FedAA: A Reinforcement Learning Perspective on Adaptive Aggregation for Fair and Robust Federated Learning

Jialuo He[1,2], **Wei Chen**[3], **Xiaojin Zhang**[1,*]

[1] National Engineering Research Center for Big Data Technology and System
Services Computing Technology and System Lab, Cluster and Grid Computing Lab
School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, 430074, China
[2] School of Microelectronics and Communication Engineering, Chongqing University, Chongqing, 400044, China
[3] School of Software Engineering, Huazhong University of Science and Technology, Wuhan, 430074, China
xiaojinzhang@hust.edu.cn

Abstract

Federated Learning (FL) has emerged as a promising approach for privacy-preserving model training across decentralized devices. However, it faces challenges such as statistical heterogeneity and susceptibility to adversarial attacks, which can impact model robustness and fairness. Personalized FL attempts to provide some relief by customizing models for individual clients. However, it falls short in addressing server-side aggregation vulnerabilities. We introduce a novel method called **FedAA**, which optimizes client contributions via **A**daptive **A**ggregation to enhance model robustness against malicious clients and ensure fairness across participants in non-identically distributed settings. To achieve this goal, we propose an approach involving a Deep Deterministic Policy Gradient-based algorithm for continuous control of aggregation weights, an innovative client selection method based on model parameter distances, and a reward mechanism guided by validation set performance. Empirically, extensive experiments demonstrate that, in terms of robustness, **FedAA** outperforms the state-of-the-art methods, while maintaining comparable levels of fairness, offering a promising solution to build resilient and fair federated systems.

Code — https://github.com/Gp1g/FedAA.
Extended version — https://arxiv.org/abs/2402.05541

Introduction

Federated learning (FL) is an emerging paradigm that enables collaborative model training while protecting the privacy of each participant's data (McMahan et al. 2017; Kaissis et al. 2020; Tan et al. 2022; Cheng et al. 2020). Despite its potential, FL faces challenges stemming from data heterogeneity across clients and susceptibility to malicious attacks. These factors can lead to suboptimal global models that favor certain clients over others or models that are not robust to adversarial behaviors, undermining the core principles of FL.

Personalized FL (PFL), represented by Ditto and lp-proj (Li et al. 2021b; Lin et al. 2022), emerged as a response to these challenges, tailoring models to individual client characteristics (Kulkarni, Kulkarni, and Pant 2020; Tan et al. 2022). Nonetheless, personalization typically focuses on the client level and does not adequately mitigate risks during the server-led aggregation phase. Similarly, existing works that incorporate notions of robustness (Chen, Su, and Xu 2017; Blanchard et al. 2017; Xie, Koyejo, and Gupta 2018; Mohri, Sivek, and Suresh 2019; Hu et al. 2022) and fairness (Li et al. 2020; Ezzeldin et al. 2023; Li et al. 2021a) often do so separately, lacking an integrated approach that provides both concerns simultaneously.

In this paper, we present a novel method called Federated Adaptive Aggregation (FedAA), which employs deep reinforcement learning (DRL) to dynamically adjust the influence of each client's update during aggregation, thus balancing robustness and fairness at the server level. Our contributions are three-fold:

- Firstly, we propose a novel FL framework, FedAA, employing DRL to enhance both robustness and fairness via dynamically optimizing client contributions in FL.
- Secondly, we conduct comprehensive experiments to validate the efficacy of the FedAA model. The results demonstrate significant improvements in robustness and maintain comparable levels of fairness against state-of-the-art (SOTA) methods.
- Lastly, we perform ablation studies to pinpoint contributions of key components to model performance and provide insights into FedAA's fairness mechanisms, particularly in handling diverse client data distributions.

The remainder of this paper is organized as follows: First, we review the related works. Then, we provide a detailed description of the proposed FedAA framework, including the client selection algorithm, the DDPG-based optimization process, and the reward formulation. Next, we present our experimental setup and discuss the results. Finally, we conclude with suggestions for future research directions.

Related Work

Robustness in Federated Learning. The robustness of FL models is a critical area of research, given the potential for adversarial interactions within decentralized training environments. Adversarial attacks, such as data poisoning and

*Corresponding author.

model update poisoning, have been identified as significant threats to the integrity of FL systems. Data poisoning introduces false information into the training datasets (Biggio, Nelson, and Laskov 2012; Li et al. 2016; Rubinstein et al. 2009; Jagielski et al. 2018; Suciu et al. 2018; Fang et al. 2020; Dai and Li 2023), while model update poisoning, one is Byzantine attacks, involves an α-fraction (typical $\alpha < 0.5$) of clients acting maliciously to disrupt the learning process. Various Byzantine robust SGD methods have been proposed to mitigate these threats to enhance the resilience of FL models against such attacks (Chen, Su, and Xu 2017; Blanchard et al. 2017; Xie, Koyejo, and Gupta 2018). This paper aligns with the robustness definition by Li et al. (2021b) and considers the following attack models for evaluation:

Definition 1 (Robustness). In the case of a certain Byzantine attack, if model w_1 achieves higher mean test accuracy across benign clients compared to model w_2, then we say that model w_1 is more robust than model w_2. We employ three common attack methods to evaluate the robustness of our model. We use $\tilde{\mathbf{w}}_k$ to represent the malicious messages sent by client k.

• **Same-value attacks**: Malicious client k sends parameters can be denoted as $\tilde{\mathbf{w}}_k = m\mathbf{1}$, where $m \sim \mathcal{N}(0, \tau^2)$ represents the intensity of attack, $\mathbf{1}$ is a vector of ones, with the same size of parameters as the benign clients.

• **Sign-flipping attacks**: Malicious client k sends sign-flipped and scaled messages, which can be represented as $\tilde{\mathbf{w}}_k = -|m|\tilde{\mathbf{w}}'_k$, where $\tilde{\mathbf{w}}'_k$ denotes the correct updates, $m \sim \mathcal{N}(0, \tau^2)$ represents the intensity of attack.

• **Gaussian attacks**: The messages sent by Byzantine client k follow a Gaussian distribution, which can be formulated as $\tilde{\mathbf{w}}_k \sim \mathcal{N}(\mathbf{0}, \tau^2\mathbf{I})$.

Fairness in Federated Learning Fairness in the context of FL is a multifaceted issue that encompasses performance fairness, collaboration fairness, and model fairness (Zhou et al. 2021). This paper focuses on performance fairness, which aims to ensure that the model performs well across diverse client datasets, thereby preventing any client with less common data from being disadvantaged. The concept of fairness is influenced by the work of Li et al. (2021b), which advocates for a fair model to provide an equitable distribution of performance across all clients:

Definition 2 (Performance Fairness). In the case of a heterogeneous federated network, if model w_1 achieves a lower standard deviation (std) of test performance across N clients than model w_2, i.e., std $\{F_k(w_1)\}_{k\in[N]} <$ std $\{F_k(w_2)\}_{k\in[N]}$, where $F_k(\cdot)$ represents the test loss of client $k \in [N]$, then we say that model w_1 is more fair than model w_2.

Prior research has proposed reweighting techniques to address fairness, adjusting the contribution of client updates based on their performance (Li et al. 2020; Ezzeldin et al. 2023; Li et al. 2021a). However, these methods may come at the cost of mean test accuracy and may not be robust against adversarial attacks (Chen, Su, and Xu 2017; Blanchard et al. 2017; Xie, Koyejo, and Gupta 2018; Mohri, Sivek, and Suresh 2019; Hu et al. 2022).

To provide robustness and fairness, several frameworks aim to tune the deviations between local and global models finely. The Ditto framework by Li et al. (2021b) address this by allowing controlled deviations from the global model to foster personalization, which contributes to fair and robust outcomes across diverse client datasets. Similarly, Lin et al. (2022) propose a projection method that manages these deviations by embedding local models within a shared low-dimensional subspace, thus enhancing communication efficiency while ensuring robustness against adversarial attacks and fairness in resource allocation. Building on these foundations, our approach integrates deep reinforcement learning (DRL) to dynamically optimize the aggregation process, focusing on achieving a robust and fair global model that adapts to real-time network changes, thus further personalizing client contributions effectively.

Deep Reinforcement Learning. The integration of DRL into FL represents a promising frontier in addressing the challenges faced by FL. Notable examples include FAVOR (Wang et al. 2020), which utilizes a deep Q-learning (DQN) algorithm for client selection (Mnih et al. 2015), and FedRL (Zhuo et al. 2019), which operates within a discrete action space. Yet, these approaches are often limited by their reliance on discrete action spaces and greedy policies, which may not be suitable for the complex, continuous action spaces inherent in FL. This paper explores the application of DRL, specifically the DDPG algorithm (Lillicrap et al. 2016), to handle continuous control in FL, offering a more nuanced approach to client aggregation.

Methodology

In this section, we first outline the foundational concepts of FL and DRL. Then we formulate the optimized function and provide the details of the proposed algorithm.

Preliminary

DRL embodies a learning paradigm in which an agent learns to interact with its environment through a process of trial and error. In the context of DRL, at each timestep t, the agent perceives the current state $s(t)$ of the environment, selects an action $a(t)$, and subsequently receives a reward $r(t)$. This interaction leads to a transition to the next state $s(t+1)$. The overarching goal of DRL is to identify a policy that maximizes the cumulative discounted reward, defined as $R = \sum_{t=1}^{T} \gamma^{t-1} r(t)$, where γ, the discount factor, is a value within the range (0, 1].

Our work introduces a DRL framework based on the Deep Deterministic Policy Gradient (DDPG) algorithm (Lillicrap et al. 2016), which extends the capabilities of traditional Q-learning methods to handle continuous action spaces. The DDPG algorithm operates using an actor-critic structure (Konda and Tsitsiklis 1999), comprising two neural networks: an actor network $\pi(s|\theta^\pi)$ that selects actions, and a critic network $Q(s, a|\theta^Q)$ that evaluates the chosen actions. These networks are parameterized by θ^π and θ^Q, respectively. To improve stability and performance, we implement experience replay, target actor network $\pi'(s|\theta^{\pi'})$, and the target critic network $Q'(s, a|\theta^{Q'})$, which facilitate more consistent learning updates.

The actor network aims to develop a policy that maximizes the expected return $J = \mathbb{E}_{r_i,s_i,a_i}[R]$ from the environment's start distribution. This policy refinement is achieved through gradient ascent, utilizing the gradient

$$\nabla_{\theta^\pi} J \approx \frac{1}{N_d} \sum_i \nabla_a Q\big(s(i), \pi(s(i))|\theta^Q\big) \nabla_{\theta^\pi} \pi(s(i)|\theta^\pi), \tag{1}$$

where N_d is the batch size. The critic network's role is to approximate the action-value function $Q(s, a|\theta^Q)$, which predicts the expected return for a given state-action pair. The critic's learning process involves minimizing the loss function:

$$L = \frac{1}{N_d} \sum_i \big(y(i) - Q(s(i), a(i)|\theta^Q)\big)^2, \tag{2}$$

with $y(i)$ defined as the target value,

$$y(i) = r(i) + \gamma Q'\big(s(i+1), \pi'(s(i+1)|\theta^{\pi'})|\theta^{Q'}\big). \tag{3}$$

The target networks are periodically updated using the soft update equation $\theta' \leftarrow \varepsilon\theta + (1-\varepsilon)\theta'$, where ε is a small positive constant (Lillicrap et al. 2016), ensuring that the target networks slowly track the primary networks and provide a stable target for the learning process.

FedAA Overview

We present Federated Adaptive Aggregation (FedAA), a framework designed to enhance server-level robustness and fairness in federated environments. Our approach streamlines the aggregation process by integrating a novel client selection algorithm that identifies the top M% of clients based on model parameter proximity, thus protecting the aggregation against adversarial influences. The server, acting as a DDPG-driven agent, leverages this selection to determine the aggregation weights for these clients, guiding the global model update. FedAA's flexibility accommodates both full and partial client participation scenarios, enhancing its applicability in diverse federated learning contexts (Table 2).

Our methodology is designed to tackle the dual goals of optimizing global models while ensuring robustness and fairness on the server-side. We articulate this through a bi-level optimization that minimizes the aggregated local client objective while maximizing the aggregated model's accuracy on a fair validation set (see **Reward**). This is formalized as:

$$\max_{w_g} \ F_g(w_g) \coloneqq \text{Acc}(w_g, \mathcal{D}_g), \tag{4}$$

where $w_g = \text{argmin}_w \ G(F_1(w), ...F_N(w))$, F_g denotes the global optimization problem, $\text{Acc}(w_g, \mathcal{D}_g)$ means the global model w_g test accuracy on the dataset $\mathcal{D}_g$, $G(\cdot)$ represents the aggregation function at the server side, N denotes the number of clients, F_k is the local optimization problem for client k, i.e., $F_k \coloneqq \mathbb{E}_{x_k}[f(w, x_k)]$, x_k is a random sample draw according to the distribution of client k, $f(w, x_k)$ is the local loss function. In this context, $w_g = \sum_{k=1}^N a_k \cdot w_k$, $a_k \geq 0$ represents the aggregate weight for client k, $\sum_{k=1}^N a_k = 1$, and w_k is the optimized local model of client k.

The reward, defined by the model's accuracy on a fair dataset $\mathcal{D}_g$, incentivizes the agent to pursue actions that lead to a balanced and robust global model.

State. In the application of DRL to FL, the most straightforward idea is to consider the parameters of each client's model as states and input them into the actor network. However, when the client's model is a neural network, the parameter size becomes exceedingly large, rendering this idea unfeasible.

Algorithm 1: Client Selection

Input: client models $w_{all} = [w_1, ..., w_N]$, an $N \times N$ distance matrix $\mathbf{C}$ filled with zeros, M%.
Output: state s, top-M% clients' model w_{top}.

```
Flatten parameters of each client model to w'_1, ..., w'_N
for i = 1 to N do
    for j = 1 to N do
        C_{i,j} = ||w'_i - w'_j||_2
    end for
end for
Summing the rows of matrix C and selecting the top-M% rows with the minimum sums.
The sum of selected M% rows forms a distance vector s = [d_m, ..., d_M] as the state.
```

Inspired by the idea of FABA (Xia et al. 2019), there is a strong likelihood that the Euclidean distance among the model parameters of benign clients is closer than the distance between the model parameters of benign clients and malicious clients. Based on this observation, we propose a novel method for client selection and apply it to the generation of states. The details are shown in Algorithm 1. Through this algorithm, we can simultaneously obtain state $s = [d_m, \ldots, d_M]$ and top-M% clients' model $w_{top} = [w_m, ..., w_M]$, d_m is the sum of the distances between the client model m and the remaining client model parameters. These clients are the M% clients whose model parameters have the minimum total distance from the model parameters of all other clients. Notably, we perform normalization on the state s to get a stable training process. Further, the reason we do not directly use the distance matrix $\mathbf{C}$ as input is that, if we consider 100 clients, the state will be in a 10,000-dimensional space, which would significantly increase computational costs.

Action. Compared to the FAVOR (Wang et al. 2020), which constrains its action space to discrete numbers ranging from 1 to N, representing the IDs of selected clients, our proposed DDPG-based algorithm enjoys a continuous control feature. The actor network will generate a continuous action $a = [a_m, ..., a_M], \sum_i a_i = 1$ based on the input state s. In our algorithm, we set the action as the aggregation weights of selected top-M% clients.

Reward. When the server aggregates the parameters of selected M% clients, the server will get a reward r as feedback. The actor network performs gradient ascent to optimize its action for a higher expected cumulative reward. Hence, the design of the reward will guide the optimization direction of the actor network. Building a small dataset on

the server side has many applications in both academia (Valadi et al. 2023; Tan et al. 2022; Cao et al. 2021; Sandeepa et al. 2024; Zhao et al. 2022; Fang et al. 2020) and industry (McMahan and Ramage 2017). For example, Google can allow its employees to use Gboard to obtain server-side server datasets for next-word prediction (McMahan and Ramage 2017). For image recognition tasks like identifying cats and dogs, a group of people can be hired to label the cat and dog images. In the case of this paper, only a small dataset (e.g., 100 to 1000 training samples) is needed, and the service provider can usually afford the cost of manual collection and labeling. This dataset then can be used for the evaluation of the global model on various performance metrics. Here, we construct a fair held-out validation set at the server and use the test accuracy of the aggregated global model w_g on this validation set as the reward.

We adopt this approach for several reasons: testing at the server does not incur additional communication overhead, and it allows training an unbiased global model. Specifically, taking the example of MNIST, we construct a validation set at the server with 100 images for each digit, totaling 1000 images. Achieving higher rewards on such a validation set incentivizes the agent to make actions that are more fair to each client, unlike FedAvg, which assigns higher weights to clients with more images, which may lead to significant unfairness in a non-identically distributed (non-IID) setting. Conversely, if our constructed dataset disproportionately features a high quantity of digits 0 and 1, while scantily representing other digits, in pursuit of obtaining higher rewards, the agent might be incited to discern which clients' datasets are richer in these particular digits, subsequently elevating their aggregation weights. Such a scenario inadvertently precipitates a disparity detrimental to the remaining clients (Table 4).

Algorithm

In this section, we provide a comprehensive overview of the DDPG-based training process. The optimization of FedAA is composed of two components: (i) within the DRL workflow, the agent updates its actor and target networks; (ii) clients solve their local problems. The details are shown in Algorithm 2.

In this context, we adopt DRL to acquire an aggregation function to trade off robustness and fairness. After initialization, the FedAA algorithm progresses through sequential steps. At each step t, the server (acting as an agent) observes the current state $s(t)$, derived through the process of **Client Selection**. It then makes a deterministic action and performs aggregation, thereby generating a new global model $w_g(t)$. Then, an evaluation of the fair held-out dataset serves as the reward $r(t)$. The server then broadcasts the updated global model to all clients, who then solve local subproblems for R rounds. Following this training phase, a new round of **Client Selection** takes place to obtain the next state $s(t+1)$ for the subsequent iteration. The acquired transition $(s(t), a(t), r(t), s(t+1))$ will be preserved in the replay buffer for subsequent network updates. The actor and critic networks update at every step t, while the target actor and critic networks update once every two steps (soft update).

Algorithm 2: FedAA: Fair and Robust Federated Learning with Adaptive Aggregation

Input: $w_g(0)$, $\pi\left(s|\theta^{\pi}\right)$, $Q\left(s, a|\theta^{Q}\right)$, $\theta^{\pi'}$, $\theta^{Q'}$, $\mathcal{U}$, T, N, R.

Server(Agent) sends global model $w_g(0)$ to all clients.
$s(0), w_{top}(0) \leftarrow$ ClientSelection$(w_{all}, \mathbf{C}, \mathrm{M})$
for $t = 0$ **to** $T - 1$ **do**
  Server observes the state $s(t)$, and makes action $a(t) = \pi\left(s(t)|\theta^{\pi}\right) + \mathcal{N}$ ($\mathcal{N}$ is an exploration noise).
  Update global model $w_g(t) \leftarrow \sum_i a_i(t) \cdot w_{top_i}(t)$.
  $r(t) \leftarrow$Evaluation$(w_g(t))$.
  Server sends $w_g(t)$ to all N clients.
  for $k = 1$ **to** N **do**
    Client k solves its local problem for R rounds.
  end for
  $s(t+1), w_{top}(t+1) \leftarrow$ ClientSelection $(w_{all}, \mathbf{C}, \mathrm{M})$
  Store $(s(t), a(t), r(t), s(t+1))$ in the experience replay buffer $\mathcal{U}$.
  Sample a batch of experience from $\mathcal{U}$ to update θ^{π}, θ^{Q}, using Equation (1) and Equation (2).
  Soft update $\theta^{\pi'}, \theta^{Q'}$ via $\theta' \leftarrow \varepsilon\theta + (1-\varepsilon)\theta'$.
end for

The slow-updating target networks provide a stable learning process (Lillicrap et al. 2016).

Trade-off Between Robustness and Fairness. In previous works (Li et al. 2021b; Lin et al. 2022), the local model is susceptible to collapse under strong attacks (such as sign flipping). This can be explained by the insufficient robustness of the global model. In contrast, our proposed FedAA can simultaneously offer robustness and fairness at the server level. Specifically, when there is a certain fraction α ($\alpha < 0.5$) of malicious clients present, we can control the percentage $\mathrm{M}\%$ of clients participating in aggregation to trade off robustness and fairness. The intuition is that: when we adopt a larger value of M, it also increases the risk of introducing malicious clients during aggregation. However, simultaneously, introducing more client parameters in non-IID situations can enhance the generalization capability of the global model, implicitly promoting fairness across the clients.

Selections of State, Action, and Reward. Note that, within DRL, the design of **states**, **actions**, and **rewards** is highly personalized and not standardized. It can differ depending on the specific objectives of different algorithms, allowing for various setups of states, actions, and rewards. The algorithm proposed in this paper, based on DDPG, only presents a framework capable of continuous control within the context of FL. There is considerable potential for further exploration and development in subsequent work.

Numerical Experiments

In this section, we provide representative evaluation results to demonstrate that FedAA can achieve superior test accuracy, robustness, and comparable fairness compared to SOTA methods. We summarize the datasets, models, and other configurations used in this paper in Appendix A.1, and

full results in Appendix A.2. We compare FedAA with two SOTA approaches in robustness and fairness, namely Ditto (Li et al. 2021b), lp-proj (Lin et al. 2022), and a baseline FedAvg (McMahan et al. 2017) which are summarised in Appendix A.

We then exhibit the tradeoff capability between the robustness and fairness of FedAA. Next, we conduct supplementary experiments regarding the reward design, actual execution time, and partial participation to illustrate the feasibility of FedAA.

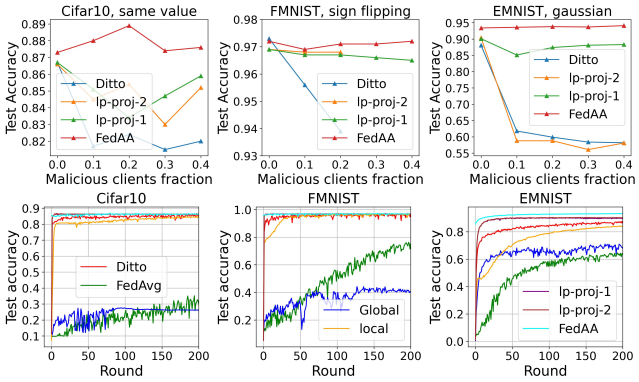

Figure 1: The figures in the first line represent robustness performance (i.e. mean test accuracy across benign clients) of three different datasets subjected to three different attacks. The figures in the second line depict the performance of three different datasets with no malicious clients.

Robustness and fairness. Following the definition of robustness (Li et al. 2021b), we provide empirical results on three different datasets, under three different attacks: same-value attack, sign-flipping attack, and Gaussian attack, with the parameter τ set to $\{100, 10, 100\}$ respectively. In the absence of malicious clients, Figure 1 demonstrates that FedAA achieves performance similar to SOTA methods on CIFAR10 and EMNIST datasets. However, it shows a slight inferiority compared to Ditto (Li et al. 2021b) on FASHION-MNIST with no adversaries. Furthermore, under the same value attack, the performance of Ditto and lp-proj (Lin et al. 2022) slightly decreases with the increasing number of malicious clients. While FedAA shows a slight improvement. Additionally, under two other types of attacks, Ditto and lp-proj-2 perform poorly. Specifically, subjected to Gaussian attacks, both algorithms exhibit a notable deterioration in performance. In the case of a strong attack, i.e. sign flipping, Ditto and lp-proj-2 collapse when the fraction of malicious clients exceeds 0.2.

The tradeoff between test accuracy and variance for different baselines is illustrated in Figure 2. We have examined two scenarios: one involving a malicious client and one without. As shown in Figure 2, FedAA achieves superior accuracy. However, its variance is marginally higher compared to other approaches. This can be mitigated by tuning M which will be discussed later.

Numerous experimental results demonstrate that FedAA can provide robustness and fairness, and due to the space

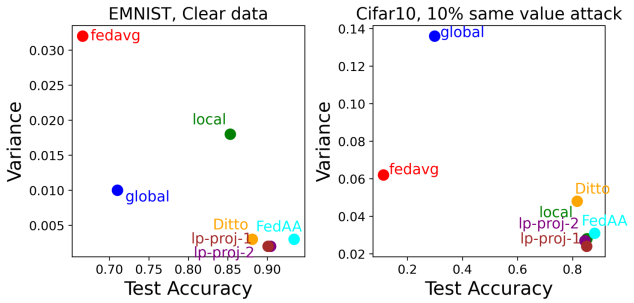

Figure 2: The tradeoff between test accuracy and fairness within different methods. The closer the approach is to the lower right corner, the better.

limitation, we show full results in Appendix A.2. Further, we present some analysis of the underlying reasons. As mentioned in Ditto (Li et al. 2021b), superior results can be achieved through the execution of local fine-tuning for 50 epochs on the global model after specific communication rounds. However, determining the optimal 'point' for early stopping during training poses a challenge, especially in the presence of a fraction of malicious clients corrupting the global model. In contrast, FedAA can constantly provide a relatively robust global model through a robust client selection algorithm. Meanwhile, the server, also referred to as the agent, aims to maximize the expected accumulative reward. It optimizes the aggregation weights in each round and learns a policy to make better decisions. In non-IID scenarios, an aggregation function that involves interaction with clients and incorporates feedback for continuous learning demonstrates superior performance compared to FedAvg (McMahan et al. 2017), which determines aggregation weights simply based on the sample size in each round.

Tradeoff between robustness and fairness in FedAA. Results are shown in Figure 3. Experiments are conducted on CIFAR10, while in the presence of different percentages of malicious clients, all subjected to the same value attack. We set the number of clients participating in aggregation, M, ranging from 10% to 100%. Specifically, there exists a certain threshold, i.e. if there are 100 clients with 20% being malicious clients, then the threshold is set at 80. We see that, under different percentages of malicious clients, there is a certain pattern in the changes of test accuracy and variance. Specifically, as the value of M continuously increases, test accuracy initially oscillates upward, reaching its maximum at the threshold, and then experiences a sharp decline. In contrast, variance exhibits the opposite pattern, with a continuous increase in M leading to initial oscillations downward, reaching its minimum at the threshold, and then rapidly rising. Taking the example of 20% malicious client, under the same value attack, $\mathrm{M} = 80\%$ achieves the highest mean test accuracy of 90.3% and the lowest variance of 0.013 (complete results are available in Appendix A.2). The reason for not optimizing M stems from the associated risks. Consider a scenario in which malicious clients collaborate (see (Xie, Koyejo, and Gupta 2020)). These clients might

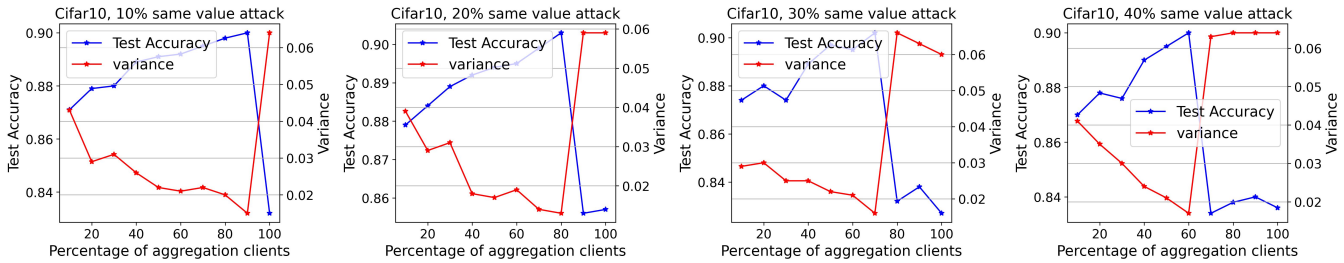

Figure 3: The performance and tradeoff between robustness and fairness of different M (The numbers on the x-axis in the figures represent the corresponding M%, e.g. 80 means M = 80%).

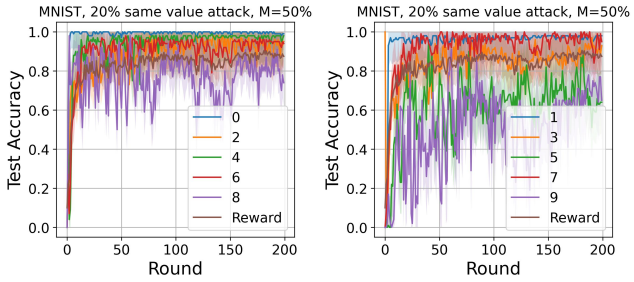

Figure 4: The convergence curve of reward r of each class at the server.

submit regular updates for t epochs, during which time M could be optimized to an excessively high value, potentially reaching 100%. Should these malicious clients then collectively submit anomalous updates, they could easily corrupt the entire FL system.

The experimental results validate the initial proposition of FedAA, indicating that we can provide a tradeoff between robustness and fairness by controlling the number of participating clients M in aggregation. To elaborate further, when the server encounters an attack at first, it may receive lower rewards. In subsequent network updates, the server learns to assign lower aggregation weights to clients suspected of being malicious, thereby enhancing robustness. Additionally, in order to attain higher rewards, the server also learns how to allocate weights among benign clients to ensure fairness.

Reward design. Figure 4 illustrates the effectiveness of carefully designed rewards. The figure depicts the convergence process in test accuracy for each digit. To enhance clarity, we separate odd and even digits into two separate figures. From Figure 4, it is evident that most digits converge to similar ranges, except for digits 5 and 9, which converge around 65%. These results are deemed fair for the majority of clients. However, it presents a relative unfairness for clients primarily composed of digits 5 and 9. This issue can be mitigated by refining the design of rewards in future work.

Compression methods and excution time. We compared two compression methods along with their corresponding actual execution times. One is reducing the number of neurons in the hidden layers of the DDPG network to 256, 128, and 64, respectively. The other method entailed selecting only the parameters of the last hidden layer (LHL) of the client model, instead of all layers (AL).

DATASET	METHODS	RUNTIME	ACC
MNIST	FEDAA(AL 256)	5,173S	0.978(0.001)
	FEDAA(AL 128)	5,127S	**0.979(0.001)**
	FEDAA(AL 64)	5,103S	0.978(0.000)
FASHIONMNIST	FEDAA(AL 256)	19,872S	**0.975(0.038)**
	FEDAA(LHL)	18,314S	0.974(0.038)
CIFAR10	FEDAA(AL 256)	14,392S	**0.875(0.024)**
	DITTO	13,096S	0.820(0.042)

Table 1: Compression methods and actual execution time.

In Table 1, AL 256 indicates uploading all parameters of the client model, with the hidden layer neuron count in the DDPG network set to 256. As can be seen in Table 1, the impact of different hidden layer dimensions in the actor and critic networks within DRL on test accuracy is limited. We delve further into applying another compression technique and measure the real-world execution time of FedAA, as is elaborated in Table 1. For clarification, the term 'last hidden layer' specifically refers to utilizing the parameters of the model's final hidden layer as input for the algorithm described in Algorithm 1 (Li, Sun, and Zheng 2022). The employed compression strategy is effective, yielding robust and competitive results. