Stochastic Client Selection for Federated Learning With Volatile Clients

Tiansheng Huang, Weiwei Lin, *Member, IEEE*, Li Shen, Keqin Li, *Fellow, IEEE*, and Albert Y. Zomaya, *Fellow, IEEE*

***Abstract*—Federated learning (FL), arising as a privacy-preserving machine learning paradigm, has received notable attention from the public. In each round of synchronous FL training, only a fraction of available clients are chosen to participate, and the selection decision might have a significant effect on the training efficiency, as well as the final model performance. In this article, we investigate the client selection problem under a volatile context, in which the local training of heterogeneous clients is likely to fail due to various kinds of reasons and in different levels of frequency. Intuitively, too much training failure might potentially reduce the training efficiency, while too much selection on clients with greater stability might introduce bias, thereby resulting in degradation of the training effectiveness. To tackle this tradeoff, we, in this article, formulate the client selection problem under joint consideration of effective participation and fairness. Furthermore, we propose E3CS, a stochastic client selection scheme as a solution. According to our experimental results over a public data set, the proposed selection scheme is able to achieve up to 2× faster convergence to a fixed model accuracy while maintaining the same level of final model accuracy, compared with the state-of-the-art selection schemes.**

***Index Terms*—Adversarial multiarm bandit (MAB), client selection, exponential-weight algorithm for exploration and exploitation (Exp3), fairness scheduling, federated learning (FL).**

Manuscript received 16 November 2020; revised 7 May 2021 and 11 February 2022; accepted 23 April 2022. Date of publication 3 May 2022; date of current version 7 October 2022. This work was supported in part by the Key-Area Research and Development Program of Guangdong Province under Grant 2021B0101420002; in part by the National Natural Science Foundation of China under Grant 62072187 and Grant 61872084; in part by the Guangzhou Science and Technology Program Key Projects under Grant 202007040002; in part by the Guangzhou Development Zone Science and Technology under Grant 2020GH10; in part by the Guangdong Marine Economic Development Special Fund Project under Grant GDNRC[2022]17; and in part by the Science and Technology Innovation 2030—"Brain Science and Brain-Like Research" Major Project under Grant 2021ZD0201402 and Grant 2021ZD0201405. *(Corresponding author: Weiwei Lin.)*

Tiansheng Huang and Weiwei Lin are with the School of Computer Science and Engineering, South China University of Technology, Guangzhou 510640, China (e-mail: tianshenghuangscut@gmail.com; linww@scut.edu.cn).

Li Shen is with JD Explore Academy, JD.com, Beijing 100000, China (e-mail: mathshenli@gmail.com).

Keqin Li is with the Department of Computer Science, State University of New York, New Paltz, NY 12561 USA (e-mail: lik@newpaltz.edu).

Albert Y. Zomaya is with the School of Computer Science, The University of Sydney, Sydney, NSW 2006, Australia (e-mail: albert.zomaya@sydney.edu.au).

This article has supplementary downloadable material available at https://doi.org/10.1109/JIOT.2022.3172113, provided by the authors.

Digital Object Identifier 10.1109/JIOT.2022.3172113

I. Introduction

A. Background

DATA privacy issue has received notable attention nowadays, making the acquisition of reliable and realistic data an even more challenging task. However, without the support of massive real-world data, model training by artificial intelligence (AI)-based technique might not be realistic. In this security-demanding context, federated learning (FL), a privacy-preserving machine learning paradigm, has come into vision. In FL, training data possessed by a client (e.g., mobile phone, personal laptop, etc.) does not need to leave the sources and be uploaded to a centralized entity for model training. All the training is done in the local devices alone, and only the posttrained models, rather than the raw data, would be exposed to other entities. By this mechanism, potential data exposure could be reduced to a minimum extent and thus, making data sharing less reluctant by the data owners.

B. Motivations

The training of canonical FL is an iterated process, in which the following basic procedure performs in sequence: 1) server makes a client selection decision and distributes the global model to the selected participants; 2) clients (or participants) take advantage of their local data to train the global model. Explicitly, it does multiple steps of optimization [like stochastic gradient descent (SGD)] to the global model; 3) clients upload the posttrained model after training is accomplished; and 4) server aggregates the uploaded model (i.e., averages the model weights of all the uploaded models) and repeats the above procedures until the aggregated model is converged.

In the client selection stage, due to the limited bandwidth as well as the budget issue, not all the clients in the system are selected for training. As a result, some portion of training data is simply missing from training in each round. Under this partial participation mechanism, multiple factors, such as the *number of aggregated models*, may play a critical role in the FL's convergence and performance. Empirical observations given in [1] (i.e., the first FL paper) agree that increasing the number of participants (equivalently, enlarging the number of posttrained models to aggregate, if no training failure presents) in each round may boost the convergence speed.

Now, consider a more realistic *volatile training context*, in which *the selected client may not successfully return their models for aggregation in each round*. These failures could be as a result of various kinds of reasons, e.g., insufficient

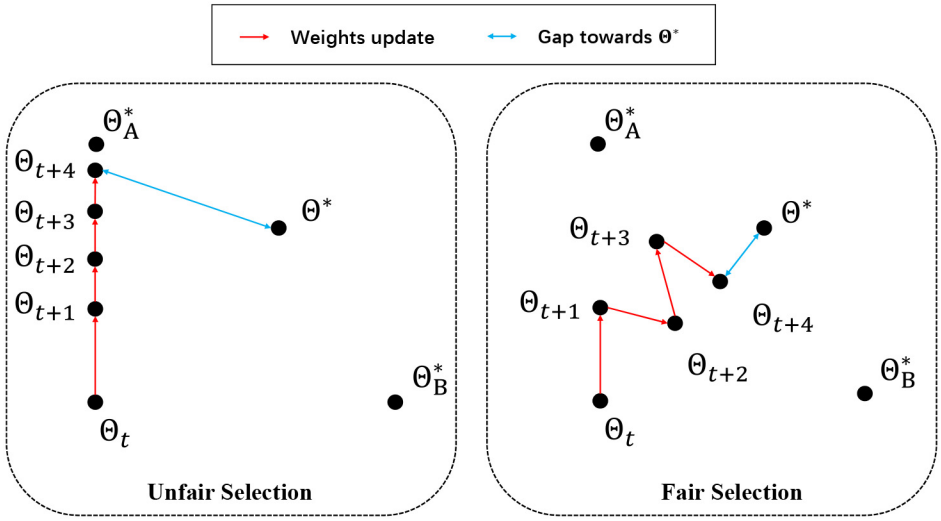


Fig. 1. Example illustrating the effect of fairness in FL, where $\boldsymbol{\Theta}_A^*$ and $\boldsymbol{\Theta}_B^*$, respectively, are the local weights of Client A and Client B [i.e., $\boldsymbol{\Theta}_A^* = \arg\min_{\boldsymbol{\Theta}} f(\boldsymbol{\Theta}; \mathcal{D}_A)$ and $\Theta_B^* = \arg\min_{\boldsymbol{\Theta}} f(\boldsymbol{\Theta}; \mathcal{D}_B)$] and Θ^* is the global optimal weights (i.e., $\Theta^* = \arg\min_{\boldsymbol{\Theta}} f(\boldsymbol{\Theta}; \mathcal{D}_A \cup \mathcal{D}_B)$). In each round of training, only one of the two clients is asked to train the global model (but no aggregation is needed in this example, which is a special case of FL). The left selection scheme consistently involves Client A while the right one gives the two clients the same opportunity to participate. It can be observed that after four rounds of training, the obtained global weights of the fairer selection is much closer to the optimal global weights $\boldsymbol{\Theta}^*$, so typically the model enjoys smaller loss and higher accuracy.

computing resources, user abort, network failure, etc. This volatile context is pretty common in *IoT scenario*, and typically, in this particular scenario, different clients may experience different rates of failure owing to their heterogeneous composition. Under this more realistic consideration, and combining the observations given in [1], we may derive a rather heuristic rule of making client selection for this context: to maximize the *effective participation* in each round. That is, *simply selecting the clients with the lowest failure probability (i.e., the stable ones) would be beneficial and may accelerate the training.*

But this statement could be erroneous as subsequent study reveals a possible drawback of this naive selection scheme. The major concern arises from the violation of *selection fairness*. In our previous work [2], we empirically substantiate that biased selection of clients may somehow hurt the model performance (or final model accuracy). Specifically, we argue that too much selection on a specific group of clients may make the global model "drifting" toward their local optimizer. That is, the model might overfit to the data owned by those clients that we frequently select, while having poor accuracy for those seldom accessed (see Fig. 1 for an example). Barring the most intuitive observation, a concurrent study [3] has provided theoretical analysis on selection fairness. They present a nonvanishing bias term in the convergence error, which is exactly resulting from the *selection bias* (*selection skewness* in their contexts) toward some specific clients.

To sum up, reasonable inference based on existing studies reveals that: 1) tendentiously selecting stable clients might increase effective participation, and thereby, accelerate convergence and 2) but selection bias might deteriorate the obtained model's performance. As both the training efficiency and effectiveness are of interest in FL, we, in this article, like to investigate the tradeoff between *effective participation* (i.e., number of returned models for each round) and *selection fairness* in a volatile training context, and discover if there is a nice way to tame these two seemingly contrastive metrics.

C. Contributions

The main contributions of this article are listed as follows.

1) We propose a deadline-based aggregation mechanism to cope with FL aggregation in a volatile training context.
2) Under the new aggregation mechanism and the premise of the client's volatility, we formulate the global optimization problem for FL. To simplify the problem, we decompose the global problem into two subproblems based on the idea of alternating minimization. After that, we propose to relax the client selection subproblem to a solvable form based on empirical observations.
3) We design an efficient solution termed exponential-weight algorithm for exploration and exploitation (Exp3)-based client selection (E3CS) for the defined stochastic client selection problem. Theoretically, we derive the regret bound of E3CS, and we further discuss its relation to the canonical Exp3 in our analysis. Empirically, numerical as well as real data-based experiments are conducted to substantiate the effectiveness of our proposed solutions.

To the best of our knowledge, this is the first paper that presents a systematic study for FL under a volatile context. Before this study, almost all the literature on synchronized FL escape the potential dropout phenomenon of clients. Consequently, this article may bring new insights into the subsequent research of synchronized FL.

II. Related Works

Open research of FL can be classified into the following aspects.

A. Statistical Heterogeneity

Zhao *et al.* [4] highlighted the issue of nonindependently identical distribution (non-iid) of data. They presented the experimental data of FL training in a non-iid setting, which indicates a test accuracy loss of up to 51% for CIFAR-10[1] and 11% for MNIST,[2] comparing to an iid case. This phenomenon can be explained by statistical heterogeneity, which makes the local optimum model weights for different clients utterly different, and is proven to affect the convergence rate of FL in [7]. As a potential solution, Zhao *et al.* [4] proposed to create a small subset of public data, which is distributed to clients to form a rectified "iid" data set. A similar idea of data exchange is also available in [8] and [9]. Identifying the potential privacy leakage and communication overhead of the above data-exchange-based approach, Jeong *et al.* [10] further proposed federated augmentation (FAug), which essentially is to train a generative adversarial network (GAN) to produce the "missing" data samples, so as to make the training data set becoming iid. However, potential privacy intrusion still persists in this data generation method, since training GAN needs seed data samples uploaded from clients.

B. System Heterogeneity

Another performance bottleneck faced by FL is the heterogeneous computing and network capacity of clients. In synchronous FL, aggregation could only be conducted when all the clients fully complete their local training and return their posttrained model. But the clients have utterly different performances in the real use case, which implies that some "faster" clients have to wait for the "slower," resulting in unnecessary time consuming.

