FAIRNESS OF FEDERATED LEARNING WITH DYNAMIC PARTICIPANTS

Anonymous authors
Paper under double-blind review

ABSTRACT

The concept of fairness has been widely caught attention in Federated Learning (FL). While there are tremendous studies about various notations of fairness in FL in recent years, all of them only consider the case where the training process starts and ends at the time point for all participants. Actually, participants could be dynamic and they may join and leave the training process at different time points. However, participants who join the training process at different time points receive similar incentive benefits can be seen as a signal of unfairness. In this paper, we provide the first study on such fairness of FL for dynamic participants. First, we propose a new mathematical definition of the above fairness namely *dynamic fairness*. Briefly speaking, an algorithm is dynamically fair and satisfies that local agents who participate in the model training longer should receive more benefits than those who participate in the process shorter. Second, we develop a simple but novel method, which could be seen as a normalized version of *Fedavg*, and theoretically show that it is fairer than *Fedavg*. Moreover, we can combine our method with the previous methods in fair FL for static participants to additionally guarantee fair treatment for local agents who join the training process at the same time point by minimizing the discrepancy of benefits they receive. Finally, empirically we propose a measure for *dynamic fairness* and demonstrate that our method can achieve a fairer performance under our definition of fairness through intensive experiments on three benchmark datasets.

1 INTRODUCTION

As one of the most fundamental learning frameworks for preserving the privacy of distributed data, Federated Learning (FL) (Konečnỳ et al., 2016) has prospered in the machine learning community in the last few years. In the canonical FL setting, there are several local agents, and each of them holds a dataset for local training. And there is a controller (server) which aggregates gradient vectors or local models from agents for global model updates. During the training process, the agents only communicate their gradients or local models to the server and the original data never leaves the local agents. Therefore, FL can protect the data information of each agent from leaking. To comply with the privacy regulations such as the General Data Protection Regulation (GDPR) (gdpr), variants of FL frameworks have been widely studied, and recently adopted in industry, such as Apple's "FE&T" (Paulik et al., 2021), Google's Gboard (gboard), and Alibaba's FederatedScope (Xie et al., 2022).

While the recent advances in FL present a promising framework to learn from distributed data privately and efficiently, most of the current research mainly focuses on the central server's benefits, i.e., developing methods to improve the convergence rate or the generalization performance in the FL setting, while ignores local agents' interests. However, such attention to the server's benefits may cause fairness issues which make local agents less interested in participating in the model training. For instance, those methods usually apply thresholds such as bandwidth and transmission speed to selectively choose clients (Shi et al., 2021), which potentially leads to unfair client selection in the FL system. Local devices with low transmission speed might be neglected frequently during the training process, and eventually become never-represented or under-represented client groups. Moreover, some researchers have noticed that the participants sometimes suffer from unfair incentive rewards (Zhan et al., 2021). Kairouz et al. (2021) notice the free-rider problem in the FL system. In the free-rider scenario, clients who contribute less (e.g., better data quality vs. worse data quality)

in training the model receive the same resulting model as those who contribute more to the training. Distributing models with performance incommensurate to each participant's contribution might discourage active clients from continuously collaborating in the model training.

To leverage the unfairness issue, there are tremendous work studies on Fair FL by considering various definitions of fairness recently, such as selection fairness (Zhou et al., 2021) and collaboration (Lyu et al., 2020) (see Related Work section for more details). However, all of these work only considers the case where all the participants are static, i.e., they join the training process at the same time point, while in practice such assumptions may not always hold as the participants may be dynamic, i.e., different agents could join or leave the training at different time points. In such a dynamic scenario, there are additional fairness issues compared with static ones. Consider the following case as an example, suppose the agents could join at different time points and they will never leave before the training process ends. In this case, the agents who join the training earlier (contributed more) will expect higher benefits than the ones who join later. Thus, participants who join the training process at different time points receive similar incentive benefits can be seen as a signal of inequality. However, to our best knowledge, there are no previous work studies on such fairness in FL.

In this paper, we provide the first study to alleviate the above fairness issue caused by dynamic participants by providing some new definitions, methods, and measures. Specifically, our contributions can be summarized as follows:

1. First, we provide a rigorous definition for the above fairness, namely *dynamic fairness*. Briefly speaking, we call an algorithm dynamically fair if its performance is commensurate to the length of each client's participating time. Equivalently, it satisfies that the agents with longer participation time receive more benefits, which could be seen as between-group fairness. Besides that, we also provide criteria to compare the dynamic fairness of two algorithms.
2. Next, we propose several dynamically fair methods. First, we propose a simple but efficient method namely *Normalized Fedavg*. Generally speaking, our method could be thought of as a normalized version of *Fedavg* where we use the normalized SGD instead of SGD for local training. Interestingly, we theoretically show that our algorithm is fairer than the vanilla *Fedavg* (McMahan et al., 2017). To further improve the convergence rate practically, we propose a method namely *Modified Normalized Fedavg*.
3. Moreover, due to the simplicity of the idea, our method is compatible with other fair FL methods. Specifically, we combine our method with the previous methods in fair FL for static participants to additionally guarantee fair treatment for local agents who join the training process at the same time point by minimizing the discrepancy of benefits they receive, i.e, we can achieve *within-group fairness* additionally.
4. Finally, we propose new measures for dynamic fairness and provide empirical studies of our methods. With extensive experiments on three datasets *MNIST* (LeCun et al., 1998), *Fashion MNIST* (Xiao et al., 2017), and *CIFAR10* (Krizhevsky et al., 2009), we find that our methods are not only dynamically fair, but also achieve better fairness compared with *Fedavg*.

Due to the space limit, all the proofs, some additional sections, algorithms, and experiments of our methods are included in Appendix.

2 Related Work

Existing studies have proposed several definitions of fairness in federated learning. Zhou et al. (2021) proposed the concept of selection fairness: a fair FL model should provide more participation opportunities for never-represented or under-represented client groups. The following literature tries to promote a fair client selection by introducing the sampling constraints to the FL model (Huang et al., 2020).

Different from selection fairness, Li et al. (2019) mentioned that one essential notion of fairness is to accomplish a relatively uniform accuracy distribution across devices, which is defined as the standard accuracy parity (Zafar et al., 2017). In a previous study, Li et al. (2021) suggested reducing

the variation of model performance on different clients' datasets can be seen as a reliable indicator for standard accuracy parity. However, the researchers ignored the importance of clients' contributions in training an FL model. For instance, as Lyu et al. (2020) proposed in their literature, a client who contributes more to the federated system deserves a better performing local model than those who contributed less, which is defined as collaboration fairness. Lyu et al. (2020) proposed that the quality of each client's uploaded gradients is sufficient to determine participants' contribution.

One critique of the above scenario is that the concept of time is ignored. When training an FL model, all the clients need to incur some cost to participate in the training. For instance, if a company wants to build a profitable FL model, they have to invest not only money and data but also plenty of time since training and commercialization of the FL models take time. Yu et al. (2020) introduced the idea of regret, which refers to the difference between the incentive rewards clients have received and what they should receive while taking how long they have waited to receive the payoff into account.

However, each participant's training time was ignored in all the above scenarios. In this paper, we proposed that in the long term, clients who join an FL model training longer should be rewarded with better model performance than those who participate in the training shorter since they contribute more time to the model training.

3 Dynamic Fairness For Federated Learning

In this section, we will formally define the fairness discussed in the Introduction. Before that, we provide an overview of the standard Federated Learning (FL) setting.

In FL, there are m agents where the i-th agent has a local dataset $D_i = \{x_{i,j}\}_{j=1}^{n_i}$ (the data samples could be either i.i.d. or non-i.i.d. sampled) and a central server. We also have a loss function ℓ and the central server aims to solve the following minimization problem:

$$\min_{w \in \mathbb{R}^d} F(w) = \sum_{i=1}^{m} p_i F_i(w), \tag{1}$$

where $F_i(w) = \frac{1}{|D_i|} \sum_{x \in D_i} \ell(w; x)$ is the empirical risk function for the i-th agent on his/her dataset D_i and p_i is the weight for the k-th agent, for example $p_i = \frac{n_i}{\sum n_i}$.

Dynamic Federated Learning Setting: While most of the previous work focus on the case where all agents are static, i.e., all of them join in the training process at the same time point (for simplicity, in this paper we assume one-time step responds to one update of the global model). Here we consider a dynamic setting of FL. For simplicity, we consider a dynamic setting with a finite number of time points. That is there are S time points $t_1, \cdots, t_S$, and for each time point there is a set of agents v_i who will join the training process (for simplicity here we denote t_1 as the time when the training process starts). Since the server cannot get the information for all participants, now its goal is to minimize $\sum_{i \leq M} \boldsymbol{L}_{v_i}(w)$ at time point t_M with $1 \leq M \leq S$, where $\boldsymbol{L}_{v_i}(w) = \sum_{j \in v_i} p_j^M F_j(w)$, where p_j^M is the weight for the j-th agent at time point M, i.e., the objective function is the weighted sum of empirical risk functions of all the agents who join at time point t_i. That is, it wants to minimize the empirical risk for all the agents who join at or before the time point t_M. [1] It is notable that when $M = S$, then the objective function is equivalent to the original one in (1).

As we mentioned earlier, in the above dynamic FL setting there could be additional fairness concerns. For example, we consider two succeed time points t_1 and t_2 with the associated participant sets v_1 and v_2 (we assume $t_2 > t_1$), and the server conducts *Fedavg* to train the model. At time point t_2, from the perspective of agents in v_1 the algorithm itself may be unfair to them as the agents in v_2 can directly use the current model (which has already been trained for several rounds by using the data in v_1) without any cost. We can see the above unfairness is ubiquitous in the dynamic FL setting. In this paper, we aim to mitigate such unfairness. However, before showing our method, we need to provide a mathematical definition for the above fairness.

Defining such fairness is challenging. The most direct way is to use the value of the empirical risk function for a different set of participants v_i, i.e., the value of $L_{v_i}(w)$. However, such measurement

[1] Note that in this paper, we assume all the agents will never leave the training process before the training process ends. We leave it as future research to study the case where each agent could join and leave the training.

is unsatisfactory as our fairness should ensure the agents gain more as they join in the training longer, and the function value cannot reveal this relationship. In practice actually, we can use the "difference of accuracy" between different time points to measure the benefit, i.e., fairness. Consider an extreme case as an example, where half of the agents join at the beginning of the training, i.e., t_1 and the other half join the training at the last time point t_S. When the training ends, we hope that the improvement of the accuracy for the first half agents is much greater than for the other half agents. Motivated by this, mathematically we can use the difference in the empirical risk function values to measure the improvement of test accuracy. Based on that, in the following Definition 1, we first define the benefit for group v_i at the current time point t by the difference between the empirical risk function of the group v_i joining the training time point t_i and the current time point t.

Definition 1 (**Benefit**). *Under our dynamic FL setting, for a training algorithm $\mathcal{A}$, the benefit at timepoint t for a group v_i joining training at timepoint t_i ($t > t_i$) is defined as*

$$\mathcal{L}_{t_i}^{v_i}(w_t) = \boldsymbol{L}_{v_i}(w_{t_i}) - \boldsymbol{L}_{v_i}(w_t), \tag{2}$$

where w_{t_i} and w_t is the trained model at timepoint t_i and t respectively. Moreover, we define the benefit agents in v_i get in timepoint t as $\boldsymbol{L}_{v_i}(w_{t-1}) - \boldsymbol{L}_{v_i}(w_t)$.

Based on the definition of benefit, we then propose our desired definition of federated learning fairness criterion with dynamic participants. Generally speaking, we consider a training algorithm is dynamically fair if the benefits of the agents who join earlier are higher than the ones who join later.

Definition 2 (**Absolute Dynamic Fairness**). *Under our dynamic FL setting, for a training algorithm $\mathcal{A}$, it is absolutely dynamically fair if for any two different time points $t_i < t_j$ and any $t > t_j$ we have*

$$\mathcal{L}_{t_i}^{v_i}(w_t) > \mathcal{L}_{t_j}^{v_2}(w_t), \tag{3}$$

where w_t is the trained model at time point t of the algorithm.

Note that the fairness we propose in Definition 2 is real-time, i.e., the definition of fairness in Definition 2 holds regardless of whether t is the last time point of training or the time point in training, as long as $t > t_j$. In practice, we not only want to design absolutely dynamically fair algorithms, but also expect to design develop new fair algorithms that are more fairer than the existing ones. In the following, we quantify such relative fairness between two algorithms, i.e., the algorithm that allows the group that participates longer to get more benefits will be more fair.

Definition 3 (**Relative Dynamic Fairness**). *Under our dynamic FL setting, consider two absolutely dynamically fair training algorithms $\mathcal{A}$ and $\tilde{\mathcal{A}}$, we call algorithm $\mathcal{A}$ is dynamically fairer than algorithm $\tilde{\mathcal{A}}$ if for any two different time points $t_i < t_j$ and any $t > t_j$ we have*

$$\mathcal{L}_{t_i}^{v_i}(w_t) - \mathcal{L}_{t_j}^{v_j}(w_t) > \mathcal{L}_{t_i}^{v_i}(\tilde{w}_t) - \mathcal{L}_{t_j}^{v_j}(\tilde{w}_t), \tag{4}$$

where w_t and $\tilde{w}_t$ is the trained model at time point t of algorithm $\mathcal{A}$ and $\tilde{\mathcal{A}}$ respectively.

It is notable that in Definition 3 we require both $\mathcal{A}$ and $\tilde{\mathcal{A}}$ be absolutely dynamically fair. This is necessary as relative dynamic fairness cannot imply absolute dynamic fairness. Moreover, although there is no data distribution assumption in our previous definition, we can see they are more suitable to the non-i.i.d. data for different agents. This is due to that if all the data are i.i.d. and when m and each n_i ($i \in [m]$) is large enough, then we have $\boldsymbol{L}_{v_i}(w) \approx \boldsymbol{L}_{v_j}(w)$ for any group i and j as both of them are approximately equal to the underlying population risk $\mathbb{E}_{x\sim\mathcal{P}}[\ell(w;x)]$ by the Hoeffding's inequality if the loss function is bounded, where $\mathcal{P}$ is the underlying distribution of the data. And in the ablation study of the experimental part we will also verify this empirically. Thus, in the following parts we will always consider the non-i.i.d. case.

Note that in Definition 1 we use the difference of the empirical loss at two time points to measure the benefit of an agent. However, there could be other ways to define the benefits, such as the relative difference. We will leave them as future work to consider these definitions of benefit.

4 Achieving Dynamic Fairness

In the previous section, we presented the dynamic fairness that we aim to study in this paper. Now we aim to develop methods that is absolutely dynamically fair. Moreover, we want it to be fairer than *Fedavg* (McMahan et al., 2017).

Before diving into details, let us back to the *Fedavg* to see why it may cause unfairness and how to improve its fairness. For simplicity we consider the case where there are only two groups v_1 and v_2 which join the training at t_1 and t_2 respectively, and we assume there is only one agent in each group with the same size of data and each agent will performs one step of Gradient Descent (GD) locally and then send the model to server to be aggregated. Suppose we have already trained the model giving agents in v_1 for long time, and now we achieve the time point t_2 with model w_{t_2}. Now we consider time point $t_2 + 1$. We will show the above variant of *Fedavg* is unfair:

Theorem 1. *Under the above setting, Fedavg is not dynamically fair at the timepoint $t_2 + 1$ if $\|\nabla_w \boldsymbol{L}_{v_2}(w_{t_2})\|_2$ is sufficiently large such that $\|\nabla_w \boldsymbol{L}_{v_2}(w_{t_2})\|_2 \geq \Omega(\|\nabla_w \boldsymbol{L}_{v_1}(w_{t_2})\|_2)$ and $\|\nabla_w \boldsymbol{L}_{v_2}(w_{t_2})\|_2 \geq \Omega(\boldsymbol{L}_{v_1}(w_{t_2}) - \boldsymbol{L}_{v_1}(w_1))$ and $\eta = O(1)$, where η is the stepsize of GD for each agent.*

Note that although in Theorem 1 we need the assume that $\|\nabla_w \boldsymbol{L}_{v_2}(w_{t_2})\|_2$ is sufficiently large, such assumption is quite natural. As we know w_{t_2} is the model trained via L_{v_1} with several rounds, which implies that $\|\nabla_w \boldsymbol{L}_{v_1}(w_{t_2})\|_2$ will be small enough. On the other side, since we get w_{t_2} before v_2 joining and we assume the each data in v_1 and v_2 is non-i.i.d. sampled, thus we have that w_{t_2} will be far from the minimizer of $\boldsymbol{L}_{v_2}(w)$, i.e., $\|\nabla_w \boldsymbol{L}_{v_2}(w_{t_2})\|_2$ is large. Moreover, as w_1 is the initializer at time t_1. Thus, when w_1 is close to the minimizer of $\boldsymbol{L}_{v_1}$ then $\boldsymbol{L}_{v_1}(w_{t_2}) - \boldsymbol{L}_{v_1}(w_1)$ could also be small.

