2025 IEEE International Conference on Big Data and Smart Computing (BigComp) | 979-8-3315-2902-4/25/$31.00 ©2025 IEEE | DOI: 10.1109/BigComp64353.2025.00054

An Approach to Enhancing Fairness in a Dynamically Growing Federated Learning Environment

Sean Vucinich
Center for Academic Innovation
University of Michigan - Ann Arbor
Ann Arbor, USA
vucinich@umich.edu

Qiang Zhu
Dept. of Computer & Information Science
University of Michigan - Dearborn
Dearborn, USA
qzhu@umich.edu

Niccolò Meneghetti
Dept. of Computer & Information Science
University of Michigan - Dearborn
Dearborn, USA
niccolom@umich.edu

***Abstract*—The demands of numerous Big Data processing applications with the data privacy requirement have led to the development of federated learning (FL), a distributed machine learning approach that offers built-in privacy protection. As with other machine learning techniques, there are increasing concerns and challenges about ensuring that decisions being made are fair and equitable to all clients/participants in FL. While many existing studies have looked at a multitude of approaches to addressing the challenges of fairness in FL, few examine fairness for dynamic FL environments where clients join or leave the system at any time as demanded by many real-world applications. This paper examines an approach to enhancing fairness in a dynamically growing FL environment, with a specific emphasis on exploring accommodation of late-joining clients. We propose two techniques, namely the Constant Weight Catch-Up Method and the Decayed Weight Catch-Up Method, to rectify the inherent unfairness in client selection for dynamic FL environments by offering additional participation opportunities for late entrants. Furthermore, we explore the fair evaluation of client contributions in dynamic FL environments through the application of the Shapley value. We present a method, called the Federated Delta-based Algorithm for Adding a Client, to efficiently compute the Shapley value for dynamically added clients in FL. Experiments demonstrate that our proposed catch-up client selection methods and the federated delta-based algorithm are quite promising in a dynamically growing FL environment.**

Keywords: Big data, federated learning, dynamic environment, fairness, client selection, Shapley value.

I. Introduction

As artificial intelligence (AI) systems become more and more complex and ubiquitous, new concerns arise over their ability to make fair and justifiable decisions, without violating the privacy of the users. These concerns are well-justified, especially when AI is used to make decisions that can have a profound impact on society [11]. Machine learning (ML) [19], one of the core branches of study within the field of artificial intelligence, aims to learn from data by creating a model of the data through the use of statistical techniques.

Federated learning [8], [9], [12], [25] is an emerging approach to large-scale machine learning; relying on the idea of distributing the training process across multiple independent clients, which all cooperate to the learning of a shared model but maintain the training data private and compartmentalized. As an inherently distributed technique, federated learning addresses critical data issues (e.g., data access, security, and privacy) while also accounting for non-independent and identically distributed (non-IID) data due to the nature of the client participation. While these features can help alleviate some of the concerns related to the rise of automated decision systems, this technique still must address issues of bias and fairness in its predictions.

There are several causes that may lead to unfairness of machine learning (ML) algorithms, both of the algorithms themselves and the models they create. Much work on ML fairness has been reported in the literature [7], [13], [16], [27]. Recently, researchers have been attracted to fairness issues in the context of federated learning (FL) [21], [23]. Due to the distributed nature of FL, the concept of fairness becomes highly relevant due to its direct impact on the client's participation in the system. Within the scope of FL, various fairness topics were studied including fair model update [14], individual fairness [26], group fairness [15], [17], [26], fair valuation [4], [22], [24], and fair client selection [6]. The main focus of this paper is to design practical mechanisms to promote the fairness of participation in dynamic FL environments, where clients can join or leave the training process at any time. As FL systems increase in scale and complexity, dynamically assimilating new participants over time, concerns regarding the fairness of the model's performance across diverse data distributions take on a heightened significance. This paper aims to scrutinize and investigate elements of fairness within such dynamic environments, exploring the intricacies of client selection (sampling) and contribution valuation through use of the Shapley value [20], a game theory concept that has been used to great success for data valuation in machine learning. We will evaluate techniques that aim to ensure fairness among all clients within a dynamic federated learning environment. Put simply, our work aims to explore fair client selection in dynamic federated learning environments, focusing on accommodating late-joining clients and evaluating client contributions using the Shapley value to ensure fairness. To our knowledge, no similar work has been reported in the literature. Existing work on fairness for FL systems is primarily focused on static environments, assuming clients in

2375-9356/25/$31.00 ©2025 IEEE
DOI 10.1109/BigComp64353.2025.00054

the FL environment do not change [4], [6], [22], [24].

The main contributions of this paper are listed as follows:

- Two new methods, called the Constant Weight Catch-Up and the Delayed Weight Catch-Up, to provide catch-up opportunities for late- joining clients in client selection for dynamic FL environments are presented.
- An efficient method, called the Federated Delta-based Algorithm for Adding a Client, to compute the Shapley value of dynamically added clients in a dynamic FL environment is introduced.
- Experiments for testing the validity and efficacy of the proposed methods are conducted.

The remainder of the paper is organized as follows. Section II presents two fairness catch-up methods for client selection in dynamic FL environments. Section III discusses an efficient method to compute the Shapley value of dynamically added clients in a dynamic FL environment. Section IV reports experimental results evaluating the proposed techniques. Lastly, Section V provides concluding remarks and future directions.

II. Client Selection Methods for Dynamically Growing FL Environments

This section delves into the client selection strategies to promote fairness and inclusivity in dynamically growing FL environments. We explore two client selection methods to balance the contributions of new clients and existing clients to the global federated model.