To tackle this problem, Li *et al.* [11] allowed the training epochs of different clients to be inconsistent, such that "weaker" clients can be allowed to perform less computing in each round of training. But this more flexible setting makes the original FedAvg scheme performs even worse in the case that the clients' data are *highly statistically heterogeneous*. An intuitive explanation for this phenomenon is that the global model is peculiarly prone to drift (or be pulled) toward the local optimum of those clients who conducted more epochs of training, resulting in an incomplete and unbalanced global model. To cope with this arising problem, Li *et al.* [11] proposed to use a proximal term to constrain the "distance" between the local model and global model, such that the local model of those clients with more computation (or more steps of update) would not be so drastically deviated from the global one, and therefore, mitigating the drifting phenomenon. In another recent work [12], the idea of imposing varying epochs for different clients is also adopted. But they propose an alternative solution for the drifting problem. Specifically, they assign epochs-related aggregation weights to different clients, and the clients with fewer training epochs are assigned with higher aggregation weights, such that their local models would not be overwhelmed by those with larger training epochs.

However, we argue that system heterogeneity in FL not only specifies the local training epochs of clients but more aspects (such as volatility of clients) should also be considered. As such, we, in this article, extend the notion of system heterogeneity in FL to the client's heterogeneous volatility, and discover its subsequent impact on the overall training performance.

C. Client Selection

In FL, due to the communication bottleneck, often only a subset of clients could be selected to put into training in each round, which is termed as partial participation (aka partial selection). Partial participation might introduce extra bias in update gradients, thereby deteriorating FL's training performance. But the proper selection of clients in each round may potentially narrow this performance gap.

The earliest record of this genre of research is perhaps [13], in which Nishio and Yonetani proposed a rather intuitive selection scheme and highlighted the importance of client update number to the model performance. In a more recent work [14], the joint bandwidth partition and client selection issue was investigated in depth, and the joint optimization problem was solved by a rather traditional numerical method. Following this line of research, Xu and Wang [15] constructed a long-term client energy constraint to the selection problem, which essentially is to reserve energy during initial rounds of training so that more clients have chances to be involved in the later rounds of training. This study yields a very similar observation as ours that the FL process indeed benefits if later rounds of training cover more clients. We do believe that such a phenomenon is due to the fairness effect that we discover, i.e., the model needs more diversified data to further improve if near convergence.

In the earliest client selection study, Nishio and Yonetani [13] assumed that the training status of clients (e.g., the training time, resource usage, etc.) is known or can at least be calculated. However, client selection is not always based on a known context. Sometimes we do need some historical information (or the reputation of clients) to facilitate decision making. In [16], a reinforcement learning-based selection scheme was proposed. The authors take dimensionality reduced model weights as states in a Markov decision process, and use the reinforcement learning technique to optimize the selection decision for each state (i.e., each group of model weights after dimensionality reduction). However, this specific method needs thousands of epochs of FL data to train the reinforcement learning model and thereby, may lose some of its applicability and genericity. Alternatively, an exploration and exploitation balance model, e.g., multiarm bandit (MAB), would be more realistic. For example, Yoshida *et al.* [17] leveraged a canonical MAB and further developed a rectified way to accommodate the intrinsic problem of canonical MAB, i.e., an exponential number of arms' combination. In another work [18], Xia *et al.* made use of a combinatorial MAB

[1]The CIFAR-10 [5] data set is a collection of images that are commonly used to train machine learning and computer vision algorithms.

[2]MNIST [6] is a collection of handwritten digits that are commonly used for training various machine learning model.

setting, in which the independent feedback of each arm is directly observed, enabling their designed UCB algorithm to boast a strict regret bound. Their work further took the fairness constraint into account, which appears to be a critical factor in FL training. Our previous work [2] shared a very similar consideration with [18], despite that we alternatively assumed a combinatorial contextual bandit for modeling of training time. Similarly, in [19], a MAB-based selection was also implemented, but the feedback obtained for clients not only involved training time (or training efficiency) but the statistical efficiency (or training quality) was also jointly modeled. All of the above-mentioned work made use of a deterministic UCB-based algorithm to solve the bandit problem. In this work, rather than employing the deterministic UCB algorithm, we shall alternatively adopt a stochastic Exp3-based algorithm, which makes it more natural and intuitive to cope with the fairness constraint, without the need of imposing dynamic queues, as [18] and [2] did.

Selection fairness is another dominant factor in performing client selection. In our concurrent work [3], Cho *et al.* have derived a theoretical analysis of the selection skewness. Based on their finding, selection bias may introduce a nonvanishing constant term in the convergence error, but bias toward clients with higher local loss may accelerate the convergence speed (see the vanishing term in their Theorem 3.1). Inspired by the derived results, they proposed a power-of-choice client selection strategy to cope with the tradeoff between the local loss of the selected clients and the selection fairness.

In [3], a stochastic selection scheme was considered in their theoretical analysis, and they concluded that selection bias toward clients with larger loss might: 1) accelerate the convergence rate, but 2) enlarge the gap between the convergence value to the global optimum. However, there is still a theoretical gap between their proposed method and their theoretical result (as they did not derive the optimal sampling probability based on the result, but used a rather heuristic method to determine the extent of skewness). In another study [20], concrete selection probability for independent sampling has been given, in which the authors inherited and applied the basic idea from [21], to make use of an unbiased estimator to do the real update. Specifically, Chen *et al.* [20] formulated the partial update from a client as an unbiased estimator of the real update. Based on this formulation, their goal is to tune the selection probability to minimize the variance between partial update and the real update. They derive the optimal probability allocation to optimize the above problem. However, to derive this optimal probability allocation, real updates from all the clients have to be known in advance, which indeed violates the real intent of performing client selection—we select a subset of clients for training in order to *save the bandwidth, as well as the computation.* Neither goal can be achieved via the optimal sampling proposed by [20].

In a high level, these two recent studies [3] and [20] actually share some common insights in terms of selection. That is, it seems to be beneficial to select those clients with a larger "distance" between the current global model and their local optimal models. In other words, it is better to select the clients whose local optimal models are most deviated from the global model, so that the global model could be pulled toward a proper direction. In [20], this distance is quantified as local update norm, while in [3], it is local loss. These seem to be two relevant metrics to quantify this "distance." Also, fairness matters. Though this issue has not been formally discussed by [20], their unbiased estimator helps maintain some degree of fairness, since the real update of clients with less selection probability is given higher aggregation weights (see their formulation of the unbiased estimator). However, both of these two concurrent works do not consider the volatility of clients, and its subsequent impact on selection, which is the main focus of this work.

D. Agnostic/Fair Federated Learning

The general average loss[3] in FL can be given by

$$\min_{\mathbf{\Theta}} \ \frac{1}{K}\sum_{i=1}^{K}\mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}_i} f(\mathbf{\Theta};(\boldsymbol{x},y)) \tag{1}$$

where $\mathcal{D} = (1/K)\sum_{i=1}^{K}\mathcal{D}_i$ is the global data distribution ($\mathcal{D}_i$ is the local data distribution). However, in agnostic/fair FL, the optimization objective of FL is no longer in this form. Specifically, Mohri *et al.* [22] argued that it is too risky to simply use this general loss as FL's optimization target, as it remains unspecified if the testing data distribution actually coincides with the average distribution among the training clients [i.e., $\mathcal{D}_{test} = (1/K)\sum_{i=1}^{K}\mathcal{D}_i$ may not trivially hold]. A natural reformulation of the objective is to apply a min–max principle, and change the global objective as follows:

$$\min_{\mathbf{\Theta}} \ \max_{\lambda}\mathbb{E}_{(x,y)\sim\mathcal{D}_{\boldsymbol{\lambda}}} f(\mathbf{\Theta};(x,y)) \tag{2}$$

where $\mathcal{D}_{\boldsymbol{\lambda}} = \sum_{i=1}^{K}\lambda_i\mathcal{D}_i$ is a specific kind of mixture of local data distribution. By this transformation, the goal of optimization has transferred to find optimal weights $\mathbf{\Theta}$, which minimizes the expected empirical loss under *any* possible mixture of local data distribution (i.e., agnostic distribution).

However, agnostic FL applying the min–max principle might be too rigid, as it maximizes the model's worst performance on any mixture of training data. Li *et al.* [23] proposed to relax the problem by changing the optimization objective to

$$\min_{\mathbf{\Theta}} \ \frac{1}{K}\sum_{i=1}^{K}\mathbb{E}_{(x,y)\sim\mathcal{D}_i}\frac{1}{q+1}f^{q+1}(\mathbf{\Theta};(x,y)) \tag{3}$$

where q is the fairness parameter. By this transformation, the authors introduce *good-intent fairness*, making the goal is instead to ensure that the training procedure does not overfit a model to any one device at the expense of another. Or more concretely, they like to acquire a global model that fits local data from each client, thereby reducing the variance of testing accuracy for each client. This new design is particularly fit to the scenario that the obtained model is used by clients themselves (but not deployed elsewhere). Obviously, fairness has to be nicely maintained in this case, since clients have no reasons to participate if the acquired model has poor performance on

[3]Let us assume the data size of each client is the same for ease of narration.

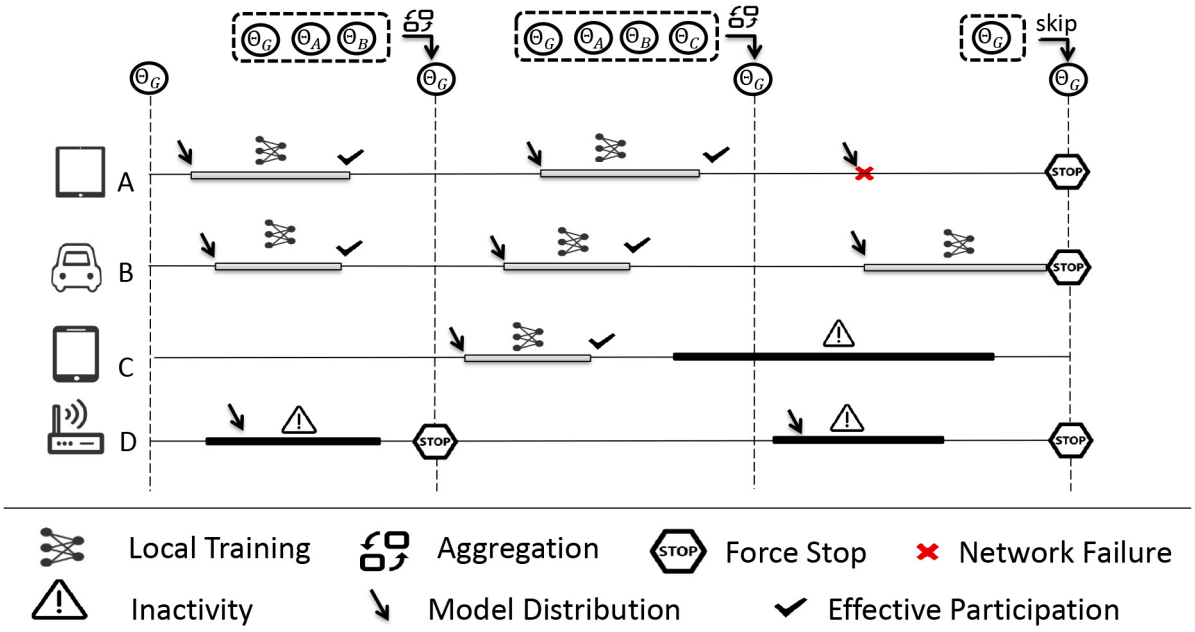


Fig. 2. Illustration of deadline-based aggregation mechanism of FL. $\mathbf{\Theta}_G$, $\mathbf{\Theta}_A$, $\mathbf{\Theta}_B$, $\mathbf{\Theta}_C$, and $\mathbf{\Theta}_D$ presented in the figure, respectively, correspond to the global model, and the posttrained model of clients A, B, C, and D. In this example, three of four clients are chosen to participate in each round. In the first round, the server selects clients A, B, and D to participate, and then distributes global model $\mathbf{\Theta}_G$ to them. But during the first round of training, client D dropouts as a result of a system crash, so only the models from clients A and B are returned before the aggregation deadline, and both of them are counted as effective participation. As no models are returned from client D, a force stop command is issued to her. After aggregation in server, a new global model is produced, via weighted averaging like $\mathbf{\Theta}_G = \omega_A \mathbf{\Theta}_A + \omega_B \mathbf{\Theta}_B + (1 - \omega_A - \omega_B)\mathbf{\Theta}_G$, where ω_A and ω_B are the aggregation weights of a client [see (5) for their definition]. The aggregated model then substitutes the old global model. Then, the second round of training ensues, in which clients A, B, and C are chosen to participate. This time, all of the three return their models on time, so no force stop command needs to be issued. In the third round of training, clients A, B, and D are chosen again, but unfortunately, no effective participation is produced in this round. Client A suffers from network failure, making the model lost during transmission. Client B cannot fulfill the designated training epochs before the deadline, and client D crashes again during training. Then, force stop commands are issued to all the three chosen clients. Obviously, this round of training is futile, as the global model is not updated and remains the same as that in the previous round.

