi_i(t)}\right\|^2\right] + 3\mathbb{E}\left[\left\|x_i^{\psi_i(t)} - x_i^t\right\|^2\right] \tag{67}$$

$$\leq 3D^2(\omega, \Omega; \Delta)\eta^2 G^2 + 6C^2(\delta, \Theta; \Delta)\eta^2 G^2 + 6q\Theta^2\eta^2 G^2 + 3D^2(\omega, \Omega; \Delta)\eta^2 G^2 \tag{68}$$

$$= 6(D^2(\omega, \Omega; \Delta) + C^2(\theta, \Theta; \Delta) + q\Theta^2)\eta^2 G^2 \tag{69}$$

Theorem 2. *With manipulated CLU uploading, assuming at least one semi-qualified upload (download) relay client exists in every upload (download) search interval, by setting $\eta \leq 1/L$, after T*

time slots, we have

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left[\|\nabla f(x^t)\|^2\right] \leq \frac{4}{\eta T}\left(f(x^0)-f^*\right)$$
$$+ 4(3D^2(\omega,\Omega;\Delta)+4C^2(\delta,\Theta;\Delta)+4q\Theta^2)L^2\eta^2G^2+\frac{2L\eta\sigma^2}{N} \tag{70}$$

Proof: The proof of Theorem 2 follows the proof of Theorem 1 by replacing Lemma 3 with Lemma 4.

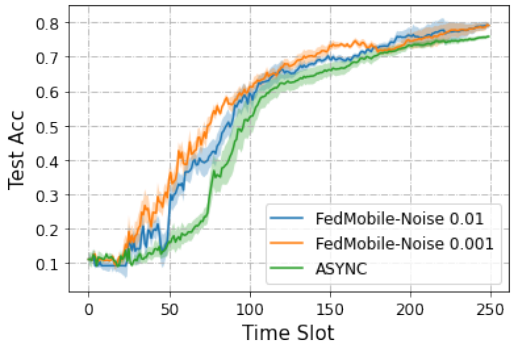


Figure 15: Privacy Experiment

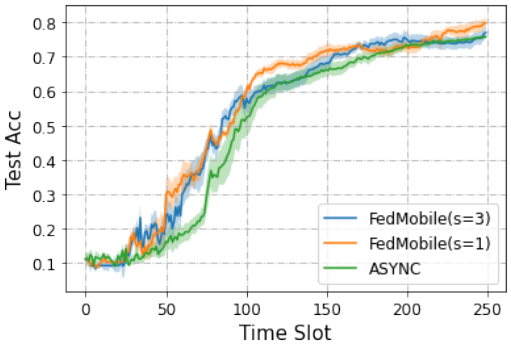


Figure 16: Quantization Experiment

E.2 Experiment Results

We test two types of manipulation. In the first type, we directly add Gaussian noises to the relayed CLU. Fig. 15 shows the convergence curves under different amount of noises. (e.g. Gaussian Noise $\mathcal{N}_1(0, 0.01)$ and Gaussian Noise $\mathcal{N}_2(0, 0.001)$). In the second type, we utilize the low precision quantizer in Alistarh et al. (2017) to quantize the CLU before relaying. Here the quantization level is defined as s. Fig. 16 shows the convergence curves under different quantization levels. In both cases, FedMobile still outperforms the baseline method provided that the added noise is small or the adopted quantization level is low.