A. Dynamic Client Scenarios in FL

Training in federated learning is distributed and collaborative, taking place across multiple clients training on their local datasets. Local model updates from training at clients are sent to a central server for aggregation. For an FL system with multiple clients, a subset of clients is typically selected. Methods are needed to promote fairness for the client selection problem in FL. The majority of existing methods for client selection in FL assume a static client pool, i.e., the set of clients does not change over the time. However, it is often the case that clients may join or drop out of a pool dynamically. In this paper, we consider only the cases in which a client pool may dynamically grow.

There are three typical types of dynamic client pools in FL: single-client changes, non-simultaneous multi-client changes, and simultaneous multi-client changes. The first type, such as a traffic data system for several cities (clients) where a new city may join the system occasionally, involves a single client joining the client pool without overlapping with the joining time of another client. The second type, such as those seen in smart vehicle networks, involves multiple clients joining the client pool with potential overlapping between training rounds but no multiple clients joining the pool within the same training round. Lastly, the third type, such as smartphone networks for training image processing algorithms, involves multiple clients joining the pool within the same training round. The last type of dynamic clients presents the greatest challenge in balancing fairness and availability due to the highly inconsistent participation patterns. Given the interest from both major smartphone ecosystems in federated learning [8], [9], [15], [18], this type of dynamic client pools represents an area that has seen significant interest.

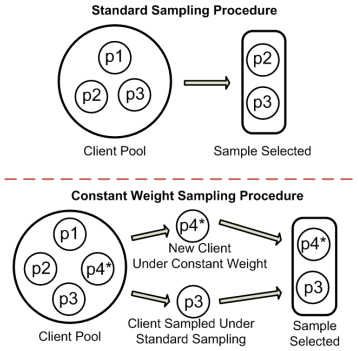

Fig. 1: Augmented Client Sampling Procedure

B. Dynamic Client Selection Procedure and Methods

While existing works within FL largely utilize static client pools to reduce the number of variables for their studies, this paper focuses on investigating scenarios where clients dynamically join after training begins. How to select dynamic clients to ensure fairness for the underlying FL system is the issue to be discussed in this subsection.

1) A Catch-up Procedure for Dynamic Client Selection in FL: In scenarios where some new clients join the system after the training has begun, such clients have missed some opportunities to contribute to the training process compared to those that have joined earlier, which may result in an unfair allocation of participation for all the clients. The catch-up mechanism is designed to ensure that these clients who join late, missing some training, are given fair opportunities to participate, similar to those who have been involved in training from the start. It augments the standard client sampling process to provide new clients a temporary increase in the opportunity to participate so they can in turn catch up with the rest of the client pool. How to balance this increase in opportunity to participate with the participation of the rest of the client pool is one of the key challenges with this approach to promoting fairness.

In existing works [6], client selection is typically handled through a simple random choice among the clients, where the proportion of the clients to participate is defined by the FL system. This proportion is often set to 50% as it provides an even split among participants and non-participants without favoring either group.

Fig. 1 illustrates simple examples of the aforementioned standard sampling procedure and the proposed augmented procedure to promote fairness. By augmenting the client sampling process and allowing the late-joining new clients to be chosen with different weights, the augmented sampling procedure may be able to positively impact fairness across the client pool.

Let I be the set of clients in an FL environment. Different clients contribute to the FL learning process at different times.

Assume that the learning process lasts for T rounds. Let S_t be the set of clients selected in round t ($1 \leq t \leq T$). In a traditional FL environment, I is static and a simple random sampling procedure is performed to select clients in S_t for each t. In a dynamic FL environment, a new set of clients I' may join the client pool starting in round t_0+1 ($1 \leq t_0 < T$). In other words, t_0 rounds of training have been performed before new clients start to join the client pool. How to select clients from the dynamic client pool $I'' = I \cup I'$ for the training at each round t, especially for $t \geq t_0 + 1$, to promote fairness is the problem to be examined in this study.

2) Constant Weight Method: A direct approach to addressing the challenge of a fair catch-up process would be applying a constant weight for each new client. That is, a new client will be chosen every round until the weight is removed. This method is formally described as follows.

Method 1: **(Constant Weight Catch-Up)**. Let I be the set of existing clients in an FL environment in which t_0 (≥ 1) rounds of training (learning) have been performed. Assume that new clients start to join the client pool at round $t_0 + 1$. Let $I' = I'_{t_0+1} \cup I'_{t_0+2} \cup ... \cup I'_T$ where $I'_{t'}$ be the set of new clients joining the client pool in round t' ($t_0+1 \leq t' \leq T$) and T denotes the number of rounds that the training (learning) lasts for the FL environment. Assume that $\beta\%$ of clients from the client pool has to be selected (sampled) and $|I| >> |I'|$.

Let $C_t(x)$ denote whether client $x \in I'' = I \cup I'$ is to be selected (sampled) in round t ($1 \leq t \leq T$), where $C_t(x) = 1$ for 'yes', while $C_t(x) = 0$ for 'no'. Let $w(x')$ denote a (constant) weight ('1' or '0') associated with new client $x' \in I'$, which can be determined as follows:

$$w(x') = \begin{cases} 1 & if\ t(x') \ < \ \lceil\ \bar{t}(x')\ \rceil \\ 0 & otherwise \end{cases} \tag{1}$$

where $t(x')$ denotes the number of rounds since client x' has joined the client pool, and $\bar{t}(x')$ denotes the average number of rounds of participation of other clients in the pool at the time when x' joined the pool. For a given round t of the client selection (sampling), whether client x is selected (1) or not (0) is determined by $C_t(x)$ as follows:

$$C_t(x) = \begin{cases} 1 & if\ (x \in I' \ \wedge \ w(x) = 1) \ \vee \ S(x, I, I') = 1 \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

where $S(x, I, I') = 1$ if and only if x is in the subset selected randomly from all subsets of $I \cup \hat{I}'$ with size $\lceil \beta * |I \cup I'| \rceil - |I' - \hat{I}'|$ where $\hat{I}' = \{x \mid x \in I' \wedge w(x) = 0\}$. Note that $I'_s = \emptyset$ for $s \geq t + 1$ in round t.