their local data. Moreover, this objective is indeed more flexible than the agnostic min–max objective, in that the fairness extent can be adjusted by tuning q, and therefore, should be more applicable in some use cases of FL.

However, in this article, we stick to the optimization of general FL, in which the optimization is still the general weighted average loss. Note that the notion of fairness (i.e., good-intent fairness) discussed in this section and the selection fairness we mention in this article are not exactly the same notion, since the objectives of maintaining them are different. For good-intent fairness, the goal of maintaining it is to ensure the training loss for different clients to be more uniform. For the selection fairness we discuss in this article, it is maintained in order to minimize the average loss (consider that bias selection might introduce difficulty in reducing average loss), but we do not consider if the local empirical losses are uniform or not.

III. Deadline-Based Aggregation Mechanism

To cope with the potential drop-out of clients, we adopt in this article a deadline-based aggregation mechanism, in which aggregation is made once a fixed deadline is met, such that the issue of "perpetual waiting" could be nicely prevented. To be specific, we involve the following stages in sequence during one round of FL training.

Client Selection and Model Distribution: In this stage, the server determines participants over the available clients and correspondingly distributes the global model to them. After distribution, the main process of the server would sleep and wait until a fixed deadline is met.

Local Training: In this stage, the selected clients conduct designated epochs of training with a local optimizer (e.g., SGD) on the basis of its local data and the distributed global model. The computing epochs of different clients could be designated to different values in advance, as clients may have different computing capacities (i.e., system heterogeneity).

Model Transmission: Once the local training of the clients completes successfully, the posttrained model would be transmitted to the aggregation server immediately.

Force Stop: Once the deadline is met, the server issues the "Force Stop" command to the selected clients, after hearing which clients that are still on the stage of training or model transmission will stop and abort immediately. Models returned after the deadline will be dropped.

Aggregation: The server will check the successful returned model of their validity, after which aggregation based on FedAvg [1] (or possibly other aggregation schemes) is conducted. After this stage, the server would repeat the above procedures to start a new training round.

For a vivid understanding of the training process, we refer the readers to Fig. 2, in which an example of three rounds of FL training is demonstrated.

IV. Problem Formulation

Optimization over FL under the deadline-based aggregation mechanism is characterized by a tuple $(\mathcal{K}, \{x_{i,t}\}_{i\in\mathcal{K},t\in\mathcal{T}}, o_1(\cdot), \{A_t\}_{t\in\mathcal{T}})$. Here, the set $\mathcal{K} \triangleq \{1, 2, \ldots, K\}$ represents the total number of K accessible clients in the system, the training round is characterized by $t \in \mathcal{T} \triangleq \{1, \ldots, T\}$, $\{x_{i,t}\}_{i\in\mathcal{K},t\in\mathcal{T}}$ is the success status of a client in each round, o_1 is the local update operation, and $\{A_t\}_{t\in\mathcal{T}}$ is a set that captures the selected clients each round. We would specify how these components constitute our optimization problem in the following.

A. Basic Assumption and Global Problem

In this section, we shall illustrate the basic setting of FL with volatile clients, and the global optimization problem.

Client Dropout: There is a chance for the clients to drop-out in the middle of training as a result of technical failures (or the client proactively quits the training process). This dropout phenomenon is quite common in IoT scenarios. To model this critical concern, we introduce a binary number i.e., $x_{i,t}$, to capture the status of client i in round t. Formally, $x_{i,t} = 1$ means the training of client i would succeed, and vice versa.

Cardinality Constraint: In each round of FL, only a fraction of clients could be selected to participate due to limited bandwidth (the same setting as in [2], [24], and [25]). Now, we assume that we select k out of K clients in each round, and formally, we need to ensure

$$|A_t| = k \quad t \in \mathcal{T} \tag{4}$$

where A_t is the selection set in round t, and $|A_t|$ is the cardinality of the selection set.

Global Problem: The global optimization problem that we like to solve within the total time frame T is as follows:

$$\begin{aligned}
\text{P1:} \min_{o_1,\{A_t\}_{t\in\mathcal{T}}} \quad & \sum_{i=1}^{K} \mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}_i} \omega_i f(\boldsymbol{\Theta}_{T+1}; (\boldsymbol{x}, y)) \\
\text{s.t.} \quad & |A_t| = k \qquad \text{(cardinality constraint)} \\
& \boldsymbol{\Theta}_{i,t} = o_1(\boldsymbol{\Theta}_t, \mathcal{D}_i) \quad \text{(local update operation)} \\
& \boldsymbol{\Theta}_{t+1} = \sum_{i\in A_t \text{ and } x_{i,t}\neq 0} \omega_i \boldsymbol{\Theta}_{i,t} + \sum_{i\notin A_t \text{ or } x_{i,t}=0} \omega_i \boldsymbol{\Theta}_t \\
& \qquad \text{(aggregation operation for volatile context)}
\end{aligned} \tag{5}$$

where:

1) $\mathcal{D}_i$ is used to denote the local data distribution of client i and $(\boldsymbol{x}, y)$ specifies one piece of data (with $\boldsymbol{x}$ being input and y being label);
2) $\boldsymbol{\Theta}_t \in \mathbb{R}^d$ and $\boldsymbol{\Theta}_{i,t} \in \mathbb{R}^d$ are the global model weights and the local model weights after local update (i.e., updated weights under the premise that local update has been successfully accomplished);
3) $\omega_i = |D_i| / \sum_{i\in[K]} |D_i|$ is used to denote the proportion of data of a specific client, where $|D_i|$ is the number of data the ith client hosts;
4) $f(\boldsymbol{\Theta}_{T+1}; \cdot)$ is the empirical loss of a piece of data given model weights $\boldsymbol{\Theta}_{T+1}$. Our primary objective is to minimize the average empirical loss over data residing in all the clients;
5) $o_1(\cdot)$ is the local update operation. Different schemes in the literature apply different update techniques here. For example, FedAvg [1] applies SGD over the general loss function here. In another work, FedProx [11] employs SGD over a proximal term-involved loss function during the update.

P1 is a comprehensive global problem for FL in a volatile context. It involves the optimization of two key processes of FL, i.e., the local update scheme (o_1) and the client selection scheme (A_t). The volatile context that we consider in this article is directly reflected by the aggregation operation shown in the third constraint. In order to accommodate the volatile training context, the aggregation operation is slightly modified from its original form of FedAvg (original formulation is shown in (3) in [26]). In our modified version, the updates from clients with unsuccessful participation (either not selected or failed in the middle of training) are replaced by the current global model, while in its original form, only those not selected are replaced.

B. Alternative Optimization

The global problem P1 is a combinatorial optimization problem, in which multiple variables are coupling together, imposing great challenges for optimization. To lower the complexity, we adopt the idea of alternating minimization to decompose the problem. Specifically, we divide the variables into two blocks, namely, the client selection block (i.e., A_t) and the local operation block (i.e., o_1). By the decomposition, we are allowed to split the problem into two subproblems, namely

$$\begin{aligned}
\text{P1-SUB1:} \min_{o_1} \quad & \sum_{i=1}^{K} \mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}_i} \omega_i f(\boldsymbol{\Theta}_{T+1}; (\boldsymbol{x}, y)) \\
\text{P1-SUB2:} \min_{\{A_t\}_{t\in\mathcal{T}}} \quad & \sum_{i=1}^{K} \mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}_i} \omega_i f(\boldsymbol{\Theta}_{T+1}; (\boldsymbol{x}, y)) \\
\text{s.t.} \quad & \text{same constraints with P1.}
\end{aligned} \tag{6}$$

P1-SUB1 is a standard optimization problem for FL, to which multiple existing studies (e.g., [1], [11], and [12]) have given concrete solutions, in order to cope with two notorious issues in FL (i.e., system heterogeneity and statistical heterogeneity). However, it remains unexplored how to solve the client selection subproblem P1-SUB2 in the presence of our third constraint under volatile context. In this article, we shall focus on the solution of P1-SUB2, while fixing existing solutions for P1-SUB1 to alternatively optimize the problem.

C. Stochastic Optimization of Client Selection Subproblem

Inspired by [11], we consider to use multinomial sampling[4] to stochastically determine the selection combination each round. Specifically, we assume: 1) $A_t \sim$

[4]Drawing samples from a multinomial distribution with multiple trials is equivalent to draw values from a set, say, $\{1, 2, \ldots, K\}$ in different probability and for multiple times (quite like rolling for multiple times a special dice, which has unequal probability for each side). No replacement specifies that the drawn values between different trials cannot be the same (which reflects the basic property of a combination). A simple implementation (or simulation) of this multinomial distribution can be found in Section V of a draft textbook (can be accessed through http://www.stat.cmu.edu/cshalizi/ADAfaEPoV/).

multinomialNR($\boldsymbol{p}_t/k, k$) $\forall t \in \mathcal{T}$, where multinomialNR($\cdot$) denotes a multinomial distribution without replacement; 2) $\sum_{i=1}^{K} p_{i,t} = k$; and 3) $0 \leq p_{i,t} \leq 1$. These assumptions suffice to ensure that $|A_t| = k$ and $\mathbb{E}[\mathbb{I}_{\{i\in A_t\}}] = p_{i,t}$, which means the probability that a client being selected is $p_{i,t}$. By this transformation, we obtain a stochastic version of P1-SUB2 as follows:

$$\begin{aligned}
\text{P2:} \min_{\{\boldsymbol{p}_t\}_{t\in\mathcal{T}}} \quad & \sum_{i=1}^{K} \mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}_i} \omega_i f(\boldsymbol{\Theta}_{T+1}; (\boldsymbol{x}, y)) \\
\text{s.t.} \quad & A_t \sim \text{multinomialNR}\big(\boldsymbol{p}_t/k, k\big) \\
& \sum_{i=1}^{K} p_{i,t} = k \\
& 0 \leq p_{i,t} \leq 1 \\
& \boldsymbol{\Theta}_{i,t} = o_1(\boldsymbol{\Theta}_t, \mathcal{D}_i) \\
& \boldsymbol{\Theta}_{t+1} = \sum_{i\in A_t \text{ and } x_{i,t}\neq 0} \omega_i \boldsymbol{\Theta}_{i,t} + \sum_{i\notin A_t \text{ or } x_{i,t}=0} \omega_i \boldsymbol{\Theta}_t.
\end{aligned} \tag{7}$$

By this transformation, the problem's optimization target has transferred to the probability allocation $\boldsymbol{p}_t \triangleq (p_{1,t}, \ldots, p_{K,t})$. However, after transformation, it is still much complicated to derive a uniform solution to problem P2, given that the local update operation o_1 could be diversified, and there is not a universal method to quantify their impacts on the client selection policy.