In the following, we will intuitively explain why the previous *Fedavg* is unfair. We assume both $\boldsymbol{L}_{v_1}(w)$ and $\boldsymbol{L}_{v_2}(w)$ are L-smooth, μ-strongly convex and 1-Lispschitz. Then by the assumption of smoothness and strong convexity, and the gradient descent in each agent we have

$$(\eta - \frac{\eta^2 L}{2})\|\nabla_w \boldsymbol{L}_{v_2}(w_{t_2})\|_2^2 \leq \boldsymbol{L}_{v_2}(w_{t_2}) - \boldsymbol{L}_{v_2}(w_{t_2+1}^2) \leq (\eta - \frac{\eta^2 \mu}{2})\|\nabla_w \boldsymbol{L}_{v_2}(w_{t_2})\|_2^2, \tag{5}$$

$$(\eta - \frac{\eta^2 L}{2})\|\nabla_w \boldsymbol{L}_{v_1}(w_{t_2})\|_2^2 \leq \boldsymbol{L}_{v_1}(w_{t_2}) - \boldsymbol{L}_{v_1}(w_{t_2+1}^1) \leq (\eta - \frac{\eta^2 \mu}{2})\|\nabla_w \boldsymbol{L}_{v_1}(w_{t_2})\|_2^2, \tag{6}$$

where η is the stepsize, and $w_{t_2+1}^i$ $(i = 1, 2)$ is the local model in the i-th agent by performing the GD. If the benefit from the aggregation step in the server is sufficiently small, then from (5) we can see the benefit for v_2, which depends on $\Theta(\|\nabla_w \boldsymbol{L}_{v_2}(w_{t_2})\|_2^2)$, could be very large. On the other side, for v_1, $\|\nabla_w \boldsymbol{L}_{v_1}(w_{t_2})\|_2$ is very small, indicating that the benefit they get in this round is quite small. If they did not get large benefit in the previous round before v_2 joining, then the total benift for v_1 will be less than the benefit for v_2, i.e., the algorithm is unfair.

From the previous intuitive analysis we can see that in order to make agents in v_2 get less benefit at time point $t_2 + 1$, we cannot use the GD (or similarly SGD) as it could make the benefit depend on $\Theta(\|\nabla_w \boldsymbol{L}_{v_2}(w_{t_2})\|_2^2)$, which is quite large. Equivalently, the ℓ_2-norm of the gradient plays an important role for the benefits of agents in v_2. Motivated by this, a natural way is performing the **normalized gradient descent** (NGD) instead of GD, i.e., $w_{t_2+1}^2 = w_{t_2} - \eta \frac{\nabla_w \boldsymbol{L}_{v_2}(w_{t_2})}{\|\nabla_w \boldsymbol{L}_{v_2}(w_{t_2})\|_2}$ and $w_{t_2+1}^1 = w_{t_2} - \eta \frac{\nabla_w \boldsymbol{L}_{v_1}(w_{t_2})}{\|\nabla_w \boldsymbol{L}_{v_1}(w_{t_2})\|_2}$. In this case, considering when $\boldsymbol{L}_{v_2}$ is 1-Lipschitz and we have the same stepsize as above, then we have

$$\boldsymbol{L}_{v_2}(w_{t_2+1}) - \boldsymbol{L}_{v_2}(w_{t_2}) \leq \|w_{t_2+1} - w_{t_2}\|_2 \leq 2\eta,$$

i.e., the benefit now is bounded by η, which is much smaller than $\|\nabla_w \boldsymbol{L}_{v_2}(w_{t_2})\|_2$. This indicates that as long as the benefit of v_1 at t_2, i.e., $\boldsymbol{L}_{v_1}(w_{t_1}) - \boldsymbol{L}_{v_i}(w_t) > 2\eta$ then the algorithm will be absolutely dynamically fair at $t_2 + 1$. Moreover since now we limit the benefit for v_2, we can show using NGD is fairer than implementing the above vanilla *Fedavg*.

Theorem 2. *Consider the same setting as in Theorem 1 with normalized GD and fixed w_1, w_{t_2} and η, then if $\|\nabla_w \boldsymbol{L}_{v_2}(w_{t_2})\|_2 \geq \Omega(1)$ we have NGD is dynamically fairer than the above Fedavg at time point $t_2 + 1$.*

Note that although in the previous theorem we only considered the time point $t_2 + 1$. As we can see from the experimental part, our algorithm is fairer than *Fedavg* in practice at each time point. Moreover, the above results relies on the assumption of $\|\nabla_w \boldsymbol{L}_{v_2}(w_{t_2})\|_2 \geq \Omega(1)$. Actually, when $\|\nabla_w \boldsymbol{L}_{v_2}(w)\|_2$ is small enough, the group v_2 will get smaller benefit, i.e., for any $q > 0$ by the

properties of the loss function we have

$$\eta\|\nabla_w \boldsymbol{L}_{v_2}(w_{t_2}+q)\|_2 - \frac{\eta^2 L}{2} \leq \boldsymbol{L}_{v_2}(w_{t_2+q}) - \boldsymbol{L}_{v_2}(\tilde{w}^2_{t_2+q+1}) \leq \eta\|\nabla_w \boldsymbol{L}_{v_2}(w_{t_2+q})\|_2 - \frac{\eta^2\mu}{2},$$
$$\eta\|\nabla_w \boldsymbol{L}_{v_1}(w_{t_2}+q)\|_2 - \frac{\eta^2 L}{2} \leq \boldsymbol{L}_{v_1}(w_{t_2+q}) - \boldsymbol{L}_{v_1}(\tilde{w}^1_{t_2+q+1}) \leq \eta\|\nabla_w \boldsymbol{L}_{v_1}(w_{t_2+q})\|_2 - \frac{\eta^2\mu}{2},$$

where $\tilde{w}^i_{t_2+q+1}$ $(i = 1, 2)$ is the local model in the i-th agent by performing NGD and $w_{t_2+q+1} = \frac{\tilde{w}^1_{t_2+q+1}+\tilde{w}^2_{t_2+q+1}}{2}$. If we ignore the benefit of the aggregation step in the server, then from the previous two results we can see the benefit the v_i group get is $\Theta(\eta\|\nabla_w \boldsymbol{L}_{v_i}(w_{t_2}+q)\|_2)$. Thus, when η and the two gradient norms are small, then the benefits are also small which could be considered to be equal. In total we have that, when $\|\nabla_w \boldsymbol{L}_{v_i}(w_{t_2}+q)\|_2$ is large, then if $\boldsymbol{L}_{v_1}(w_{t_1}) - \boldsymbol{L}_{v_i}(w_t) \geq \omega(\eta)$, our previous algorithm will be dynamically fair. And when $\|\nabla_w \boldsymbol{L}_{v_i}(w_{t_2}+q)\|_2$ becomes sufficiently small then since both groups get almost the same benefit in time point t_2+q. Therefore, our algorithm is still dynamically fair.

Algorithm 1 *Normalized Fedavg*: Two groups v_1, v_2 with joining time point t_1, t_2 $(t_1 = 1, t_1 < t_2)$. $|v|$ indicates the number of clients in group v, $|\mathcal{B}|$ is the local minibatch size, E is the number of local epochs, and η is the learning rate. C is a constant

Server executes:

1: **initialization**: w_1
2: **for** each round t = 1, 2, . . . **do**
3: **if** $t < t_2$ **do** $m \leftarrow \max(C \cdot |v_1|, 1)$ **else do** $m \leftarrow \max(C \cdot (|v_1| + |v_2|), 1)$
4: $S_t \leftarrow$ (random set of m clients)
6: **for** each client $k \in S_t$ **in parallel do**
7: $w^k_{t+1} \leftarrow$ ClientUpdate(k, w_t)
8: **if** $t \geq t_2$ **do**
$w^k_{t+1} \leftarrow w_t - \eta_t \frac{w_t - w^k_{t+1}}{||w_t - w^k_{t+1}||}$ *// Normalization*
9: $w_{t+1} \leftarrow \sum_{k\in S_t} p^i_k w^k_{t+1}$, where p^i_k is the weight where $i = 1$ when $t < t_2$ otherwise $i = 2$.

ClientUpdate (k, w)**:** *// Run on client* k

1: $\mathcal{B} \leftarrow$ (split D_k into batches of size $|\mathcal{B}|$)
2: **for** each local epoch i from 1 to E **do**
3: **for** batch $b \in \mathcal{B}$ **do**
6: $w \leftarrow w - \eta_t \nabla l(w; b)$
7: return w to server

Based on our above idea of normalizing the gradient to limit the benefit for each new agent, we can modify the vanilla *Fedavg* to improve its dynamic fairness, i.e., we propose *Normalized Fedavg* in Algorithm 1 (for simplicity we only present the case where only two groups are trained, and the multi-group case can be easily generalized). Compared with the previous normalized stochastic gradient descent (NSGD) (Zhao et al., 2020; Cutkosky & Mehta, 2020; You et al., 2019; Hazan et al., 2015), there are two critical differences: While in the existing work on NSGD we normalize the gradients each iteration, in Algorithm 1 each agent still uses SGD to train local model and then send model to the server, then the server normalizes these local model updates to update the model (step 8) and then perform the aggregation step (step 9). This is due to that in practice we find that using directly NSGD locally for all agents will make the algorithm hard to be convergent. Thus, before the normalization step, we still need each agent perform SGD (step 7). The second difference it that, where in the previous NSGD based methods we need to calculating the global norm for all parameters of the model and then perform the normalization step. In Algorithm 1, for the normalization step we use layer-wise norm (LN) as using the global norm could lead to non-convergence (see experiments in Section D.1 in Appendix for details).