Note that the three types of client change scenarios described in Section II-A are all captured in Method 1:

- **Single-client change type**: when (1) $|I'_{t'}| \in \{0, 1\}$ for $t' \in [t_0+1, T]$ and there exists at least one $i \in [t_0+1, T]$ such that $|I'_i| = 1$, and (2) if $w(x') = 1$ for $x' \in I'_i$, then $|I'_j| = 0$ with $j - i < \lceil\ \bar{t}(x')\ \rceil$ and $t_0 + 1 \leq i < j \leq T$.
- **Non-simultaneous multi-client change type**: when $|I'_{t'}| \in \{0, 1\}$ for $t' \in [t_0 + 1, T]$ and there is at least one pair of $x' \in I'_i$ and $y' \in I'_j$ such that $w(x') = w(y') = 1$ and $j - i < \lceil\ \bar{t}(x')\ \rceil$ and $t_0 + 1 \leq i < j \leq T$.
- **Simultaneous multi-client change type**: when at least one $|I'_{t'}| > 1$ for $t' \in [t_0 + 1, T]$.

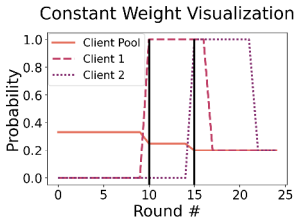

Fig. 2: Const. Weight Visual.

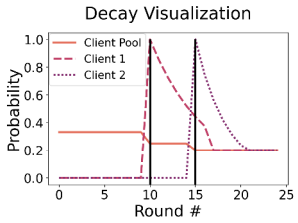

Fig. 3: Decay Visualization

The Constant Weight Catch-Up Method (i.e., Method 1) can be visualized via a simple example in Fig. 2, which demonstrates the impact the constant weight mechanism has on new clients as they join. In this example, two new clients join, at rounds 10 and 15, with the constant weight augmenting the sampling for these clients for 6 rounds. For this 6 rounds period, each new client is ensured to be selected in the sample for each round, then will revert to random sampling afterwards.

While this method can accomplish the goal of allowing a new client to participate enough to reach parity with the other clients in the pool, this method has a drawback. By ensuring that the new client is chosen for a given number of rounds, the opportunities for the remaining clients in the pool to be chosen are conversely reduced. In scenarios where multiple clients are joining the FL system in a short window, this can potentially cause very skewed client selection rounds where few or even none of the existing clients in the pool are chosen.

3) Decayed Weight Method: While Method 1 may provide some benefits comparing to the conventional approach of treating new clients with no difference from the rest clients in the pool, refining this method may yield more benefit. One improvement would utilize a decay function to initially provide a high opportunity for a new client to participate, and then gradually decrease it over time. Rather than having an all-or-nothing guarantee for the new client's participation in Method 1, the following decayed weight catch-up method provides a fine-tuned approach that can be adjusted towards whatever favors the underlying FL system has through using a decay factor γ ($0 < \gamma < 1$).

Method 2: **(Decayed Weight Catch-Up)**. Let I be the set of existing clients in an FL environment in which t_0 (≥ 1) rounds of training (learning) have been performed. Assume that new clients start to join the client pool at round $t_0 + 1$. Let $I' = I'_{t_0+1} \cup I'_{t_0+2} \cup ... \cup I'_T$ where $I'_{t'}$ be the set of new clients joining the client pool in round t' ($t_0+1 \leq t' \leq T$) and T denotes the number of rounds that the training (learning) lasts for the FL environment. Assume that $\beta\%$ of clients from the client pool has to be selected (sampled) and $|I| >> |I'|$.

Let $C_t(x)$ denote whether client $x \in I'' = I \cup I'$ is to be selected (sampled) in round t ($1 \leq t \leq T$), where $C_t(x) = 1$ for 'yes', while $C_t(x) = 0$ for 'no'. Let $w(x')$ denote a decayed weight associated with new client $x' \in I'$, which can be determined as follows:

$$w(x') = \begin{cases} \gamma^{\,t(x')} & if\ t(x') < \rho \wedge \gamma^{\,t(x')} > \varepsilon \\ 0 & otherwise \end{cases} \quad (3)$$

where $t(x')$ denotes the number of rounds since client x' has joined the client pool, γ $(0 < \gamma < 1)$ is the decay factor, ρ is the cutoff threshold, and ε $(0 < \varepsilon < \gamma)$ is a small constant. For a given round t of the client selection, whether client x is selected (1) or not (0) is determined by $C_t(x)$ as follows:

$$C_t(x) = \begin{cases} 1 & if\ [x \in I' \ \wedge\ w(x) \neq 0 \ \wedge\ S'(x, w(x)) = 1] \\ & \quad \vee\ S(x, I, I') = 1 \\ 0 & \text{otherwise} \end{cases} \quad (4)$$

where $S'(x, w(x)) = 1$ if and only if x is selected (sampled) according to the probability of $w(x)$; and $S(x, I, I') = 1$ if and only if x is in the subset selected randomly from all subsets of $I \cup \hat{I}'$ with size $\lceil \beta * |I \cup I'| \rceil - |SI'|$ where $SI' = \{x \mid x\ is\ sampled\ from\ Eq.\ (3)\}$ and $\hat{I}' = I' - SI'$. Note that $I'_s = \emptyset$ for $s \geq t + 1$ in round t.