D. Problem Relaxation Based on Empirical Findings

Then, we shall shed light on how we resort to an empirical observation to further relax the problem. More specifically, under the framework of both FedAvg and FedProx (i.e., fixing o_1 according to their setting), we find empirically that two critical factors, i.e., *expected cumulative effective participation (CEP)* and *selection fairness*, might impose a critical effect on the optimization of $\boldsymbol{p}_t$ in the presence of volatility. Now, we shall introduce these two factors in sequence, as follows.

Expected Cumulative Effective Participation: Factually, the training processes of volatile clients are not always bound to succeed. In the course of training, there are various kinds of reasons for the clients to drop-out (or crash), i.e., not capable of returning models to the server on time. For example, clients might unintentionally shut down the training process due to resource limitations, or perhaps clients are too slow to return the posttrained model, leading to a futile participation.[5] From the perspective of an FL server, a frequent failure of the clients is the least desirable to see (since there are fewer models available for aggregation), and therefore, those "stable" clients should be more welcome than the "volatile" ones. Also, empirical observation given in the first FL paper [1] indicates that increasing the client fraction (C in their context) would accelerate the learning process,[6] which means increasing the number of aggregated models would be beneficial in our volatile context. To model this particular concern, we define the metric named *expected CEP*, i.e., the expected number of posttrained models that have been successfully returned. Formally, we propose to maximize the expected CEP as follows:

$$\mathbb{E}\left[\sum_{t=1}^{T}\sum_{i=1}^{K} \mathbb{I}_{\{i\in A_t\}} x_{i,t}\right] \tag{8}$$

where $\mathbb{I}_{\{i\in A_t\}} x_{i,t}$ equals to one only if the ith client is selected (i.e., $i \in A_t$) and the training is success (i.e., $x_{i,t} = 1$). Furthermore, considering the fact that $\mathbb{E}[\mathbb{I}_{\{i\in A_t\}}] = p_{i,t}$, we obtain that

$$\mathbb{E}\left[\sum_{t=1}^{T}\sum_{i=1}^{K} \mathbb{I}_{\{i\in A_t\}} x_{i,t}\right] = \sum_{t=1}^{T}\sum_{i=1}^{K} p_{i,t} x_{i,t}. \tag{9}$$

While maximizing expected CEP, conceivably, the model would be biased toward data from clients that have higher success rate. To control the extent of skewness, we introduce the fairness metric in the following.

Selection Fairness: In the real training scenario, data that reside in clients are normally statistically heterogeneous and therefore, have its own value in promoting training performance. If we intentionally skip the training of some clients (perhaps those with a higher probability to drop-out), then most likely, the final model accuracy would suffer an undesirable loss, since the global model will lean toward the local optimal models of those frequently selected. This training pattern is akin to consistently using a portion of data for training in pure centralized machine learning. Conceivably, with this biased pattern, the trained model will be overfitted to a subset of the frequently involved data, thereby degrading model accuracy. Empirical observation of this performance degradation phenomenon is reported in [2], and is theoretically studied in [3].

As such, we need to ensure fairness to control (though not eliminate) the selection bias. We achieve this goal by reserving some probability for each client to be selected. More concretely, we need to make sure the probability that each client being selected should at least be σ_t, such that each client is at least given some chances to get involved. Formally, we need to make sure

$$p_{i,t} \geq \sigma_t \quad \forall i \in \mathcal{K} \ \ \forall t \in \mathcal{T}. \tag{10}$$

Obviously, we can find that the higher the value of σ_t is, the evener the selection would become. In this regard, we refer to σ_t as *fairness quota*,[7] which directly measures the fairness degree of this round of selection. Also, we need to ensure the constraint $0 \leq \sigma_t \leq k/K$ holds for any t, since: 1) setting $\sigma_t < 0$ is meaningless under the assumption $p_{i,t} \geq 0$ and 2) setting $\sigma_t > k/K$ violates the assumption $\sum_{i=1}^{K} p_{i,t} = k$. Importantly, $\sigma_t = k/K$ implies that $p_{i,t} = k/K$ holds for all clients. In this case, all the clients share the same probability to be selected, and selection would be reduced to uniform

[5] In our deadline-based aggregation scheme, FL server would set up a deadline for collecting models from clients and model submitted after which would be dropped.

[6] See the conclusion made by [1, Table 1].

[7] It is noticeable that we allow the fairness quota in each round not necessarily be the same. This setting is associated with some intrinsic features of FL training, which we will detail in our experimental part.

sampling adopted by FedAvg. Therefore, the fairness quota σ_t serves as a key hyperparameter to tradeoff selection skewness and training efficiency (which could be promoted via biased selection over the clients with higher success rate).

Relaxed Client Selection Problem: Putting all the pieces together, now we shall introduce the relaxed client selection problem, as follows:

$$\begin{aligned} \text{P3:} \max_{\{\boldsymbol{p}_t\}} \ & \sum_{t=1}^{T}\sum_{i=1}^{K} p_{i,t}x_{i,t} \\ \text{s.t.} \ & \sum_{i=1}^{K} p_{i,t} = k \quad \forall t \in \mathcal{T} \\ & \sigma_t \le p_{i,t} \le 1 \quad \forall t \in \mathcal{T} \ \ \forall i \in \mathcal{K}. \end{aligned} \tag{11}$$

However, the problem cannot be solved offline, since the success status of a client, i.e., $x_{i,t}$, is generally unknown by the coordinator before round t. We in the following section would introduce an adversary bandit-based online scheduling solution for problem solving.

V. Solution and Algorithms

In this section, we shall give a solution about how to determine the selection probability of different clients such that the expected CEP is being maximized in P3. Before our formal introduction, we shall first give a preliminary illustration of Exp3, a rather intuitive solution of the adversarial bandit problem, which serves as the base of our proposed solution.

A. Preliminary Introduction to Canonical Exp3

Subjecting to the selection constraint $0 \le p_{i,t} \le 1$ and $\sum_i p_{i,t} = 1$, the objective of adversary bandit is to maximize the following objective: $\max_{\{\boldsymbol{p}_t\}} \ \sum_{t=1}^{T}\sum_i p_{i,t}x_{i,t}$ where $x_{i,t}$ is the reward of drawing the ith arm, and $p_{i,t}$ is the selection probability.

As a solution to adversary bandit, Exp3 first defines the *unbiased estimator* of the real outcome of arms as follows:

$$\hat{x}_{i,t} = \frac{\mathbb{I}_{\{i\in A_t\}}}{p_{i,t}} x_{i,t}. \tag{12}$$

It can be found that $\mathbb{E}[\hat{x}_{i,t}] = x_{i,t}$. Using the unbiased estimator, the *exponential weights*, which reveals the future rewards of an arm, can be formulated as follows:

$$w_{i,t+1} = w_{i,t}\exp\big(\eta\hat{x}_{i,t}\big). \tag{13}$$

As the obtained weights reflect our estimation of each arm's potential outcome, intuitively, we might need to allocate more selection probability to those with a higher weight. Moreover, we need to ensure that the allocated probability sums up to 1. An intuitive way to achieve this requirement is to calculate the allocated probability as follows:

$$p_{i,t} = \frac{w_{i,t}}{\sum_{j\in\mathcal{K}} w_{j,t}}. \tag{14}$$

By this kind of probability allocation, the algorithm should be able to identify the arm with higher rewards and allocate them more probability. However, the canonical Exp3 could only apply to the situation that only one arm is selected in each round. Moreover, the canonical Exp3 cannot guarantee that the selection probability of each arm is at least greater than a constant. Therefore, we need to make necessary adaptations toward it in order to enable multiple plays each round, and accommodate the fairness constraint in our context. In the next section, we shall introduce our adaptation based on an elementary framework [27] for adversarial bandit with multiple plays.

B. EXP3 With Multiple Play and Fairness Constraint

As can be found in problem P3, our formulated problem is in the same form of the adversary bandit problem with multiple plays, though with an extra constraint over the minimum probability to be allocated to a client. In the following, we propose an Exp3-based solution for problem solving.

Unbiased Estimator and Exponential Weights: We consistently use the same unbiased estimator of $x_{i,t}$ as in the canonical Exp3 solution, namely

$$\hat{x}_{i,t} = \frac{\mathbb{I}_{\{i\in A_t\}}}{p_{i,t}} x_{i,t}. \tag{15}$$

Intuitively, we can derive that $\hat{x}_{i,t} \in (0, \infty]$ and $\mathbb{E}[\hat{x}_{i,t}] = x_{i,t}$.

But the exponential weights update should be modified to

$$w_{i,t+1} = \begin{cases} w_{i,t}\exp\Big(\frac{(k-K\sigma_t)\eta\hat{x}_{i,t}}{K}\Big) & i \notin S_t \\ w_{i,t} & i \in S_t \end{cases} \tag{16}$$

where $0 < \eta < 1$ denotes the learning rate of weights update. S_t is the set of clients that experience probability overflow during the probability allocation stages, which will be specified later. The in-depth reason of this modification is available in our proof of regret (see Appendix B in the supplementary material).

Probability Allocation: Given the exponential weights of clients, the way we derive probability allocation should be revised to a more sophisticated form, in order to accommodate $p_{i,t} \ge \sigma_t$ (i.e., the expected fairness constraint) and $\sum_{i=1}^{K} p_{i,t} = k$. In order to qualify these two constraints, our idea is to first allocate a total amount of σ_t probability to each client to accommodate the fairness constraint, and then further allocate the residual amount of probability (i.e., $k - K\sigma_t$, since $\sum_{i=1}^{K} p_{i,t} = k$, which is imposed by the cardinality constraint) according to the clients' weight. Formally, we give

$$p_{i,t} = \sigma_t + (k - K\sigma_t)\frac{w_{i,t}}{\sum_{j\in\mathcal{K}} w_{j,t}}. \tag{17}$$

However, a critical issue might persist in this form of probability allocation: the allocated $p_{i,t}$ would be possibly greater than 1 if the exponential weight of a client is too large, which violates the constraint $p_{i,t} \le 1$ in P3. We refer to such an unexpected phenomenon as *probability overflow*.

Capping the Probability: To address the probability overflow issue, we shall employ a capping-based technique for the probability allocation.

Formally, we need to rewrite (17) as follows:

$$p_{i,t} = \sigma_t + (k - K\sigma_t)\frac{w'_{i,t}}{\sum_{j\in\mathcal{K}} w'_{j,t}} \tag{18}$$

where $w'_{i,t} = \min\{w_{i,t}, (1-\sigma_t)\alpha_t\}$.

By introducing the revised weights, we cap those weights that are too large to a smaller value, i.e., α_t, such that the selection probability of those "overflowed" clients would be capped to 1. Now, the problem has been transferred to choose a proper α_t, making $p_{i,t} \leq 1$ holds for all i. But before we introduce how to derive α_t, we shall first determine the existence of at least one qualified α_t to ensure the applicability of our capping method. Here, we need the following claim.

Claim 1: By the calculation indicated by (18), there exists at least one qualified α_t such that $p_{i,t} \leq 1$ holds for all $i \in \mathcal{K}$.

Proof: To formally start our proof, we need the following observation:

For any $i \in \mathcal{K}$

$$p_{i,t} \leq \sigma_t + (k - K\sigma_t)\frac{(1-\sigma_t)\alpha_t}{\sum_{j\in\mathcal{K}} w'_{j,t}}. \tag{19}$$

Now, we let $\alpha_t = \min_{i\in\mathcal{K}}\{w_{i,t}/(1-\sigma_t)\}$. Then, we know that $\sum_{j\in\mathcal{K}} w'_{j,t} = K\alpha_t(1-\sigma_t)$. It follows that the right hand size (R.H.S) of the above inequality is equivalent to $\sigma_t + [(k - K\sigma_t)/K]$, by which we can ensure that $p_{i,t} \leq 1$ since $k \leq K$. As such, we conclude that there exist at least one α_t that ensures $p_{i,t} \leq 1$ for any i, which completes the proof. ∎

By Claim 1, we justify the existence of α_t such that $p_{i,t} \leq 1$ holds for all i. But to determine our selection of α_t, we should note that α_t should be set as large as possible.[8] As such, following the observation as per inequality (19), we indeed need to set α_t as follows:

$$\sigma_t + (k - K\sigma_t)\frac{(1-\sigma_t)\alpha_t}{\sum_{j\in\mathcal{K}} w'_{j,t}} = 1. \tag{20}$$

By this equation, we maximize α_t while making $p_{i,t} \leq 1$ for any client i, and it can be simplified as follows:

$$\frac{\alpha_t}{\sum_{j\in\mathcal{K}} w'_{j,t}} = \frac{1}{k - K\sigma_t}. \tag{21}$$

Then, we show the way to find the solution of (21). First, it is noticeable that the possible "structures" of $\sum_{j\in\mathcal{K}} w'_{j,t}$ are finite. Then, we can simply divide it into at most $N-1$ cases. Let $\Psi_{i,t} = w_{i,t}/(1-\sigma_t)$. Formally, we assume *case* v satisfying: $\Psi_{i_v,t} \leq \alpha_t < \Psi_{i_{v+1},t}$ where i_v denotes the vth smallest $\Psi_{i,t}$. As the structure of $\sum_{j\in\mathcal{K}} w'_{j,t}$ is fixed for case v, we can derive

$$\sum_{w_{j,t}\leq\Psi_{i_v,t}} \frac{w_{j,t}}{\alpha_t} + \sum_{w_{j,t}>\Psi_{i_v,t}} (1-\sigma_t) = k - K\sigma_t. \tag{22}$$

Reorganizing the term, we have

$$\alpha_t = \frac{\sum_{j\in\mathcal{K}: w_{j,t}\leq\Psi_{i_v,t}} w_{j,t}}{k - K\sigma_t - \sum_{w_{j,t}>\Psi_{i_v,t}}(1-\sigma_t)}. \tag{23}$$

By iterating all the cases and checking if the calculated α_t satisfies the premise, i.e., $\Psi_{i_v,t} \leq \alpha_t < \Psi_{i_{v+1},t}$, finally we are allowed to derive a feasible α_t.

[8]Too see why we make this statement, considering the case when setting $\alpha_t \to 0$, then the allocation differentiation is completely eliminated (i.e., all the clients share the same probability), which contradicts our tenet to give more chances to the stable contributors.

Algorithm 1 Exp3-Based Client Selection (E3CS) for FL

Input:
The number of involved clients each round; k
Fairness quota; $\{\sigma_t\}$
Final round; T
Local data distribution; $\{\mathcal{D}_i\}$
Local update operation; $o_1(\cdot)$

Output:
Global network weights; $\boldsymbol{\Theta}_{T+1}$

Initialize $w_{i,1} = 1$ for $i = 1, 2, \ldots, K$
for $t = 1, 2, \ldots, T$ **do**
  $\boldsymbol{p}_t, S_t = \text{ProbAlloc}(k, \sigma_t, \{w_{i,t}\})$
  $A_t \sim \text{multinomialNR}(\boldsymbol{p}_t/k, k)$
  for client $i \in A_t$ **in parallel do**
    $\boldsymbol{\Theta}_{i,t} = o_1(\boldsymbol{\Theta}_t, \mathcal{D}_i)$
  end for
  for client $i \in [K]$, $x_{i,t} = 1$ if succeed; else, $x_{i,t} = 0$
  $\boldsymbol{\Theta}_{t+1} = \sum_{i\in A_t \text{and} x_{i,t}\neq 0} \omega_i \boldsymbol{\Theta}_{i,t} + \sum_{i\notin A_t \text{or} x_{i,t}=0} \omega_i \boldsymbol{\Theta}_t$
  $\hat{x}_{i,t} = \frac{\mathbb{I}_{\{i\in A_t\}}}{p_{i,t}} x_{i,t} \quad i \in \mathcal{K}$
  for $i \in \mathcal{K}$:

$$w_{i,t+1} = \begin{cases} w_{i,t}\exp\left(\frac{(k-K\sigma_t)\eta\hat{x}_{i,t}}{K}\right) & i \notin S_t \\ w_{i,t} & i \in S_t \end{cases}$$

end for

C. Algorithm

To enable a better understanding of our proposed selection solution, we present the detailed procedure of our Exp3-based client selection (E3CS) for FL in Algorithm 1. Explicitly, E3CS runs in the following procedure.

Initialization: Initialize the exponential weights of all the clients to 1 in the first round and then the algorithm formally goes into the iterative training process.

Probability Allocation: In each iteration, Algorithm 2, which calculates the probability allocation as per we describe in Section V-B, is called into execution and return the probability allocation $\{p_{i,t}\}$ and overflowed client set S_t.

Stochastic Selection: Then, our incoming task is to randomly select the clients based on the obtained probability. This specific task is done by drawing samples from a weighted multinomial distribution with no replacement for k times. In our implementation, we simply use the API in PyTorch,[9] i.e., torch.multinomial($\boldsymbol{p}_t$, k, replacement=False).

Local Training and Aggregation: Once A_t is drawn from the multinomial distribution, our main algorithm would distribute the global model $\boldsymbol{\Theta}_t$ to the selected clients, who will immediately start local training (e.g., conduct multiple steps of SGD). This local training process could be different by using different optimizers (or adding a proximal term to the local loss function as FedProx [11] does). At the same time, the server's main process sleeps until the deadline is met, and after that, models would be aggregated based on the returned models' parameters. Specifically, clients with successful returns would

[9]Similar implementation of multinomial sampling is also available in scipy, numpy, and R.

Algorithm 2 Probability Allocation (ProbAlloc)

Input:
The number of involved clients each round; k
Fairness quota; $\{\sigma_t\}$
Exponential Weight for round t: $\{w_{i,t}\}$
Output:
Probability allocation vector for round t: $\boldsymbol{p}_t$
Overflowed set for round t: S_t

1: **if** $\sigma_t + (k - K\sigma_t)\frac{\max_{i\in\mathcal{K}}\{w_{i,t}\}}{\sum_{j\in\mathcal{K}} w_{j,t}} > 1$ **then**
2: Decide α_t such that: $\frac{\alpha_t}{\sum_{j\in\mathcal{K}} w'_{j,t}} = \frac{1}{k-K\sigma_t}$
3: $S_t = \{i \in \mathcal{K} : w_{i,t} > (1-\sigma_t)\alpha_t\}$
4: $w'_{i,t} = \min\{w_{i,t}, (1-\sigma_t)\alpha_t\}$ for $i = 1, 2, \ldots, K$
5: $p_{i,t} = \sigma_t + (k - K\sigma_t)\frac{w'_{i,t}}{\sum_{j\in\mathcal{K}} w'_{j,t}}$ for $i = 1, 2, \ldots, K$
6: **else**
7: $S_t = \emptyset$
8: $p_{i,t} = \sigma_t + (k - K\sigma_t)\frac{w_{i,t}}{\sum_{j\in\mathcal{K}} w_{j,t}}$ for $i = 1, 2, \ldots, K$
9: **end if**
10: Return $\boldsymbol{p}_t$, S_t

be involved in aggregation, and for those clients whose models are not successfully returned on time, or simply not being chosen into training, the global model would take its place in aggregation (see line 9 in Algorithm 1).

Exponential Weights Update: After the aggregation is done, we need to update our "expectation" of clients (or arms). This part of update process is consistent with (15) and (16).

D. Theoretical Regret Guarantee

In this section, we might need to evaluate our proposed solutions at a theoretical level. Let us first define an optimal solution of $P3$.

Definition 1 (Optimal Solution): The optimal solution for P3 performs client selection based on the following probability allocation:

$$p^*_{i,t} = q^*_{i,t}(k - K\sigma_t) + \sigma_t \tag{24}$$

where $q^*_{i,t}$ is the optimal allocation quota of $k-K\sigma_t$ probability.

The definition is quite intuitive. We simply reserve σ_t probability for each client and optimally allocate the residual $k-K\sigma_t$ probability. Based on our definition, it is not hard to derive the optimal expected CEP until round T (CEP^*_T)

$$\mathbb{E}\big[\text{CEP}^*_T\big] = \sum_{t=1}^{T}\sum_{i\in\mathcal{K}}\big(q^*_{i,t}(k - K\sigma_t) + \sigma_t\big)x_{i,t}. \tag{25}$$

Given $\mathbb{E}[\text{CEP}^*_T]$, we like to compare the performance of E3CS with the optimal solution, which motivates us to give the following definition of regret.

Definition 2 (Regret of E3CS): Given $\mathbb{E}[\text{CEP}^*_T]$, the regret of E3CS (or performance gap to the optimal) is given by

$$R_T = \mathbb{E}\big[\text{CEP}^*_T\big] - \mathbb{E}\Big[\text{CEP}^{\text{E3CS}}_T\Big] \tag{26}$$

where $\mathbb{E}[\text{CEP}^{\text{E3CS}}_T] = \sum_{t=1}^{T}\sum_{i=1}^{K} p_{i,t}x_{i,t}$.

Now, we introduce an upper bound of the defined regret, as in Theorem 1.

Theorem 1 (Upper Bound of Regret): The regret of E3CS is upper bounded by

$$R_T \le \eta\sum_{t=1}^{T}(k - K\sigma_t) + \frac{K}{\eta}\ln K \tag{27}$$

and if $\eta = \sqrt{([K\ln K]/[\sum_{t=1}^{T}(k - K\sigma_t)])}$, we have

$$R_T \le 2\sqrt{\sum_{t=1}^{T}K(k - K\sigma_t)\ln K}. \tag{28}$$

Proof: Complete proof is available in Appendix B in the supplementary material. ■

Remark 1: Notably, the regret of E3CS would diminish to 0 as $\sigma_t \to k/K$. This phenomenon can be explained by looking into what happens if $\sigma_t = k/K$: both E3CS and the optimal solution yield the same solution, which is essentially an even random selection among clients, so the regret would be exactly 0. At the other extreme when $\sigma_t = 0$ for all t, the regret would be reduced to $R_T \le 2\sqrt{TKk\ln K}$ if $\eta = \sqrt{[(K\ln K)/Tk]}$. In this case, the derived bound of regret under multiple play setting is $\sqrt{k}$ times of that of the canonical single play Exp3 (see [28, Th. 11.1])

VI. Experiments

In this section, we shall present the experimental results of our proposed client selection solution. The evaluation is based on numerical simulation and real training in two iconic public data sets: 1) EMNIST-Letter [29] and 2) CIFAR-10 [5].

A. Setup

In this section, we shall illustrate the basic setup of our experiment.

Programming and Running Environment: We have implemented E3CS and the volatile training feature to an open-source lightweighted FL simulator named FlSim,[10] which is built on the basis of PyTorch and has efficiently implemented parallel training for the learning process. In terms of the runtime environment, all the computation in our simulation is run by a high-performance workstation (Dell PowerEdge T630 with 2xGTX 1080Ti).

Simulation of Volatile and Heterogeneous Clients: In our simulation, a total number of $K = 100$ volatile clients are available for selection, and in each round, $k = 20$ of which are being selected. Under a volatile training context, clients might suffer unexpected drop-out during their training. In our experiment, we adopt a Bernoulli distribution to simulate the training status of clients. Formally, we have $x_{i,t} \sim \text{Bern}(\rho_i)$ where ρ_i is the success rate of client i. To simulate the *heterogeneous volatility* of clients, we equally divide the whole set of clients into four classes, with the success rate, respectively, set as 0.1, 0.3, 0.6, and 0.9. In addition, we also allow the local training epochs of clients to be different to simulate the client's *heterogeneous computing capacity*. Specifically, we designate

[10]Source code available in https://github.com/iQua/flsim.

TABLE I
PARAMETERS SETTING OF FL TRAINING. NOTE: 1) SYMBOLS THAT DO NOT APPEAR IN OTHER PLACES OF THIS ARTICLE WOULD BE WRITTEN AS "−" AND 2) SGD IS ABBREVIATION OF STANDARD STOCHASTIC GRADIENT DESCENT

Setting	Symbols	Task 1	Task 2
Dataset	-	EMNIST-Letter	CIFAR-10
# of labels	-	26	10
Max # of rounds	T	400	2500
# of data per client	$\|D_i\|$	500	500
Learning rate of E3CS	η	0.5	0.5
Optimizer	-	SGD	SGD
Learning rate of optimizer	-	$1e{-}2$	$1e{-}2$
Momentum	-	0.9	0.9
Local epoches	E_i	{1,2,3,4}	{1,2,3,4}
Mini-batch size	B	40	40

different training epochs for different clients, and the designated epochs are randomly chosen from the set {1, 2, 3, 4}. But note that we do not introduce correlation between *heterogeneous volatility* and *heterogeneous computing capacity*. That is, a client that is asked to perform more rounds of training does not necessarily lead to a higher or lower failure rate. These two properties are assumed to be independent in our analyzed scenario.

Simulation of Data Distribution: We simulate the data distribution by both iid and non-iid setting:[11] 1) for an iid one, each client independently samples $|\mathcal{D}_i|$ pieces of data from the data set and 2) for a non-iid distribution, we randomly select one primary label for each client. Then, we sample $0.8 \times |\mathcal{D}_i|$ pieces of data from those that coincide with their primary label and $0.2 \times |\mathcal{D}_i|$ from those with the remaining labels. For both the iid and non-iid settings, each client randomly reserves 10% of the data for testing.

Data Sets and Network Structure: To evaluate the training performance, we prepare two tasks for FL to conquer: 1) EMNIST-Letter and 2) CIFAR-10. For EMNIST-Letter, we employ a CNN with two 5×5 convolution layers (each with ten channels), followed by 2×2 max pooling and two fully connected layers with respective 1280 and 256 units, and finally, a softmax output layer. For CIFAR-10, which is known to be a harder task, we use another CNN model with two 5×5 convolution layers (each with 64 channels), also followed with 2×2 max pooling, two fully connected layers with 384 and 192 units, and finally, a softmax output layer. Other training-related setting is available in Table I.

Baselines and Implementation Details: We prepare three state-of-the-art client selection scheme, namely, FedCS [13], Random [1], and pow-d [3] for baselines. Detailed description of these baselines is available in Appendix A in the supplementary material. Based on different settings of σ_t, we like to evaluate the performance of the following selection schemes.

1) *E3CS-(number):* Fairness quota σ_t of E3CS is stationary and fixed to (number) $\times\, k/K$. In this setting, the fairness constraint would be ineffective and the algorithm would seek to maximize the eCEP regardless of fairness.
2) *E3CS-inc:* We allow fairness quota σ_t to be incremental with training round t. More explicitly, we let $\sigma_t = k/K$ if $T \geq t > T/4$ and $\sigma_t = 0$ if $1 \leq t \leq T/4$. The motivation of our setting is that we want the algorithm to gain more effective participation in the beginning stage, but when training is approaching convergence, it should be better to expand the selection scope (i.e., to be fair) in order to improve the final accuracy. We will formally discuss the motivation behind this setting after we present the training result.

Besides, as we note that we in this article only focus on the client selection subproblem, we need to borrow the update schemes (see o_1 in our global problem *P1*) from the state-of-the-art methods. Explicitly, we mimic these two schemes.

1) *FedAvg [1]:* The local update scheme in FedAvg is SGD with cross-entropy loss function.
2) *FedProx [11]:* The local update scheme in FedProx is also SGD, but with a regularization involved loss function. It adds a proximal term $(\gamma/2)\|\boldsymbol{\Theta}_{i,t} - \boldsymbol{\Theta}_t\|^2$ into the typical cross-entropy loss.

In our simulation, we use "scheme+(A)" to denote the selection scheme with update operation in FedAvg [e.g., Random(A), E3CS-0(A), etc]. Similarly, we use "scheme+(P)" to denote the scheme with the same operation with FedProx [e.g., Random(P), E3CS-0(P), etc]. For all the FedProx-based schemes, the proximal coefficient is set to $\gamma = 0.5$.

B. Numerical Simulation

In this section, we present the numerical evaluation to show how different selection algorithms perform in terms of important statistics, such as effective participation, times of selection, and we further discuss the results combining the working patterns of these algorithms.

By running 2500 rounds of simulation step, we obtain the selection records of different selection schemes over different groups of clients. The results are formally displayed in several equal-scale box plots (see Fig. 3). From the figure, we can derive the following observations.

Effectiveness of E3CS in Choosing Reliable Clients: One observation is that E3CS algorithms are able to learn the most reliable clients by only a modicum number of tries. E3CS-0, a scheme that has no regard for fairness, only wrongly selects suboptimal classes of clients dozens of times over a total of 2500 rounds of selection. Besides, by comparison of E3CS-0 and FedCS, we can observe another unexpected advantage of E3CS, i.e., a certain degree of fairness would be reserved if a sufficient amount of clients shares the same (or near) high success ratio. Take our setting as an example. In our setting,

[11] In FL, each client only has a small number of data for training. To mimic the federated setting, we split the whole training data set (like 50 000 images in CIFAR-10) into small portions (like 500 images per client), and send each portion to different clients to simulate distributed training environment. Also, in FL, the data of clients are usually heterogeneous. For example, consider a simple MNIST FL task, in which the data of a client are mostly labeled as "1," while that of another is mostly labeled as "2." Then, two of the clients are said to have non-iid data, and a similar data distribution like this is pretty common in reality (due to personalization). As such, we attempt to simulate FL in both iid (the ideal learning case) and non-iid setting (the more realistic case).

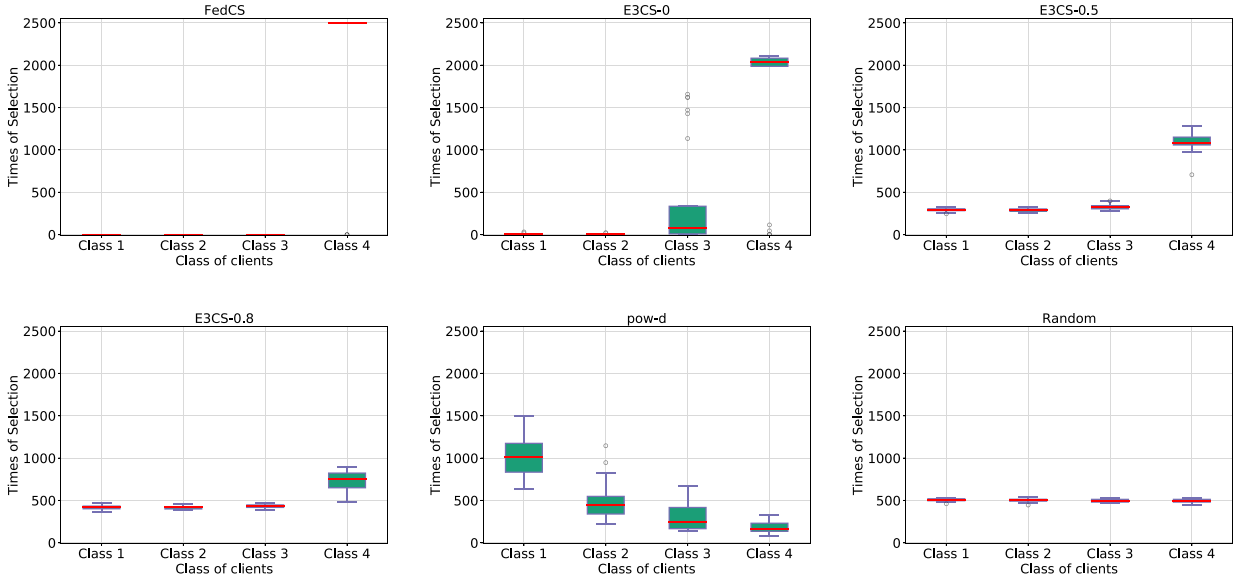


Fig. 3. Under different selection schemes, the box plots display times of selection over four classes of clients, respectively, with success ratio 0.1, 0.3, 0.6, 0.9, and an equal division of a total number of 100 clients. The total simulated communication rounds are all set to 2500 and in each round, 20 clients are to be selected. Note that for the FedCS scheme, all the selections are dedicated toward 20 of 25 class 4 clients, so the interquartile range (IQR) of class 4 (which is not displayed in the plot) is 2500, while the IQR of other classes is 0.

FedCS would consistently choose specific 20 out of 25 clients belonging to Class 4, but in contrast, E3CS-0 would share the most probability over all the 25 clients in Class 4 while giving minor probability (resulting from the cost of learning) to others classes of clients. This selection mechanism gives some degree of fairness while not necessarily sacrificing a lot in terms of CEP. It can be observed that pow-d is prone to select the clients which are more likely to fail, which is directly against the pattern of E3CS and FedCS. This phenomenon can be explainable by the following analysis: 1) by its objective, we know that pow-d tends to select the clients with higher losses and 2) those clients that are more likely to fail typically have a higher loss, since their local model has less chance to be aggregated into the global model. Consequently, clients with higher failure probability are more favored by pow-d.

Selection Fairness: As depicted, all the schemes barring Random and pow-d would deliver more selection frequency to the clients in Class 4, who boast the highest success rate. By this biased selection, CEP could be maximized, but fairness degree might correspondingly hurt. Obviously, the order of fairness degree among all the selection schemes is: Random $>$ E3CS-0.8 $>$ pow-d $>$ E3CS-0.5 $>$ E3CS-0 $>$ FedCS. This particular order of fairness might serve a critical function to reach effective training, which would be specified later.

Success Ratio and CEP: We define two metrics to evaluate the proposed solutions: 1) success ratio, explicitly formulated as $\sum_{t=1}^{T}\sum_{i\in A_t} x_{i,t}/Tk$ and 2) the CEP, formulated as, $\sum_{t=1}^{T}\sum_{i\in A_t} x_{i,t}$. We present the evolvement of these two quantities with communication rounds, as depicted in Fig. 4. From the top subfigure, it is notable that the success ratio of all the E3CS (except E3CS-inc) shares a similar trend of convergence and the final convergence value is largely determined by the constant setting of σ_t, indicating that fairness quota

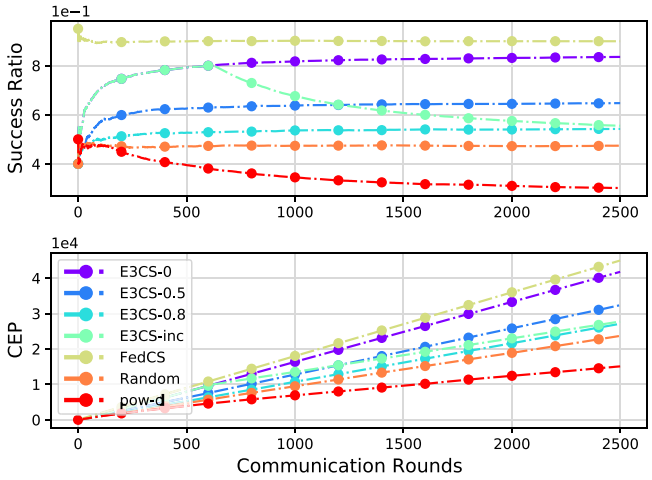


Fig. 4. Communication rounds versus success ratio and CEP for different selection schemes.

has a negative correlation on training success ratio. This phenomenon is well justified since a larger fairness quota might necessarily expand the selection of clients that are prone to fail. Another interesting observation is that E3CS-inc undertakes a significant turn in Round 625 i.e., the exact round of $4/T$. This phenomenon is as expected since E3CS-inc essentially reduces to an unbiased random selection scheme after $4/T$ and therefore, its success ratio would be correspondingly plunged. Theoretically, E3CS-inc will eventually converge to the convergence value of Random if T goes sufficiently large. Also, it is interesting to find that pow-d has a very low CEP and success ratio throughout the whole training session. This experimental result is in accordance with our previous analysis, in that pow-d is prone to select clients with lower success rates.

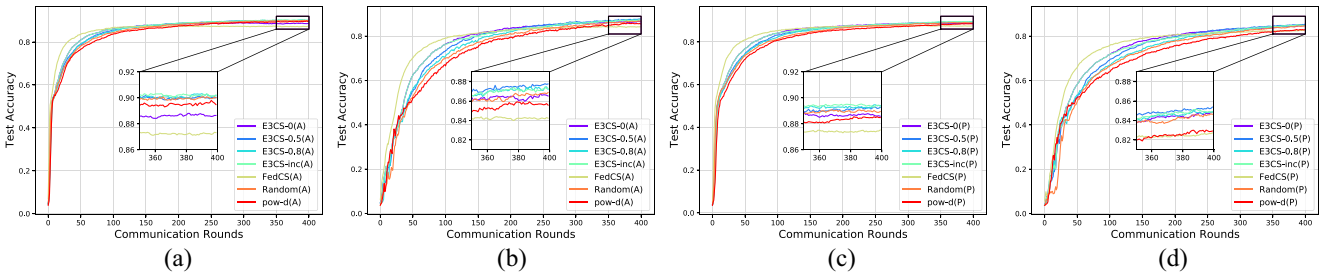


Fig. 5. Test accuracy versus communication rounds for EMNIST-Letter. (a) iid, FedAvg-based. (b) non-iid, FedAvg-based. (c) iid, FedProx-based. (d) non-iid, FedProx-based.

TABLE II
PERFORMANCE EVALUATION FOR EMNIST-LETTER. NOTE: 1) ACCURACY@*Number* REPRESENTS THE FIRST ROUND TO REACH A CERTAIN TEST ACCURACY AND 2) NAN MEANS THAT THE ACCURACY NEVER REACH THE CORRESPONDING SETTING OVER THE TRAINING SESSION (400 ROUNDS)

Methods	Accuracy@**65**		Accuracy@**75**		Accuracy@**85**		Final Accuracy (%)	
	iid	non-iid	iid	non-iid	iid	non-iid	iid	non-iid
E3CS-0(A)	18	58	32	94	82	257	88.64	86.52
E3CS-0.5(A)	20	67	38	107	94	**243**	90.04	**87.76**
E3CS-0.8(A)	21	75	39	117	98	272	90.18	87.2
E3CS-inc(A)	18	58	32	94	82	268	**90.22**	87.3
FedCS(A)	**11**	**39**	**20**	**69**	**66**	NaN	87.32	84.24
Random(A)	21	78	40	131	107	312	89.98	86.88
pow-d(A)	24	92	42	148	119	339	89.48	85.58
E3CS-0(P)	24	69	45	120	112	NaN	88.58	84.82
E3CS-0.5(P)	27	82	52	140	131	**375**	89.26	**85.38**
E3CS-0.8(P)	30	89	53	157	137	387	**89.4**	84.9
E3CS-inc(P)	24	69	45	132	127	386	89.34	84.84
FedCS(P)	**15**	**48**	**28**	**93**	**93**	NaN	87.52	82.72
Random(P)	29	92	57	167	157	NaN	88.94	84.64
pow-d(P)	34	104	63	192	171	NaN	88.48	82.92

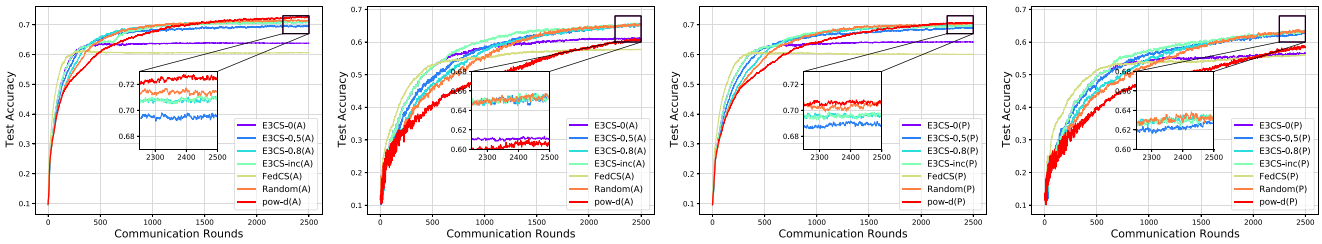


Fig. 6. Test accuracy versus communication rounds for CIFAR-10. (a) iid, FedAvg. (b) non-iid, FedAvg. (c) iid, FedProx. (d) non-iid, FedProx.

C. Real Training on Public Data Set

We then show our experimental results of real training on public data set EMNIST-Letter [29] and CIFAR-10 [5].

We, respectively, ran 400 and 2500 communication rounds for EMNIST-L and CIFAR-10 data sets over different selection schemes in order to evaluate the convergence speed as well as the final model performance in the real training. Respectively for the two data sets, we depict in Figs. 5 and 6 the training performance of different schemes under both iid and non-iid scenario, and we further present some statistical data in Tables II and III. Based on the experimental results, we derive the following observations.

Impact of CEP in the Initial Stage: It is interesting to see that CEP has a conspicuous impact on convergence speed in the initial training stage. For the FedAvg-based solution, we see that FedCS(A), which consistently chooses the clients with higher success rate, and is confirmed by Fig. 3 to be the selection scheme with the highest CEP, obtains the fastest growth of accuracy during the first stage of training for both EMNIST-Letter and CIFAR-10 data set. It is followed closely by E3CS-0(A) and E3CS-inc(A), which are also confirmed as having relatively high CEP in that stage. The gap of convergence speed over different schemes can also be observed in Tables II and III, in which we found that E3CS-0(A),

TABLE III
PERFORMANCE EVALUATION FOR CIFAR-10. NOTE: 1) DATA UNDER ACCURACY@*Number* REPRESENTS THE FIRST ROUND TO REACH A CERTAIN TEST ACCURACY AND 2) NAN MEANS THE ACCURACY NEVER REACHES THE CORRESPONDING SETTING OVER THE WHOLE TRAINING SESSION

Methods	Accuracy@**45**		Accuracy@**55**		Accuracy@**65**		Final Accuracy (%)	
	iid	non-iid	iid	non-iid	iid	non-iid	iid	non-iid
E3CS-0(A)	89	307	164	636	NaN	NaN	63.8	61.0
E3CS-0.5(A)	106	362	198	767	468	2252	69.48	**65.6**
E3CS-0.8(A)	123	429	228	849	**454**	2279	70.94	65.42
E3CS-inc(A)	89	307	170	**631**	650	**2127**	71.12	65.5
FedCS(A)	**59**	**246**	**119**	817	NaN	NaN	60.6	57.78
Random(A)	121	472	251	930	515	2156	71.46	65.18
pow-d(A)	126	689	328	1517	742	NaN	**72.52**	60.54
E3CS-0(P)	130	374	252	1130	NaN	NaN	64.18	56.56
E3CS-0.5(P)	160	406	292	863	749	NaN	68.98	62.7
E3CS-0.8(P)	183	499	356	995	720	NaN	69.62	63.04
E3CS-inc(P)	130	374	252	**710**	**698**	NaN	69.68	**63.48**
FedCS(P)	**93**	**285**	**189**	1822	NaN	NaN	60.06	56.08
Random(P)	201	533	392	1155	806	NaN	**70.54**	62.94
pow-d(P)	208	736	483	1928	1083	NaN	70.38	58.48

E3CS-inc(A), and FedCS(A) all have been accelerated to reach a certain fixed accuracy, compared with the vanilla selection scheme Random. In contrast, pow-d, which we confirm to have a relatively low CEP, does not promise us a commensurate convergence speed in our simulated volatile context. This may imply that when clients can drop out, the heuristic idea of always selecting the clients with higher loss might not necessarily accelerate convergence. Moreover, the impact on convergence speed seems to grow more significant in non-iid scenarios, as the gap between the "fastest" [i.e., FedCS(A)] and the "slowest" [i.e., pow-d(A)] has further expanded under this trend. Based on such an observation and since it is generally believed that non-iid data would further enhance the training difficulty, we conjecture that the difficulty of the task might have some sort of influence on CEP's impact on convergence speed, i.e., the more difficult the task is, the greater the influence of CEP will be. This conjecture is also aligned with another observation that the impact of CEP in accelerating the training is more significant for training on the CIFAR-10 data set, a harder task compared with training on EMNIST-L. Similar observations can also be found for FedProx-based solutions, as depicted Figs. 5(c) and (d) and 6(c) and (d).

Diminished Impact of CEP: However, the effect of CEP diminishes in the middle/later stage of training. When the accuracy reaches a certain accuracy, aggregating more successful returns does not benefit much to the FL process, as we can observe that other fairer schemes, e.g., Random(A), though with a smaller CEP, gradually emulates FedCS(A) and eventually dominates it with the evolvement of the training process. A similar phenomenon is also observable for FedProx-based training.

Impact of Fairness: The impact of the fairness factor is visually observable when training reaches its convergence. For all the experimental groups, we see that the final test accuracy of the most unfair scheme, FedCS, is the lowest among the evaluated methods. The method with the second-lowest accuracy is E3CS-0, which is also quite an "unfair" selection scheme, as it does not reserve probability for each client to ensure fairness.

Motivation Behind Incremental Fairness Quota: Based on the above observations, we find that: 1) CEP is critical for the initial stage of training in order to yield a faster convergence speed, but; 2) the effect diminishes with the training rounds goes, and that; 3) the importance of fairness reinforces when the model approaches convergence. Our motivation to propose E3CS-inc is exactly based on the above observations. In E3CS-inc, we set $\sigma_t = 0$ during the first $T/4$ rounds, so the algorithm will have no regard for fairness and put all the focus on increasing CEP. For the later $3T/4$ rounds, when the model has some sort of "overfitting" on a portion of frequently selected data, we expand the selection fairness by making $\sigma_t = k/K$ (which makes it exactly an unbiased random selection), in order to make data on those seldom access clients being available for FL training. According to our result, E3CS-inc yields a performance as we have expected: it gets a very promising convergence speed in the first stage while its final test accuracy does not suffer an undesirable drop.

Varying Selection Cardinality: We test different selection schemes based on varying selection cardinality $k = 10$, 20, and 30. Due to the space limit, this part of content has been moved to Appendix C in the supplementary material.

VII. CONCLUSION AND FUTURE PROSPECT

In this article, we have studied a joint optimization problem under the volatile training context. During our investigation, we empirically discovered a tradeoff between CEP and fairness during the selection process, which in essence leads to another tradeoff between training convergence speed and final model accuracy. Aiming at optimizing the tradeoffs, we proposed

E3CS, an efficient stochastic selection algorithm, for which we further designed a practical setting of "fairness quota," such that the algorithm is enabled to tame the tradeoff between training convergence speed and final model accuracy.

In this article, we proposed to decompose the global problem P1 into two subproblems based on the idea of alternating minimization. In our solution process, we focused on optimization of the client selection subproblem (i.e., P1-SUB2) while fixing the solution of another subproblem (i.e., P1-SUB1). However, as all the two essential components in FL, i.e., the local update operation and the client selection decision, are mutually coupling, joint optimization of them needs to be further considered in future work.

REFERENCES

[1] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas, "Communication-efficient learning of deep networks from decentralized data," 2016, *arXiv:1602.05629*.

[2] T. Huang, W. Lin, W. Wu, L. He, K. Li, and A. Y. Zomaya, "An efficiency-boosting client selection scheme for federated learning with fairness guarantee," *IEEE Trans. Parallel Distrib. Syst.*, vol. 32, no. 7, pp. 1552–1564, Jul. 2021.

[3] Y. J. Cho, J. Wang, and G. Joshi, "Client selection in federated learning: Convergence analysis and power-of-choice selection strategies," 2020, *arXiv:2010.01243*.

[4] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated learning with non-IID data," 2018, *arXiv:1806.00582*.

[5] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Dept. Comput. Sci., Univ. Toronto, Toronto, ON, Canada, Rep. TR-2009, 2009.

[6] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[7] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of fedavg on non-IID data," 2019, *arXiv:1907.02189*.

[8] N. Yoshida, T. Nishio, M. Morikura, K. Yamamoto, and R. Yonetani, "Hybrid-FL for wireless networks: Cooperative learning mechanism using non-IID data," in *Proc. IEEE Int. Conf. Commun. (ICC)*, 2020, pp. 1–7.

[9] Y. Sun, S. Zhou, and D. Gündüz, "Energy-aware analog aggregation for federated learning with redundant data," in *Proc. IEEE Int. Conf. Commun. (ICC)*, 2020, pp. 1–7.

[10] E. Jeong, S. Oh, H. Kim, J. Park, M. Bennis, and S.-L. Kim, "Communication-efficient on-device machine learning: Federated distillation and augmentation under non-IID private data," 2018, *arXiv:1811.11479*.

[11] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," 2018, *arXiv:1812.06127*.

[12] Y. Ruan, X. Zhang, S.-C. Liang, and C. Joe-Wong, "Towards flexible device participation in federated learning," in *Proc. Int. Conf. Artif. Intell. Stat.*, 2021, pp. 3403–3411.

[13] T. Nishio and R. Yonetani, "Client selection for federated learning with heterogeneous resources in mobile edge," in *Proc. IEEE Int. Conf. Commun. (ICC)*, 2019, pp. 1–7.

[14] Q. Zeng, Y. Du, K. K. Leung, and K. Huang, "Energy-efficient radio resource allocation for federated edge learning," Jul. 2019, *arXiv:1907.06040*.

[15] J. Xu and H. Wang, "Client selection and bandwidth allocation in wireless federated learning networks: A long-term perspective," 2020, *arXiv:2004.04314*.

[16] H. Wang, Z. Kaplan, D. Niu, and B. Li, "Optimizing federated learning on non-IID data with reinforcement learning," in *Proc. IEEE INFOCOM Conf. Comput. Commun.*, 2020, pp. 1698–1707.

[17] N. Yoshida, T. Nishio, M. Morikura, and K. Yamamoto, "MAB-based client selection for federated learning with uncertain resources in mobile networks," 2020, *arXiv:2009.13879*.

[18] W. Xia, T. Q. S. Quek, K. Guo, W. Wen, H. H. Yang, and H. Zhu, "Multi-armed bandit-based client scheduling for federated learning," *IEEE Trans. Wireless Commun.*, vol. 19, no. 11, pp. 7108–7123, Nov. 2020.

[19] F. Lai, X. Zhu, H. V. Madhyastha, and M. Chowdhury, "Oort: Efficient federated learning via guided participant selection," 2020, *arXiv:2010.06081*.

[20] W. Chen, S. Horvath, and P. Richtarik, "Optimal client sampling for federated learning," 2020, *arXiv:2010.13723*.

[21] J. Wangni, J. Wang, J. Liu, and T. Zhang, "Gradient sparsification for communication-efficient distributed optimization," 2017, *arXiv:1710.09854*.

[22] M. Mohri, G. Sivek, and A. T. Suresh, "Agnostic federated learning," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 4615–4625.

[23] T. Li, M. Sanjabi, A. Beirami, and V. Smith, "Fair resource allocation in federated learning," 2019, *arXiv:1905.10497*.

[24] W. Wu, L. He, W. Lin, R. Mao, C. Maple, and S. Jarvis, "SAFA: A semi-asynchronous protocol for fast federated learning with low overhead," *IEEE Trans. Comput.*, vol. 70, no. 5, pp. 655–668, May 2021.

[25] W. Wu, L. He, W. Lin, and R. Mao, "Accelerating federated learning over reliability-agnostic clients in mobile edge computing systems," *IEEE Trans. Parallel Distrib. Syst.*, vol. 32, no. 7, pp. 1539–1551, Jul. 2021.

[26] Y. Fraboni, R. Vidal, L. Kameni, and M. Lorenzi, "Clustered sampling: Low-variance and improved representativity for clients selection in federated learning," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 3407–3416.

[27] T. Uchiya, A. Nakamura, and M. Kudo, "Algorithms for adversarial bandit problems with multiple plays," in *Proc. Int. Conf. Algorithmic Learn. Theory*, 2010, pp. 375–389.

[28] T. Lattimore and C. Szepesvári, *Bandit Algorithms*. Cambridge, U.K.: Cambridge Univ. Press, 2020.

[29] G. Cohen, S. Afshar, J. Tapson, and A. Van Schaik, "EMNIST: Extending MNIST to handwritten letters," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, 2017, pp. 2921–2926.

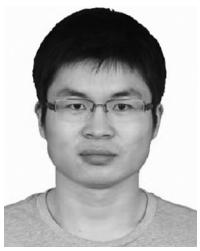
Tiansheng Huang is currently pursuing the master's degree with the School of Computer Science and Engineering, South China University of Technology, Guangzhou, China.

His research interests include parallel and distributed computing, distributed machine/federated learning, and system design.

Weiwei Lin (Member, IEEE) received the B.S. and M.S. degrees from Nanchang University, Nanchang, China, in 2001 and 2004, respectively, and the Ph.D. degree in computer application from South China University of Technology, Guangzhou, China, in 2007.

He has been serving as a Visiting Scholar with Clemson University, Clemson, SC, USA, from 2016 to 2017. He is currently a Professor with the School of Computer Science and Engineering, South China University of Technology. He has published more than 100 papers in refereed journals and conference proceedings. His research interests include distributed systems, cloud computing, big data computing, and AI application technologies.

Prof. Lin has been the reviewer for many international journals, including IEEE TRANSACTIONS ON COMPUTERS, IEEE TRANSACTIONS ON SERVICES COMPUTING, IEEE TRANSACTIONS ON CLOUD COMPUTING, IEEE TRANSACTIONS ON SUSTAINABLE COMPUTING, *Information Sciences*, and *Future Generation Computer Systems*. He is a Senior Member of CCF.



Li Shen received the Ph.D. degree from the School of Mathematics, South China University of Technology, Guangzhou, China, in 2017.

He is currently a Research Scientist with the JD Explore Academy, Beijing, China. Previously, he was a Research Scientist with the Tencent AI Laboratory, Shenzhen, China. His research interests include theory and algorithms for large-scale convex/nonconvex/minimax optimization problems, and their applications in statistical machine learning, deep learning, reinforcement learning, and game theory.

Keqin Li (Fellow, IEEE) received the B.S. degree in computer science from Tsinghua University, Beijing, China, in 1985, and the Ph.D. degree in computer science from the University of Houston, Houston, TX, USA.

He is a SUNY Distinguished Professor of Computer Science with the State University of New York at New Paltz, New Paltz, NY. He is also a Distinguished Professor with Hunan University, Changsha, China. He has published over 640 journal articles, book chapters, and refereed conference papers. His current research interests include cloud computing, fog computing and mobile-edge computing, energy-efficient computing and communication, embedded systems and cyber–physical systems, heterogeneous computing systems, big data computing, high performance computing, CPU-GPU hybrid and cooperative computing, computer architectures and systems, computer networking, machine learning, and intelligent and soft computing.

Dr. Li has received several best paper awards. He is currently or has served on the editorial boards of the IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, IEEE TRANSACTIONS ON COMPUTERS, IEEE TRANSACTIONS ON CLOUD COMPUTING, IEEE TRANSACTIONS ON SERVICES COMPUTING, and IEEE TRANSACTIONS ON SUSTAINABLE COMPUTING.

Albert Y. Zomaya (Fellow, IEEE) received the B.S. degree in electrical engineering from Kuwait University, Kuwait City, Kuwait, in 1987, and the Ph.D. degree in control engineering from Sheffield University, Sheffield, U.K., in 1990.

He is currently the Chair Professor of High Performance Computing and Networking with the School of Information Technologies, The University of Sydney, Sydney NSW, Australia, where he is also the Director of the Centre for Distributed and High Performance Computing, which was established in late 2009. He published more than 500 scientific papers and articles and is a author, coauthor, or editor of more than 20 books. His research interests are in the areas of parallel and distributed computing and complex systems.

Dr. Zomaya received the IEEE Technical Committee on Parallel Processing Outstanding Service Award in 2011, the IEEE Technical Committee on Scalable Computing Medal for Excellence in Scalable Computing in 2011, and the IEEE Computer Society Technical Achievement Award in 2014. He served as the Editor-in-Chief of the IEEE TRANSACTIONS ON COMPUTERS from 2011 to 2014. He serves as an Associate Editor for 22 leading journals, such as the *ACM Computing Surveys*, IEEE TRANSACTIONS ON COMPUTATIONAL SOCIAL SYSTEMS, IEEE TRANSACTIONS ON CLOUD COMPUTING, and *Journal of Parallel and Distributed Computing*. He delivered more than 150 keynote addresses, invited seminars, and media briefings and has been actively involved, in a variety of capacities, in the organization of more than 600 national and international conferences. He is a Chartered Engineer, and a Fellow of AAAS and IET (U.K.).