Although in practice we found *Normalized Fedavg* can indeed improve the dynamic fairness compared with *Fedavg*, its convergence rate is quite slow. The main reason is that the model update of local agents becomes quite small at the early stage after adding group v_2. To address the issue, we simply modify the normalization update step (step 8 in Algorithm 1) by adding a decayed coefficient. We choose the model update norm $\mathcal{G}$ at the time point when group v_2 joined in the training

Algorithm 2 The *Modified Normalized Fedavg* algorithm. The β is a hyperparameter and takes value from $0 \sim 1$. The default value of β is 1. $\mathbf{LN} : (w^{[0]}, ..., w^{[L-1]}) \to (||w^{[0]}||_2, ..., ||w^{[L-1]}||_2)$ is the function to compute the model update norm at each layer.

Server executes:

initialization: w_0, β

for each round $t = 1, 2, \ldots$ **do**

  if $t < t_2$ **do** $m \leftarrow \max(C \cdot |v_1|, 1)$ **else do** $m \leftarrow \max(C \cdot (|v_1| + |v_2|), 1)$

  $S_t \leftarrow$ (random set of m clients)

  if $t = t_2 + 1$ **do** $\mathcal{G} \leftarrow \mathbf{LN}(w_{t-1} - w_t)$

  for each client $k \in S_t$ **in parallel do**

    $w_{t+1}^k \leftarrow \text{ClientUpdate}(k, w_t)$

    if $t \geq t_2 + 1$ **do**

      $w_{t+1}^k \leftarrow w_t - \left(\beta \frac{\mathcal{G}}{1+t-t_2} + (1-\beta)\mathbf{LN}(w_t - w_{t+1}^k)\right) \frac{w_t - w_{t+1}^k}{\mathbf{LN}(w_t - w_{t+1}^k)}$

  $w_{t+1} \leftarrow \sum_{k \in S_t} p_k^i w_{t+1}^k$

ClientUpdate (k, w)**:** same as Algorithm 1

as the initial value of this coefficient, and decay it with the training round increasing. The modified formula for local device model update is

$$w_{t+1}^k := w_t - \frac{\mathcal{G}}{1+t-t_2} \frac{w_t - w_{t+1}^k}{||w_t - w_{t+1}^k||_2}, \tag{7}$$

where $t \geq t_2 + 1, \mathcal{G} = ||w_{t_2} - w_{t_2+1}||$. In order to further enhance the applicability of the algorithm, we introduce another hyperparameter β to combine *Fedavg* and normalized *Fedavg*:

$$w_{t+1}^k := w_t - \left(\beta \frac{\mathcal{G}}{1+t-t_2} + (1-\beta)||w_t - w_{t+1}^k||_2\right) \frac{w_t - w_{t+1}^k}{||w_t - w_{t+1}^k||_2} \tag{8}$$

where the value of β is from 0 to 1. If the value of β is close to 1, the algorithm will be more fair; And if the value is close to 0, the algorithm will converge faster and close to *Fedavg*.

Note that in the above methods we normalize the model update (gradients) to limit each agent's benefit. A natural question is whether we can use other ways. We known that other than normalization, clipping is another commonly used operation in deep learning (e.g., poisoning attacks (Guo et al., 2021; Xie et al., 2021; Panda et al., 2022) and privacy (Truex et al., 2019; Lee & Kifer, 2018)). Motivated by this we propose the *Clipping Fedavg* (Algorithm 3). We find that clipping can also improve the fairness via experiments. However, its improvement compared with *Fedavg* is quite limited. See Section B in Appendix for details.

Actually, due to the simplicity of our idea, our methods are compatible with other fair FL methods. Specifically, we combine our method with the existing methods in fair FL for static participants to additionally guarantee fair treatment for local agents who join the training process at the same time point by minimizing the discrepancy of benefits they receive, i.e, we can achieve *within-group fairness* additionally. See Section C in Appendix for details.

5 Experiments

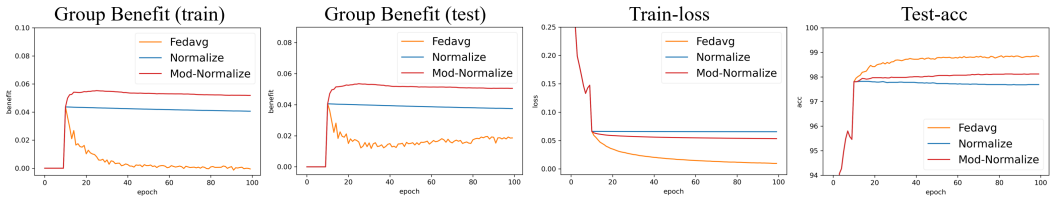
In this section, we will study the practical performance of our proposed algorithms on several benchmark datasets.

Experimental Settings: To verify whether the normalization-based methods can indeed improve dynamic fairness, we design the *Two-groups experiment* to simulate the scenario in which some clients join first (group1) while some join in training at timepoint t_2 (group2). In this type of experiment, each group contains 5 clients.

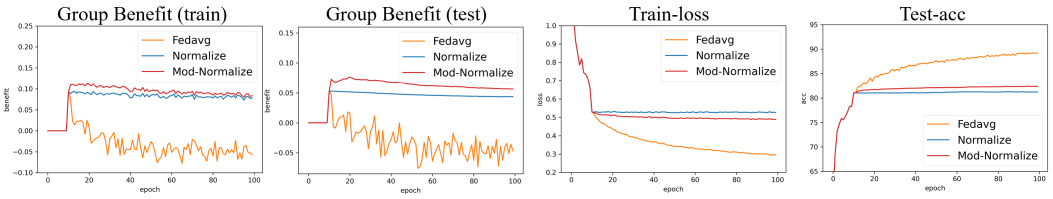
To make our experiments more convincing and applicable, we also design the *Multi-groups experiment*, in which more clients are added to the training at several different timepoints. In detail, there are S groups $\{v_1, ..., v_S\}$ ($S \geq 2$), and each group is added to trained at a specific time point $\{t_1, ..., t_S\}$ ($t_1 = 1$). In the *Multi-groups experiment*, each group contains 3 clients.

For all experiments, all clients run 5 local epochs, 32 local batch-size and $\eta = 1e-2$ in each round. Also, we define a paramter α to control the degree of non-i.i.d of the dataset, i.e., if two groups

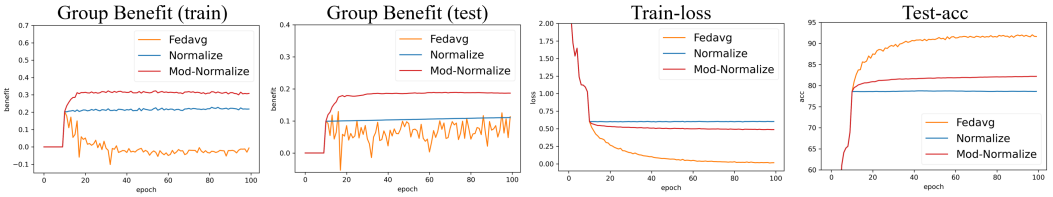
join the training with $\alpha = 0.9$ and the size of class of the dataset is 10, then one group has 90% of the data in 5 classes and the second group has 90% of the data in the other 5 classes. For the *Two-groups experiment*, we choose 10 clients in total and each group contains 5 clients. The second group is added to training in round 10. For the *Multi-groups experiment*, we choose 30 clients in total and each group contains 3 clients). The timepoint each group join in training is in the set $\{0, 10, 20, 30, 40, 50, 60, 70, 80, 90\}$.



(a) Fairness evaluation on MNIST



(b) Fairness evaluation on FMNIST



(c) Fairness evaluation on CIFAR10

Figure 1: Two-groups experiment. (a), (b), (c) represent the evaluation of our proposed algorithm against the original *Fedavg* on the three datasets MNIST, FMNIST, and CIFAR10, respectively.

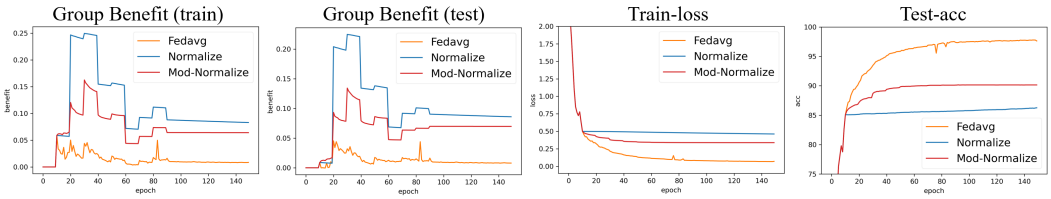
Datasets and Models We use three classical dataset *MNIST* (LeCun et al., 1998), *Fashion MNIST (FMNIST)* (Xiao et al., 2017), and *CIFAR10* (Krizhevsky et al., 2009) and two popular model *LeNet* (LeCun et al., 1998) and *ResNet18* (He et al., 2016) to evaluate our algorithms. Like most of works, we used *LeNet* on *MNIST* and *FMNIST*, and *ResNet18* on *CIFAR10* for evaluation, respectively.