The main difference between Methods 1 and 2 is that, the former guarantees that each client is selected under a constant (=1) weight (probability) to the training client set in each round until the average number of rounds of participation for prior clients is reached, while the latter applies a gradually decayed weight (probability) to decide if a new client is selected to the training client set in each round until a cutoff threshold for the number of rounds is reached or the decayed weight becomes too small. In either case, a new client is reverted back to the existing client pool for selection through the simple random sampling scheme when its weight becomes zero (removed). With the gradual decay of weight and ability to be more fine-tuned, Method 2 can potentially be more broadly applicable to scenarios than Method 1. In fact, Method 2 is degraded to Method 1 when $\gamma = 1$ and $\rho = \lceil\ \bar{t}(x')\ \rceil$.

Note that the decay function used in Method 2 can be replaced by any decay function $f(t)$ for $t \geq 1$ such as γ^t $(0 < \gamma < 1)$, t^μ $(\mu < 0)$, and $log_\alpha t$ $(0 < \alpha < 1)$. This function is intended to model a simple decay to better showcase the impact of the method on client sampling. In this study, we consider the decay function $f(t) = \gamma^t$ only.

As the decay rate γ is able to influence the effect the approach has in the augmented sampling process, the decayed weight approach is able to be better tailored to the specific fairness needs of a given FL environment. Compared to the constant weight approach, the decayed weight can also be varied depending on an individual client as well as can be dynamically changed as more clients join to positively influence the balance of fairness among clients.

Fig. 3 provides the visualization of an example of how Method 2 works. In this example, two new clients join at rounds 10 and 15, respectively, with the decay function stopping after 6 rounds after the client joined. The decay rate, γ, is set to 0.85 for new Client 1, and is set to 0.75 for new Client 2. For this 6 rounds period, each new client is selected (sampled) according to the delayed weight in each round, then is reverted to the simple random sampling with the remaining pool after this period is exhausted.

4) Cutoff for New Clients: As the dynamic catch-up mechanisms are intended to act for a limited time, one has to determine when a given new client should no longer be treated as such and revert back to the default client selection scheme. This scheme needs to balance two key factors, new clients should be afforded the opportunity to participate according to the amount of training rounds the client may have missed, and the opportunity under the catch-up mechanism should be brief enough so that the rest of the client pool is not unfairly biased against. Having an appropriate cutoff (threshold) ρ is required as without one, new clients can become over-favored with the client selection becoming biased towards them.

There are two reasonable ways of determining the cutoff, i.e., using a static cutoff or using a scaling (dynamic) cutoff. The static cutoff is the naive approach, namely, using a constant number of rounds tied to the amount of overall training rounds, ensuring that each new client would have the same number of rounds under the catch-up mechanism. A scaling cutoff, by contrast, uses the number of training rounds that a new client has missed as the cutoff. This ensures that a given new client has the catch-up mechanism last only long enough to allow it the opportunity to catch up about the number of rounds it missed. Note that the client selection process ends when the total number of training rounds (T) for the underlying FL environment has been reached.

III. Fairness Evaluation via Shapley Value in Dynamic FL Environments

Across the challenges raised by the quest for fairness and interpretability in FL, our study focuses on a specific problem of great practical value: the problem of rewarding each client for its contribution to the global model in a way that is fair and unbiased. This problem has been formalized in [24]. A natural approach to designing a fair rewarding scheme is to apply existing results from the discipline of cooperative Game Theory [2], leveraging the seminal work by Lloyd Shapley [3], [20]. This section explores the application of this concept, aptly named the Shapley value, as a foundational framework for addressing the challenges of fairness in federated learning. In particular, we investigate dynamic Shapley methodologies, with a specific focus on a delta-based addition strategy, for efficient model updates and client valuation in dynamic FL environments.

A. The Shapley Value

There are many methods and strategies for determining the valuation of data and their contributions made to a given model in machine learning. The Shapley value has had an out-sized influence, mainly due to its usage in feature evaluation.

In the context of cooperative games, the Shapley value [20] is a measure to assess the expected marginal contribution that a player provides to a coalition by joining it. This metric carries many desirable properties, of which there are four key assumptions/requirements of fairness inherent to the Shapley value (Symmetry, Fairness and Efficieny, Additivity, and Uniqueness), as shown by Dubey in [3]. These assumptions, while not fully applicable to derivative implementations of the Shapley value in other domains, they remain important

in framing how future implementations may compromise on aspects of fairness.

The computation of an exact Shapley value is a #P-Complete problem [1], owing to the complexity growth inherent to calculating the marginal contributions of every subset. The computation of the exact Shapley value, as defined in the classical method, can be found in [20].

B. The Shapley Value within FL

To date, literature investigating the usage of the Shapley value for the valuation of client participation within federated learning has been limited. Given the inherent need to retrain a model with different subsets of participants, a strict implementation of the Shapley value is not viable in an FL environment. With relaxed fairness constraints, the need for retraining may be avoided, providing potential ways to avoid excessive training. An example of work that attempts to avoid retraining can be found in [22], which extends the data Shapley value [5] into federated learning.

The Federated Shapley Value (FedSV) [24] further reduces the need for retraining by computing the Shapley values for clients in each round of training and reporting the summation over all training rounds as the final result. The goal of this approach is to reduce the communication costs and time required for model retraining, while retaining the majority of fairness properties of the Shapley value, making it practical for use in large-scale FL.

Let $v(\cdot)$ denote the utility function that links an individual client to a performance valuation on the global model. This utility function takes the ordered set of selected clients as input. Let $I = \{1, ..., N\}$ denote the set of clients participating in the FL system, where these clients have participated at least once for T rounds of model training. Let I_t represent the set of clients selected in training round t ($I_t \subseteq I$). Let $I_{1:t-1}$ be shorthand for $I_1 + ... + I_{t-1}$ where $t \geq 2$ and $\emptyset$ for $t = 1$. The federated Shapley value s of client i in round t can be defined as:

$$s_t^v(i) = \frac{1}{|I_t|} \sum_{S \subseteq I_t \setminus \{i\}} \frac{1}{\binom{|I_t|-1}{|S|}} [v(I_{1:t-1} + (S \cup \{i\})) - v(I_{1:t-1} + S)] \quad \text{if } i \in I_t \tag{5}$$

and $s_t^v(i) = 0$ otherwise, representing non-selected clients are assigned a federated Shapley value of 0. The FedSV then sums all the individual Shapley values for each round in the following manner:

$$s^v(i) = \sum_{t=1}^{T} s_t^v(i) \tag{6}$$

To implement a data valuation approach that utilizes the Shapley value within an FL environment, some of the fairness constraints inherent to the Shapley value were relaxed. Specifically, the attribute of symmetry inherent in the classic Shapley value is not present in the Federated version, as the order of participants within an FL system impacts their contributions to the global model. As order is, therefore, a fundamental requirement of FL, the classic Shapley value would conflict on that property of fairness. The following are the unique properties of the Federated Shapley Value:

- Group Rationality: For all clients, the value of the global model is distributed completely: $v(I) = \sum_{i \in I} s^v(i)$
- Fairness: (1) Two clients that are identical in respect to their contributions should therefore have the same value assigned. That is, clients i and j are equivalent if $v(S \cup \{i\}) = v(S \cup \{j\}), \forall S \subseteq I \setminus \{i, j\}$. Therefore, $s^v(i) = s^v(j)$. (2) Clients with zero contributions to all subsets receive zero value. That is, if $v(S \cup \{i\}) = 0, \forall S \subseteq I \setminus \{i\}$, then $s^v(i) = 0$.
- Additivity: The values of each of a client's utilities sum up to the sum of all utilities: $s^v(i_1) + s^v(i_N) = s^v(i_1 + i_N)$ for $i \in I$.

C. Dynamic Shapley

The aforementioned Shapley value based methods all are targeted at static environments. These static environments are related to non-changing data or features within ML, or non-changing client pools within FL systems.

1) Dynamic Shapley within ML: Work in investigating the use of the Shapley value in a dynamic ML environment is just emerging, while no such work for a dynamic FL environment has been reported. Zhang et al. [27] studied the topic of Dynamic Shapley Value Computation for data valuation purposes in ML when data points are added or removed from a dataset. The authors propose multiple methods including a pivot-based approach and a delta-based algorithm for addressing the problem with dynamically adding and removing data points. The main goal of these methods is to achieve more efficiency and effectiveness when approximating the Shapley value, through the reduction of the computational time, computational amount, or memory needed.

Their proposed delta-based algorithm is the most pertinent to our study, aiming to reduce the sample size needed to achieve convergence during the approximation of the Shapley value. This approach accomplishes this by only computing the relative changes or delta between the Shapley values, rather than computing the complete Shapley values. By starting with pre-computed Shapley values from the original data points and computing the differential contributions relative to the new data point, the delta algorithm is able to achieve a smaller overall sample within similar accuracy to the other proposed methods. The original delta based algorithm for adding a data point in ML can be found in [27].

2) Delta-based Addition for Dynamic FL Environments: Adapting the delta-based algorithm for adding singular data points in ML described above to function within a dynamic FL environment, we can evaluate the potential efficacy of utilizing a dynamic Shapley value approach in FL. As the delta-based algorithm in [27] is based upon the original definition of the Shapley value, which is incompatible with FL. Similar to how the Federated Shapley Value required changes to the original definition to allow approximation of the Shapley value, we make necessary changes to this delta-based algorithm in order for it to function using the Federated Shapley Value.

As utilities are calculated during the course of training for the federated Shapley value, the proposed federated delta-

based algorithm for adding a client in a dynamic FL environment is able to utilize the computed utilities and the approximated federated Shapley values to compute the changes in contributions resulting from the addition of a new client, $\Delta\mathcal{SV}$. The original delta addition algorithm can therefore be simplified as permutations are no longer necessary due to the removal of the training requirement of the algorithm.

Algorithm 1 shows the details of our proposed Federated Delta-based Algorithm for Adding a Client. The computation of the updated federated Shapley values $\mathcal{SV}_i^+$ begins with the federated Shapley value computed prior to the new client, x_{n+1} joining. Lines 3-5 take each client, from first to last, and calculates the differential contribution, $\Delta\mathcal{SV}$ for each. Line 6 approximates the new Shapley value of the new client. This process is repeated over each client, resulting in the approximation of the difference of the federated Shapley value through averaging the differential contributions, $\Delta\mathcal{SV}$. Lines 7-8 then combine the original federated Shapley value and the $\Delta\mathcal{SV}$ for each client to approximate the new federated Shapley values of the original clients.

Algorithm 1 Federated Delta-based Algorithm for Adding a Client

Input: Clients $x_1, ..., x_n, x_{n+1}$; $\mathcal{SV}_i$ for x_i and utilities $\mathcal{U}(\{x_1, ..., x_i\})$ $(1 \leq i \leq n)$
Output: Shapley value $\mathcal{SV}_i^+$ for each client $x_i (1 \leq i \leq n+1)$
1: $\Delta\mathcal{SV}_i \leftarrow 0$ $(1 \leq i \leq n)$;
2: $\mathcal{SV}_i^+ \leftarrow 0$ $(1 \leq i \leq n+1)$;
3: **for** $i = 1$ to n **do**
4: $\quad \Delta\mathcal{SV}(x_i) = [\mathcal{U}(\{x_1, ..., x_i, x_{n+1}\}) - \mathcal{U}(\{x_1, ..., x_i\})] - [\mathcal{U}(\{x_1, ..., x_{i-1}, x_{n+1}\}) - \mathcal{U}(\{x_1, ..., x_{i-1}\})]$;
5: $\quad \Delta\mathcal{SV}_i += \Delta\mathcal{SV}(x_i)/(n+1) \cdot i$;
6: $\quad \mathcal{SV}_{n+1}^+ += [\mathcal{U}(\{x_1, ..., x_i, x_{n+1}\}) - \mathcal{U}(\{x_1, ..., x_i\})]/(n+1)$;
7: **for** $k = 1$ to n **do**
8: $\quad \mathcal{SV}_k^+ = \mathcal{SV}_k + \Delta\mathcal{SV}_k$
return $\mathcal{SV}_1^+, ..., \mathcal{SV}_{n+1}^+$;

IV. Empirical Evaluation and Analysis

Experiments were conducted to evaluate our proposed methods. Representative results are reported in this section due to space limitation. To evaluate the efficacy of the catch-up strategies, we utilized a baseline method with no change to address new clients joining the client pools late. Note that this study examines fairness exhibited by the proposed methods only within the context of a horizontal FL system [25].

A. Experiment Setting

A workstation having an Intel i7-8700k with a single NVIDIA RTX 2080 GPU and 64GB of system memory was used for running the experiments. The FL environment was configured using Python 3.11 and PyTorch 2.1.1 to facilitate GPU acceleration. The FL experiments utilized a modified environment initially based on the work from authors in [4]. All experiments used set random seeds for reproducibility, starting at a seed of 0, up through an equal number of iterations for the given experiment.

The main dataset used in the experiments is the commonly used real-world benchmark, the MNIST dataset [10]. Our experiments considered the non-IID (independent and identically distributed) situation. Each client is randomly distributed data consisting of a sample of just two classes of the MNIST data, following the experimental design as laid out in the CompFedSV study [4] and the original FedAvg FL study [24].

1) Experiment Design: There are two main representative experiments presented in this study. The first evaluates the client selection (sampling) techniques within the context of a dynamic FL environment. The second experiment investigates the performance of the Federated Delta-based Addition Algorithm (Algorithm 1), examining the efficacy of the dynamic Shapley based method to better analyze the effects the method has on client contribution valuation.

In the first experiment, the following catch-up methods are compared:

- Baseline – Each new client is simply added to the client pool, which is selected via the simple random sampling.
- Constant Weight – The constant weight catch-up method as discussed in Section II-B2.
- Decay (Slow) – The decayed weight catch-up method described in Section II-B3, utilizing a decay rate γ of 0.975.
- Decay (Fast) – The decayed weight catch-up method, utilizing a decay rate γ of 0.875.

The different decay rates aim to demonstrate the impact from changing the decay rate on client participation. The cutoff for the implementation of each catch-up method is the average number of rounds that clients have participated in prior to the new client joining. Each experiment utilizes a participation fraction β of 50%.

In the second experiment, the Federated Delta-based Addition Algorithm (Delta Add) is compared with the baseline method (Baseline).

In experiments, the Federated Shapley Value is used as the client contribution metric. To measure the participation fairness for a new client in a method, we use the average distance (called the average Shapley value distance) between the Shapley value of this client and the Shapley values of other selected clients in trainings. A smaller average Shapley value distance indicates that the contribution (Shapley value) of this new client is closer to the contributions (Shapley values) of other selected clients, demonstrating a better participation fairness is achieved.

To measure the participation fairness among clients in a method, we also use the difference (called the maximum Shapley value distance) between the maximum Shapley value and the minimum Shapley value among all selected clients in each training. Several observations about the maximum Shapley value distances are made and analyzed. The observed lower and upper bounds of the maximum Shapley value distances would show the best and worst fairness scenarios, respectively. If a method can make the lower bound of the maximum Shapley value distance smaller, the difference among contributions of the clients is smaller in the best case. If a method can make the upper bound of the maximum Shapley value distance smaller, the difference among contributions of the clients is smaller in the worst case. In general, we want a method to have more cases with the maximum Shapley value distances

closer to the lower bound ("best" cases in the bin/range near the lower bound) and less cases with the maximum Shapley value distances closer to the upper bound ("worst" cases in the bin/range near the upper bound). Therefore, when we evaluate the performance of participation fairness achieved by the methods under consideration, we compare their lower and upper bounds, as well as the number of best cases and the number of worst cases.

B. Experimental Results

1) Experiment 1 - Client Selection Sampling in FL: This first experiment examines how the augmented client selection/sampling methods (i.e., the catch-up mechanisms) can impact fair participation in a dynamic FL environment. For all selection methods, 100 iterations of the FL training scenario were run, with each iteration consisting of 25 training rounds. The client pool started with 10 initial clients, with 2 new clients joining throughout training, with the new clients joining at rounds 10 and 15, respectively.

TABLE I: Client Avg. Shapley Value Distances in Experiment 1

Observations on Average Shapley Value Distances		
Method	New Client1	New Client2
Baseline	0.0353	0.0824
Constant Weight	0.0224	0.0809
Decay - Slow	0.0197	0.0798
Decay - Fast	0.0276	0.0812

Table I shows the average Shapley value distances. From the table, we can see that each client selection method compared to the baseline method results in a smaller average distance. This smaller distance represents that the measured contributions of each new client are closer to the other clients in the client pool, indicating that each of the augmented selection methods are positively impacting participation fairness. Looking closer at the specific methods, we can see that with respect to both new clients, the slow decay method resulted in the smallest distance gap, with the constant weight and fast decay methods following closely behind.

Table II shows the maximum Shapley value distances. First, we can see that each augmented selection method resulted in smaller lower bound distances, i.e., improving fairness in the best case. Next, both the Constant Weight and the Fast Decay resulted in smaller upper bound distances, i.e., improving fairness in the worst case. However, the Slow Decay did not outperform the baseline in the worst case since it yielded a larger upper bound distance. Each selection method outperformed or tied with the baseline method with respect to both the bin $S_{smallest}$ of smallest distances (≤ 0.15) and the bin $S_{largest}$ of largest distances (≥ 0.30). In fact, for all the augmented selection methods, the number of best cases

TABLE II: Client Max. Shapley Value Distances in Experiment 1

Observations on Maximum Shapley Value Distances					
Method	Lower Bound Dist.	Upper Bound Dist.	# $\leq$ 0.15	# in between	# $\geq$ 0.30
Baseline	0.1039	0.3976	7	81	12
Constant Weight	0.0979	0.3910	10	79	11
Decay - Slow	0.0977	0.4293	13	77	10
Decay - Fast	0.0989	0.3492	7	84	9

TABLE III: Client Avg. Shapley Value Distances in Experiment 2

Observations on Average Shapley Value Distances						
Method	Shapley Val (avg)	New Client1	New Client2	New Client3	New Client4	New Client5
Baseline	0.2960	0.0166	0.0629	0.0824	0.0697	0.1361
Delta Add	0.2934	0.0033	0.0598	0.0782	0.0688	0.1338

in $S_{smallest}$ is increased or tied, and the number of worst cases in $S_{largest}$ is reduced. Specifically, the Slow Decay demonstrated improvements in both increasing the cases of smaller distances and reducing the cases of larger distances. The Constant Weight demonstrated similar improvements but on a smaller scale. The Fast Decay demonstrated the best improvement in reducing the cases of larger distances, but is tied with the baseline in increasing the cases of the smaller distances.

The above observations show that the goal of allowing new clients an opportunity to catch up in contributions for an overall more fair distribution has been achieved, which validates the idea that the augmented selection methods are able to impact client selection and participation fairness in a positive manner, compared to the baseline.

2) Experiment 2 - Federated Delta-based Addition: In this second experiment, we examine the impact that the proposed Federated Delta-based Addition algorithm (Algorithm 1) has on participation fairness in a simultaneous multi-client join situation. 200 iterations of the FL training scenario were run, with each iteration consisting of 25 training rounds. The client pool started with 10 initial clients, with 2 new clients joining simultaneously at round 10, and 3 new clients at round 15. By having the clients join at the same time rather than spread out across multiple rounds, we can evaluate the efficacy of the Federated Delta-based Addition algorithm in the most complex of the dynamic client selection scenarios.

Table III shows the average Shapley value distances in this experiment. From the table, we can see that with respect to every new client and the overall client pool, the Federated Delta-based Addition algorithm is able to reduce the average distance in Shapley values, demonstrating that the algorithm is able to positively impact client participation fairness.

Furthermore, there are some interesting findings that can observed from examining the new clients specifically. First, despite new clients 1 and 2 joining at the same time, they show different levels of improvement in the average Federated Shapley value distances, with client 1 showing a much better improvement. A similar trend is visible with new clients 3 through 5. Specifically, clients 4 and 5 have similar average distances, while client 3 is significantly different. A possible explanation for this phenomenon is that one of the clients in the multi-client join scenario is inadvertently biased towards in the delta-addition algorithm or during client selection.

TABLE IV: Client Max. Shapley Value Distances in Experiment 2

Observations on Maximum Shapley Value Distances					
Method	Lower Bound Dist.	Upper Bound Dist.	# $\leq$ 0.20	# in between	# $\geq$ 0.45
Baseline	0.14	0.70	18	138	44
Delta Add	0.15	0.70	26	137	37

Table IV shows the maximum Shapley value distances in the second experiment. Compared to the baseline, the Federated Delta-based Addition had the same upper bound (maximum) distance but a slightly larger lower bound distance, indicating no improvement in this measure. On the other hand, the Federated Delta-based Addition demonstrated significant improvements in both increasing the best cases (with distances ≤ 0.20) and reducing the worst cases (with distances ≥ 0.45), resulting in better participation fairness.

From the above observations using both the average federated Shapley value distances and the maximum Shapley value distances, we can see that the Federated Delta-based Addition Algorithm shows a significantly positive impact on fairness with respect to client participation as compared to that of the baseline in a dynamic FL environment.

V. Conclusions and Future Work

There is an increasing demand for dynamic federated learning in which clients join and/or leave the system dynamically. This paper examines the problem of ensuring fairness for dynamically growing federated learning environments. As most existing studies explore the challenges associated with ensuring fairness for static FL environments for a fixed pool of clients, this problem represents a new area of intrigue that is more applicable to real-world FL environments.

In this paper, we explored client selection within dynamically growing FL environments, with a particular focus on developing fair strategies for accommodating late-joining clients. We have presented two novel methods, namely, the Constant Weight Catch-Up and the Decayed Weight Catch-Up, which aim to provide catch-up opportunities for these late-joining clients in dynamic FL environments. We have also studied the topic of the evaluation of fairness in a dynamic FL environment utilizing the Shapley value, a game-theory based method that has seen great success within machine learning. We have investigated the new concept of a dynamic Shapley value based fairness valuation, which adapts an existing Delta-based method for computing Shapley value of dynamically added data points in ML to a dynamic FL environment by applying the concept of the Federated Shapley value for FL, resulting a new method, called the Federated Delta-based Algorithm for Adding a Client. This proposed method provides a promising method for computing the Federated Shapley value of dynamically added clients in FL.

While this study investigated dynamically growing FL environments, dynamically shrinking environments are an interesting space of future work. The topic of general fairness in dynamic FL environments is another area of future work. Ensuring fair representation of clients in highly dynamic FL systems is exceptionally difficult as the state of the client pool is rapidly changing. More sophisticated client selection algorithms that consider factors such as client heterogeneity, data distribution, or granular network conditions could help mitigate areas of unfairness. Lastly, further work on the Shapley value in FL may result in establishing standardized benchmarks and evaluation metrics for assessing the performance of approaches within FL from the perspective of fairness, facilitating comparisons across different approaches by quantifying their impact and promoting reproducible research.

References

[1] H. Aziz and B. de Keijzer. Shapley meets shapley. *arXiv preprint arXiv:1307.0332*, 2013.

[2] R. Branzei, D. Dimitrov, and S. Tijs. *Models in cooperative game theory*, volume 556. Springer Science & Business Media, 2008.

[3] P. Dubey. On the uniqueness of the shapley value. *Int'l J. of Game Theory*, 4:131–139, 1975.

[4] Z. Fan, H. Fang, et al. Improving fairness for data valuation in horizontal federated learning. In *2022 IEEE 38th Int'l Conf. on Data Eng. (ICDE)*, pages 2440–2453. IEEE, 2022.

[5] A. Ghorbani and J. Zou. Data shapley: Equitable valuation of data for machine learning. In *Int'l Conf. on ML*, pages 2242–2251. PMLR, 2019.

[6] T. Huang, W. Lin, et al. An efficiency-boosting client selection scheme for federated learning with fairness guarantee. *IEEE Trans. on Parall. and Distr. Syst.*, 32(7):1552–1564, 2021.

[7] J. Kleinberg, S. Mullainathan, and M. Raghavan. Inherent trade-offs in the fair determination of risk scores. *arXiv preprint arXiv:1609.05807*, 2016.

[8] J. Konečnỳ, H. B. McMahan, et al. Federated optimization: Distributed machine learning for on-device intelligence. *arXiv preprint arXiv:1610.02527*, 2016.

[9] J. Konečnỳ, H. B. McMahan, F. X. Yu, et al. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016.

[10] Y. LeCun and C. Cortes. MNIST handwritten digit database. 2010.

[11] S. Makridakis. The forthcoming artificial intelligence (ai) revolution: Its impact on society and firms. *Futures*, 90:46–60, 2017.

[12] B. McMahan, E. Moore, et al. Communication-efficient learning of deep networks from decentralized data. In *Artifi. Intelli. and Stat.*, pages 1273–1282. PMLR, 2017.

[13] N. Mehrabi, F. Morstatter, et al. A survey on bias and fairness in machine learning. *ACM Comput. Surv.*, 54(6):115:1–115:35, 2022.

[14] Z. Pan, S. Wang, et al. Fedmdfg: Federated learning with multi-gradient descent and fair guidance. In *Proc. of the AAAI Conf. on Artifi. Intelli.*, volume 37, pages 9364–9371, 2023.

[15] A. Papadaki, N. Martinez, et al. Minimax demographic group fairness in federated learning. In *2022 ACM Conf. on Fairness, Accountability, and Transparency*, pages 142–159, 2022.

[16] D. Pessach and E. Shmueli. A review on fairness in machine learning. *ACM Comput. Surv.*, 55(3):1–44, 2022.

[17] B. Rodríguez-Gálvez, F. Granqvist, et al. Enforcing fairness in private federated learning via the modified method of differential multipliers. *arXiv preprint arXiv:2109.08604*, 2021.

[18] B. Rodríguez-Gálvez, F. Granqvist, et al. Enforcing fairness in private federated learning via the modified method of differential multipliers. In *NeurIPS Workshop*, 2021.

[19] A. L. Samuel. Some studies in machine learning using the game of checkers. *IBM J. of Res. and Devel.*, 3(3):210–229, 1959.

[20] L. S. Shapley et al. A value for n-person games. 1953.

[21] Y. Shi, H. Yu, and C. Leung. Towards fairness-aware federated learning. *IEEE Trans. on Neural Net. and Learn. Syst.*, 35(9):11922–11938, 2024.

[22] T. Song, Y. Tong, and S. Wei. Profit allocation for federated learning. In *2019 IEEE Int'l Conf. on Big Data*, pages 2577–2586. IEEE, 2019.

[23] S. Vucinich and Q. Zhu. The current state and challenges of fairness in federated learning. *IEEE Access*, 2023.

[24] T. Wang, J. Rausch, et al. A principled approach to data valuation for federated learning. *Federated Learning: Privacy and Incentive*, pages 153–167, 2020.

[25] Q. Yang, Y. Liu, et al. Federated machine learning: Concept and applications. *ACM Trans. on Intelli. Syst. and Tech.*, 10(2):1–19, 2019.

[26] X. Yue, M. Nouiehed, and R. Al Kontar. Gifair-fl: A framework for group and individual fairness in federated learning. *INFORMS J. on Data Sci.*, 2(1):10–23, 2023.

[27] J. Zhang, H. Xia, et al. Dynamic shapley value computation. In *2023 IEEE 39th Int'l Conf. on Data Eng. (ICDE)*, pages 639–652. IEEE, 2023.