Evaluation Metrics Based on our Definition 1, to better describe the benefits of all groups during the training process, we propose groups benefits (GB) as one of our experimental metrics. This metric shows the difference in the value of the loss between the group that joins later and the group that joins first. A positive value of GB indicates that the algorithm is fair, and larger value indicates better fairness of the algorithm. The metric is defined in (9). GB (train) and GB (test) are calculated from the train dataset and the test dataset, respectively. If there is only one group, we set GB as 0.

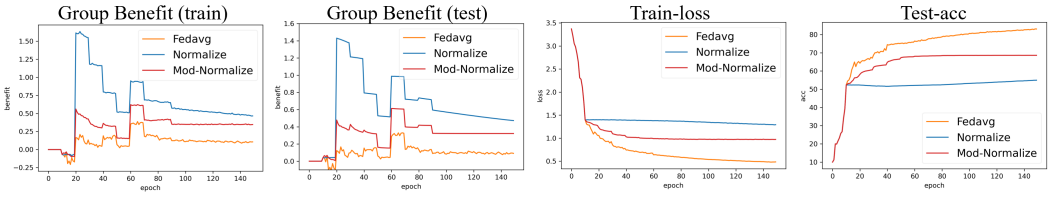
$$GB_t = \frac{1}{n-1} \sum_{i=1}^{n-1} \exp\left(\boldsymbol{L}_{v_{i+1}}(w_t) - \boldsymbol{L}_{v_i}(w_t)\right) - 1 \tag{9}$$

where $n \in [2, N]$ denotes the number of current running groups. It should be noted that the metric here does not exactly follow Definition 1, the reason can be seen in section E in Appendix.

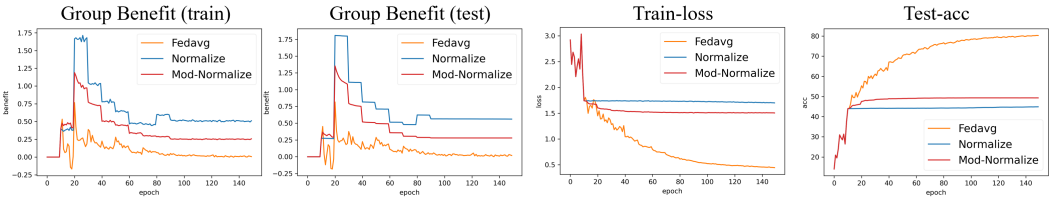
Main Experiment Results The experiment results are shown in the Figure 1 and 2. It can be seen that Algorithm 1 and 2 exhibit much higher benefits than *Fedavg*, and their benefits eventually converge to a positive large value on all the datasets. In contrast, *Fedavg* makes the value of GB smaller



(a) Fairness evaluation on MNIST



(b) Fairness evaluation on FMNIST



(c) Fairness evaluation on CIFAR10

Figure 2: Multi-groups experiment. (a), (b), (c) represent the evaluation of our proposed algorithm against the original *Fedavg* on the three datasets MNIST, FMNIST, and CIFAR10, respectively.

or even negative (absolute unfairness). A noteworthy phenomenon is that for *Fedavg*, compared with its GB values that keep decreasing during the training process, the GB values will increase slightly during the testing process for both datasets (*MNIST* and *CIFAR10*). However, the increased values are still much smaller than those of our methods. Remarkably, we find that Algorithm 2 is fairer than Algorithm 1 in Figure 1, but Figure 2 shows the exact opposite phenomenon. Therefore, for those two algorithms, we cannot conclude which algorithm outperforms measured by GB, which indicates that our strategies in Algorithm 2 do not significantly reduce fairness compared with Algorithm 1.

Figure 1 and 2 both illustrate a negative correlation between fairness and convergence rate. Algorithms 1 and 2 have better fairness performance but their convergence rates are slower than *Fedavg*. Meanwhile, we can conclude that Algorithm 2 converges faster than Algorithm 1, which shows the effectiveness of our proposed Algorithm 2.

To summary, both of Algorithm 1 and 2 demonstrate greater dynamic fairness than *Fedavg*. Besides, Algorithm 2 can achieve a faster convergence rate than Algorithm 1 while maintaining a similar dynamic fairness with Algorithm 1.

We defer the ablation study to Section D.1 in Appendix due to space limit.

6 Conclusion

In this paper, we focused on the fairness in the setting of Federated Learning with dynamic participants, meaning that clients can join in the training at different time points. We proposed a new definition of federated learning fairness namely dynamic fairness to guarantee higher benefits for local agents who participate in the FL model training for longer time periods than those do not. We developed algorithms with normalization to guarantee the dynamic fairness based on *Fedavg*. Furthermore, we improved the efficiency of *Normalized Fedavg* via some strategies. Intensive experiment results showed that our methods are dynamically fair. And specifically, our algorithms are fairer than *Fedavg*.

REFERENCES

Ashok Cutkosky and Harsh Mehta. Momentum improves normalized sgd. In *International conference on machine learning*, pp. 2260–2268. PMLR, 2020. 4

gboard. Federated learning: Collaborative machine learning without centralized training data. https://ai.googleblog.com/2017/04/federated-learning-collaborative.html. 1

gdpr. General data protection regulation (gdpr). https://gdpr.eu/what-is-gdpr/. 1

Yifan Guo, Qianlong Wang, Tianxi Ji, Xufei Wang, and Pan Li. Resisting distributed backdoor attacks in federated learning: A dynamic norm clipping approach. In *2021 IEEE International Conference on Big Data (Big Data)*, pp. 1172–1182. IEEE, 2021. 4, B

Elad Hazan, Kfir Levy, and Shai Shalev-Shwartz. Beyond convexity: Stochastic quasi-convex optimization. *Advances in neural information processing systems*, 28, 2015. 4

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016. 5

Tiansheng Huang, Weiwei Lin, Wentai Wu, Ligang He, Keqin Li, and Albert Y Zomaya. An efficiency-boosting client selection scheme for federated learning with fairness guarantee. *IEEE Transactions on Parallel and Distributed Systems*, 32(7):1552–1564, 2020. 2

Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *Foundations and Trends® in Machine Learning*, 14(1–2):1–210, 2021. 1

Jakub Konečnỳ, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016. 1

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 4, 5

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 4, 5

Jaewoo Lee and Daniel Kifer. Concentrated differentially private gradient descent with adaptive per-iteration privacy budget. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1656–1665, 2018. 4, B

Tian Li, Maziar Sanjabi, Ahmad Beirami, and Virginia Smith. Fair resource allocation in federated learning. *arXiv preprint arXiv:1905.10497*, 2019. 2, C, C, C

Tian Li, Shengyuan Hu, Ahmad Beirami, and Virginia Smith. Ditto: Fair and robust federated learning through personalization. In *International Conference on Machine Learning*, pp. 6357–6368. PMLR, 2021. 2

Lingjuan Lyu, Xinyi Xu, Qian Wang, and Han Yu. Collaborative fairness in federated learning. In *Federated Learning*, pp. 189–204. Springer, 2020. 1, 2

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pp. 1273–1282. PMLR, 2017. 2, 4

Ashwinee Panda, Saeed Mahloujifar, Arjun Nitin Bhagoji, Supriyo Chakraborty, and Prateek Mittal. Sparsefed: Mitigating model poisoning attacks in federated learning with sparsification. In *International Conference on Artificial Intelligence and Statistics*, pp. 7587–7624. PMLR, 2022. 4, B

Matthias Paulik, Matt Seigel, Henry Mason, Dominic Telaar, Joris Kluivers, Rogier van Dalen, Chi Wai Lau, Luke Carlson, Filip Granqvist, Chris Vandevelde, et al. Federated evaluation and tuning for on-device personalization: System design & applications. *arXiv preprint arXiv:2102.08503*, 2021. 1

Yuxin Shi, Han Yu, and Cyril Leung. A survey of fairness-aware federated learning. *arXiv preprint arXiv:2111.01872*, 2021. 1

Stacey Truex, Nathalie Baracaldo, Ali Anwar, Thomas Steinke, Heiko Ludwig, Rui Zhang, and Yi Zhou. A hybrid approach to privacy-preserving federated learning. In *Proceedings of the 12th ACM workshop on artificial intelligence and security*, pp. 1–11, 2019. 4, B

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017. 4, 5

Chulin Xie, Minghao Chen, Pin-Yu Chen, and Bo Li. Crfl: Certifiably robust federated learning against backdoor attacks. In *International Conference on Machine Learning*, pp. 11372–11382. PMLR, 2021. 4, B

Yuexiang Xie, Zhen Wang, Daoyuan Chen, Dawei Gao, Liuyi Yao, Weirui Kuang, Yaliang Li, Bolin Ding, and Jingren Zhou. Federatedscope: A comprehensive and flexible federated learning platform via message passing. *arXiv preprint arXiv:2204.05011*, 2022. 1

Yang You, Jing Li, Jonathan Hseu, Xiaodan Song, James Demmel, and Cho-Jui Hsieh. Reducing bert pre-training time from 3 days to 76 minutes. *arXiv preprint arXiv:1904.00962*, 2019. 4

Han Yu, Zelei Liu, Yang Liu, Tianjian Chen, Mingshu Cong, Xi Weng, Dusit Niyato, and Qiang Yang. A fairness-aware incentive scheme for federated learning. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, pp. 393–399, 2020. 2

Muhammad Bilal Zafar, Isabel Valera, Manuel Gomez Rodriguez, and Krishna P Gummadi. Fairness beyond disparate treatment & disparate impact: Learning classification without disparate mistreatment. In *Proceedings of the 26th international conference on world wide web*, pp. 1171–1180, 2017. 2

Yufeng Zhan, Jie Zhang, Zicong Hong, Leijie Wu, Peng Li, and Song Guo. A survey of incentive mechanism design for federated learning. *IEEE Transactions on Emerging Topics in Computing*, 2021. 1

Shen-Yi Zhao, Yin-Peng Xie, and Wu-Jun Li. Stochastic normalized gradient descent with momentum for large batch training. *arXiv preprint arXiv:2007.13985*, 2020. 4

Pengyuan Zhou, Pei Fang, and Pan Hui. Loss tolerant federated learning. *arXiv preprint arXiv:2105.03591*, 2021. 1, 2

A OMITTED PROOFS

Proof of Theorem 1. In *Fedavg* from the server side it computes $w_{t_2+1} = \frac{w^2_{t_1+1}+w^2_{t_2+1}}{2}$, where $w^2_{t_1+1} = w_{t_2} - \eta\nabla_w \boldsymbol{L}_{v_2}(w_{t_2})$ and $w^1_{t_1+1} = w_{t_2} - \eta\nabla_w \boldsymbol{L}_{v_1}(w_{t_2})$ with the stepsize η. Thus, from the first order approximation we have

$$\begin{aligned}\boldsymbol{L}_{v_1}(w_{t_2+1}) &= \boldsymbol{L}_{v_1}(w_{t_2}) - \eta\nabla_w \boldsymbol{L}_{v_1}(w_{t_2}) \cdot \frac{\nabla_w \boldsymbol{L}_{v_1}(w_{t_2}) + \nabla_w \boldsymbol{L}_{v_2}(w_{t_2})}{2} \\ &\quad + o\left[||w_{t_2+1} - w_{t_2}||^2\right] \end{aligned} \tag{10}$$

$$\begin{aligned}\boldsymbol{L}_{v_2}(w_{t_2+1}) &= \boldsymbol{L}_{v_2}(w_{t_2}) - \eta\nabla_w \boldsymbol{L}_{v_2}(w_{t_2}) \cdot \frac{\nabla_w \boldsymbol{L}_{v_1}(w_{t_2}) + \nabla_w \boldsymbol{L}_{v_2}(w_{t_2})}{2} \\ &\quad + o\left[||w_{t_2+1} - w_{t_2}||^2\right] \end{aligned} \tag{11}$$

Thus we have

$$[\boldsymbol{L}_{v_2}(w_{t_2}) - \boldsymbol{L}_{v_2}(w_{t_2+1})] - [\boldsymbol{L}_{v_1}(w_{t_2}) - \boldsymbol{L}_{v_1}(w_{t_2+1})] \approx \eta\|\nabla_w \boldsymbol{L}_{v_2}(w_{t_2})\|_2^2 - \eta\|\nabla_w \boldsymbol{L}_{v_1}(w_{t_2})\|_2^2$$

Thus, based on the definition 1 the difference of benefit for agents in v_2 and benefit for agents in v_1 is approxiamtely equal to

$$\eta[\|\nabla_w \boldsymbol{L}_{v_2}(w_{t_2})\|_2^2 - \|\nabla_w \boldsymbol{L}_{v_1}(w_{t_2})\|_2^2] + [\boldsymbol{L}_{v_1}(w_{t_2}) - \boldsymbol{L}_{v_1}(w_1)].$$

Thus, when $\|\nabla_w \boldsymbol{L}_{v_2}(w_{t_2})\|_2 \geq \Omega(\|\nabla_w \boldsymbol{L}_{v_1}(w_{t_2})\|_2)$ and $\|\nabla_w \boldsymbol{L}_{v_2}(w_{t_2})\|_2 \geq \Omega(\boldsymbol{L}_{v_1}(w_{t_2}) - \boldsymbol{L}_{v_1}(w_1))$ then the benefit for v_2 is larger and the algorithm is no longer fair. □

Proof of Theorem 2. In *Fedavg* from the server side it computes $w_{t_2+1} = \frac{w^1_{t_2+1}+w^2_{t_2+1}}{2}$, where $w^2_{t_1+1} = w_{t_2} - \eta\frac{\nabla_w \boldsymbol{L}_{v_2}(w_{t_2})}{\|\nabla_w \boldsymbol{L}_{v_2}(w_{t_2})\|_2}$ and $w^1_{t_2+1} = w_{t_2} - \eta\frac{\nabla_w \boldsymbol{L}_{v_1}(w_{t_2})}{\|\nabla_w \boldsymbol{L}_{v_1}(w_{t_2})\|_2}$ with the stepsize η. Thus, from the first order approximation we have

$$\begin{aligned}\boldsymbol{L}_{v_1}(w_{t_2+1}) &= \boldsymbol{L}_{v_1}(w_{t_2}) - \frac{\eta}{2}\nabla_w \boldsymbol{L}_{v_1}(w_{t_2}) \cdot \left(\frac{\nabla_w \boldsymbol{L}_{v_1}(w_{t_2})}{\|\nabla_w \boldsymbol{L}_{v_1}(w_{t_2})\|_2} + \frac{\nabla_w \boldsymbol{L}_{v_2}(w_{t_2})}{\|\nabla_w \boldsymbol{L}_{v_2}(w_{t_2})\|_2}\right) \\ &\quad + o\left[||w_{t_2+1} - w_{t_2}||^2\right] \end{aligned} \tag{12}$$

$$\begin{aligned}\boldsymbol{L}_{v_2}(w_{t_2+1}) &= \boldsymbol{L}_{v_2}(w_{t_2}) - \frac{\eta}{2}\nabla_w \boldsymbol{L}_{v_2}(w_{t_2}) \cdot \left(\frac{\nabla_w \boldsymbol{L}_{v_1}(w_{t_2})}{\|\nabla_w \boldsymbol{L}_{v_1}(w_{t_2})\|_2} + \frac{\nabla_w \boldsymbol{L}_{v_2}(w_{t_2})}{\|\nabla_w \boldsymbol{L}_{v_2}(w_{t_2})\|_2}\right) \\ &\quad + o\left[||w_{t_2+1} - w_{t_2}||^2\right] \end{aligned} \tag{13}$$

Thus we have

$$\begin{aligned}&[\boldsymbol{L}_{v_2}(w_{t_2}) - \boldsymbol{L}_{v_2}(w_{t_2+1})] - [\boldsymbol{L}_{v_1}(w_{t_2}) - \boldsymbol{L}_{v_1}(w_{t_2+1})] \\ &\approx \frac{\eta}{2}(\|\nabla_w \boldsymbol{L}_{v_2}(w_{t_2})\|_2 - \|\nabla_w \boldsymbol{L}_{v_1}(w_{t_2})\|_2)(1 + \frac{\nabla_w \boldsymbol{L}_{v_1}(w_{t_2}) \cdot \nabla_w \boldsymbol{L}_{v_2}(w_{t_2})}{\|\nabla_w \boldsymbol{L}_{v_1}(w_{t_2})\|_2\|\nabla_w \boldsymbol{L}_{v_2}(w_{t_2})\|_2})\end{aligned}$$

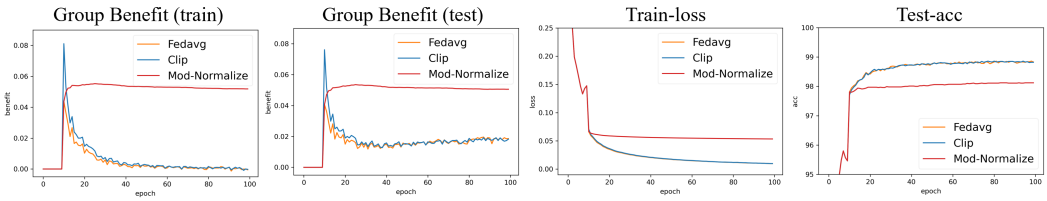
Thus, based on Definition 1 the difference of benefit for agents in v_2 and benefit for agents in v_1 is

$$\begin{aligned}&\frac{\eta}{2}(\|\nabla_w \boldsymbol{L}_{v_2}(w_{t_2})\|_2 - \|\nabla_w \boldsymbol{L}_{v_1}(w_{t_2})\|_2)(1 + \frac{\nabla_w \boldsymbol{L}_{v_1}(w_{t_2}) \cdot \nabla_w \boldsymbol{L}_{v_2}(w_{t_2})}{\|\nabla_w \boldsymbol{L}_{v_1}(w_{t_2})\|_2\|\nabla_w \boldsymbol{L}_{v_2}(w_{t_2})\|_2}) \\ &\quad + [\boldsymbol{L}_{v_1}(w_{t_2}) - \boldsymbol{L}_{v_1}(w_1)].\end{aligned} \tag{14}$$

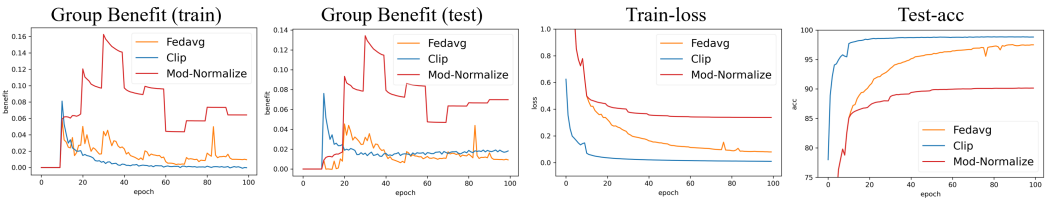
And it is smaller than the difference in the case of Theorem 1 with fixed w_1, w_{t_2} and η when $\|\nabla_w \boldsymbol{L}_{v_2}(w_{t_2})\|_2 \geq 2$. Thus, it is more fairer than Theorem 1. □

B CLIPPING FEDAVG AND EXPERIMENT

Note that in the above methods we normalize the model update (gradients) to improve algorithms' fairness. A natural question is whether we can use other ways. Other than normalization, clipping is another commonly used operation in deep learning (e.g., poisoning attacks (Guo et al., 2021; Xie et al., 2021; Panda et al., 2022) and privacy (Truex et al., 2019; Lee & Kifer, 2018)). Motivated by this, we propose the *Clipping Fedavg* algorithm, whose complete pseudo-code is given in Algorithm 3. With experiments in figure 3, we found that on the whole training stage, it is hard to argue that clipping can improve the fairness of the algorithm. we can see that although algorithm 3 can maintain a high level of fairness in the early stage after new group joins, it is not significantly different from *Fedavg* in the later stage. Therefore, we do not recommend using algorithm 3 to improve fairness in practical applications.



(a) Two-groups experiment on MNIST



(b) Multi-groups experiment on MNIST

Figure 3: Two-groups and Multi-groups experiments for definition 2, 3. (a), (b) represent the Two-groups and Multi-groups experiments of our proposed clipping algorithm against the original *Fedavg* on MNIST, respectively.

Algorithm 3 The *Clipping Fedavg* algorithm. The running clients are indexed by k, and η is the learning rate. $\mathbf{LM} : (w^{[0]}, ..., w^{[L-1]}) \rightarrow (max(w^{[0]}), ..., max(w^{[L-1]}))$ is the function to compute the max value of model update at each layer, and $\mathbf{Clip}_g : (w^{[0]}, ..., w^{[L-1]}) \rightarrow (threshold(w^{[0]}, \pm g), ..., threshold(w^{[L-1]}, \pm g))$ is the function to clip the paramters at each layer with specified value, where $threshold(w, \pm g)$ can limit w with $\pm g$.

Server executes:

initialization: w_0
for each round t = 1, 2, . . . **do**
  if $t < t_2$ **do** $m \leftarrow max(C \cdot |v_1|, 1)$ **else do** $m \leftarrow max\left(C \cdot (|v_1| + |v_2|), 1\right)$
  $S_t \leftarrow$ (random set of m clients)
  if $t = t_2$ **do** $\mathcal{G} \leftarrow \mathbf{LM}(w_{t-1} - w_t)$
  for each client $k \in S_t$ **in parallel do**
    $w_{t+1}^k \leftarrow$ ClientUpdate(k, w_t)
    if $t \geq t_2$ **do**
      $w_{t+1}^k \leftarrow w_t - \mathbf{Clip}_{\mathcal{G}}(w_t - w_{t+1}^k)$ *// Clipping*
  $w_{t+1} \leftarrow \sum_{k=1}^{S_t} p_k w_{t+1}^k$
  if $t \geq t_2$ **do** compute test loss: $L_{v_1}(w_t)$, $L_{v_2}(w_t)$

ClientUpdate (k, w)**:** same to algorithm 1

C FURTHER EXPANSION OF THE FAIRNESS DEFINITION

Additionally, we combine our method with the existing methods in fair FL for static participants. Via such approach, we can minimize the discrepancy of benefits for the agents who join the training process at the same time point, and guarantee a fair treatment for them, i.e, we can achieve *within-group fairness*.

By combining the fairness definition of Li et al. (2019) and ours, we extend the definition 2, 3 to definition 4, including *between-group fairness* (guarantees that agents with longer participating time benefit more) and *within-group fairness* (guarantees performance uniformity for agents with the same participating time).

Definition 4. ***Dynamic Fairness (Extended)****: Under our dynamic FL setting, for a training algorithm $\mathcal{A}$, it is absolutely dynamically fair if for any two different time points $t_i < t_j$ and any $t > t_j$ we have*

Between-group fairness:

$$\mathcal{L}_{t_1}^{v_1}(w_t) > \mathcal{L}_{t_2}^{v_2}(w_t) \tag{15}$$

Within-group fairness:

$$\mathbf{std}_{k \in v} \{F_k(w)\} \to 0 \tag{16}$$

Under our dynamic FL setting, consider two absolutely dynamically fair training algorithms $\mathcal{A}$ and $\tilde{\mathcal{A}}$, we call algorithm $\mathcal{A}$ is dynamically fairer than algorithm $\tilde{\mathcal{A}}$ if for any two different time points $t_i < t_j$ and any $t > t_j$ we have

Between-group fairness:

$$\mathcal{L}_{t_1}^{v_1}(w_t) - \mathcal{L}_{t_2}^{v_2}(w_t) > \mathcal{L}_{t_1}^{v_1}(\tilde{w}_t) - \mathcal{L}_{t_2}^{v_2}(\tilde{w}_t) \tag{17}$$

Within-group fairness:

$$\mathbf{std}_{k \in v_i} \{F_k(w)\} < \mathbf{std}_{k \in v_i} \{F_k(\tilde{w})\} \tag{18}$$

Here $\mathcal{L}_{t_i}^{v_i}(w_t)$ is defined in Definition 1, $\mathbf{std}_{k \in v} \{F_k(w)\}$ denotes the standard deviation of the test loss of all devices in group v_i, and v_i is anyone of all groups currently participating.

We further modified our algorithm by combining above algorithms with *q-Fedavg* (Li et al. (2019)), which is an excellent solution of *within-group fairness*. The pseudo-code of our modified algorithm is given in algorithm 4.

Algorithm 4 We merged our methods (step 8) into the *q-Fedavg*. The notation k is the index of running clients, w_t is the global model at current round t, and η is the learning rate. $\mathbf{LN} : (w^{[0]}, ..., w^{[L-1]}) \to (||w^{[0]}||, ..., ||w^{[L-1]}||)$ is the function used to compute the model update norm at each layer. q is a hyperparameter of *q-Fedavg*, and its default value is 0.1.

Server executes:

1: **initialization**: w_0
2: **for** each round t = 1, 2, . . . **do**
3: **if** $t < t_2$ **do** $m \leftarrow max(C \cdot |v_1|, 1)$ **else do** $m \leftarrow max\left(C \cdot (|v_1| + |v_2|), 1\right)$
4: $S_t \leftarrow$ (random set of m clients)
6: **for** each client $k \in S_t$ **in parallel do**
7: $w_{t+1}^k, F_k(w_t) \leftarrow$ ClientUpdate(k, w_t)
8: $w_{t+1}^k \leftarrow$ Our operation (based on algorithm 1, 2, 3)
9: $\triangle_t^k = F_k^q(w_t) * (w_t - w_{t+1}^k)$
10: $h_t^k = qF_k^{q-1}(w_t)\mathbf{LN}(w_t - w_{t+1}^k) + F_k^q(w_t)$
11: $w_{t+1} \leftarrow w_t - \sum_{v \in S_t} \sum_{k \in v} p_k \frac{\triangle_t^k}{h_t^k}$
12: **if** $t \geq t_2$ **do** compute test loss: $L_{v_1}(w_t)$, $L_{v_2}(w_t)$

ClientUpdate (k, w)**:** same to algorithm 1

Then we provide the experiment results for Algorithm 4.

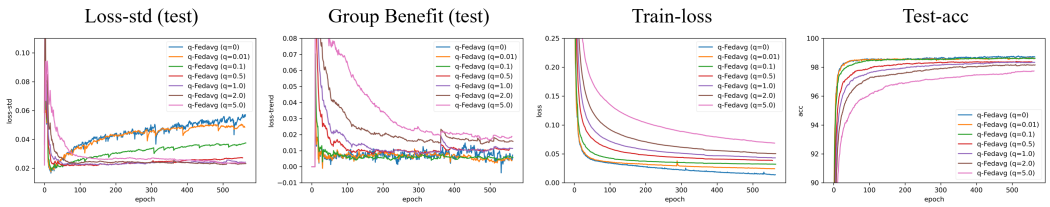


Figure 4: Experiment to verify if the *q-Fedavg* is still valid under our *Two-groups experiment* condition (section 5) and select the most suitable q with algorithm 4

We provide the experiment results of Algorithm 4.

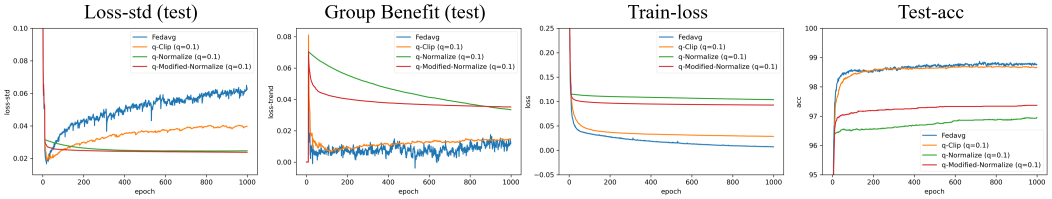


Figure 5: Two-groups experiment on MNIST for definition 4.

First, we define an evaluation metric loss std (LS) for *within-group fairness*. This metric indicates the level of performance uniformity across clients within the same group. A lower value of the metric indicates higher uniformity. If only one group runs, we set LS to 0.

$$LS_t = \frac{1}{n}\sum_{i=1}^{n}\sqrt{\sum_{k=1}^{|v_i|}\frac{(F_t^k - L_t^{v_i})^2}{|v_i|}} \tag{19}$$

Second, we verified that the *q-Fedavg* Li et al. (2019) is still valid in our *Two-groups experiment* condition in figure 4(section 5). We then selected $q = 0.1$ that best fits algorithm 4.

Last, as shown in figure 5, we find that the algorithm 4 can improve both *between-groups fairness* (lower LS) and *within-group fairness* (greater Group benefit).

D Additional Experimental Results

D.1 Ablation Study

In ablation study, we use the experiment results of "Two-groups experiment" (*MNIST*) as the control group. We explore the effects of three key variables (α, global or layer norm, and β) on the experiment results.

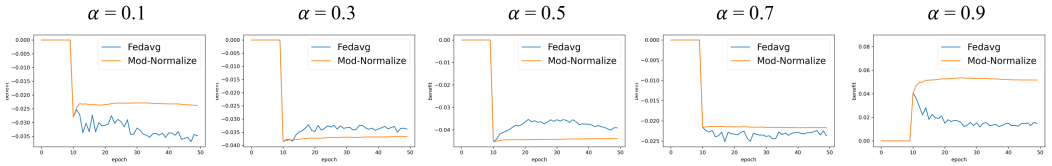


Figure 6: Ablation experiment on MNIST for impact of α value.

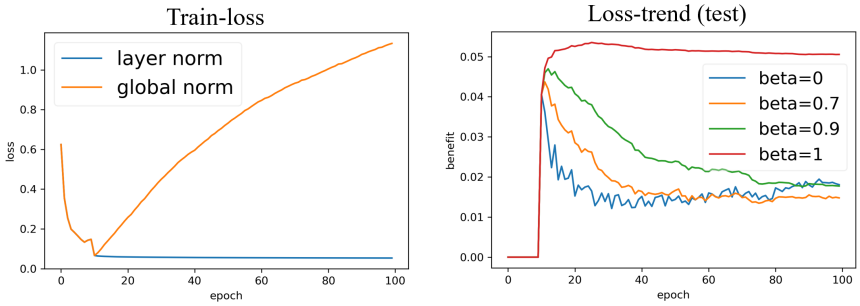
Impact of α. Figure 6 shows that in a scenario of a low level of non-iid, fairness is not guaranteed regardless of whether normalization is implemented or not. And only with strong non-iid, normalization can guarantee fairness and is fairer than *Fedavg*, which is consistent with our idea in the previous section.

Impact of global / layer norm. We investigate whether the global parameter norm of the model update or the per-layer parameter norm should be used in algorithm 1 and 2. As seen in figure 7(a), the global norm for all parameters of model update in algorithm 2 prevents the model from converging. On the contrary, the layer norm (LN) is able to make the model converge.

Impact of β for algorithm 2. Due to a necessary trade-off between fairness and convergence speed in practical applications, we expect the hyperparameter β to regulate the degree of fairness of algorithm 2. Figure 7(b) also demonstrates that the effect of adjusting β is consistent with our expectation.

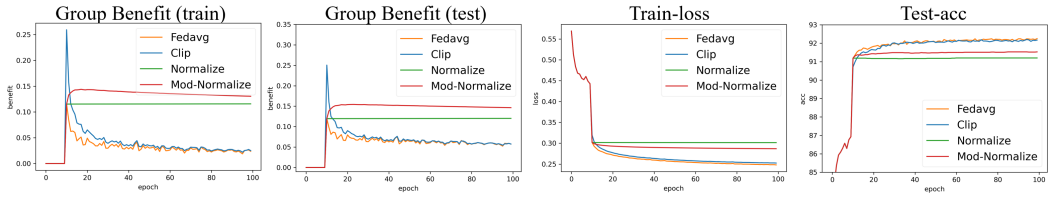
D.2 Evaluation of other models

We redid the Two-group experiment using Linear-regression and 2-layer neural network and got the same results (figure 8) as Lenet. This proves that our method is not limited by the model

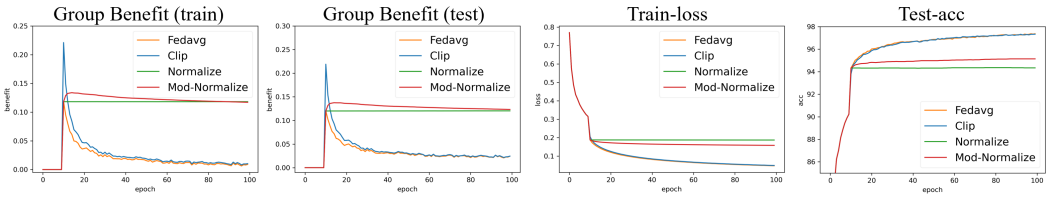


(a) Impact of global / layer norm (b) Impact of β

Figure 7: Three ablation experiments on MNIST dataset. (a) represents the impact of global or layer norm in algorithm 2, and (b) illustrates the impact of β in algorithm 2



(a) Fairness evaluation on MNIST with Linear-regression



(b) Fairness evaluation on MNIST with 2-layer neural network

Figure 8: Two-groups experiments for definition 2. (a), (b) represent the Linear-regression and 2-layer neural network are used to evaluate the fairness of our proposed algorithms against the original *Fedavg* on MNIST, respectively.

E SUPPLEMENTAL NOTION

Explanation of implementing Evaluation Metric "Group Benefit" If we follow the benefit definition (2), then the equation (9) should be rewritten as

$$\begin{aligned} GB_t &= \frac{1}{n-1}\sum_{i=1}^{n-1}\exp\left(\mathcal{L}_{t_i}^{v_i}(w_t) - \mathcal{L}_{t_{i+1}}^{v_{i+1}}(w_t)\right) - 1 \\ &= \frac{1}{n-1}\sum_{i=1}^{n-1}\exp\left(\underbrace{(\boldsymbol{L}_{v_i}(w_{t_i}) - \boldsymbol{L}_{v_{i+1}}(w_{t_{i+1}}))}_{\text{Our proposed algorithms do not change it}} + (\boldsymbol{L}_{v_{i+1}}(w_t) - \boldsymbol{L}_{v_i}(w_t))\right) - 1 \end{aligned} \tag{20}$$

However, we find that the former term $\boldsymbol{L}_{v_i}(w_{t_i})$ in (2) is much larger than the latter term $\boldsymbol{L}_{v_i}(w_t)$ in the actual experiments, which makes the actual benefit $\mathcal{L}_{t_i}^{v_i}(w_t)$ always be close to $\boldsymbol{L}_{v_i}(w_{t_i})$ and remains constant. Moreover, we find that $\boldsymbol{L}_{v_i}(w_{t_i}) - \boldsymbol{L}_{v_{i+1}}(w_{t_{i+1}})$ in (20) is the same for our algorithms and *Fedavg*, so we remove the former term $(\boldsymbol{L}_{v_i}(w_{t_i}) - \boldsymbol{L}_{v_{i+1}}(w_{t_{i+1}}))$ in the metric (20) and change the metric to (9).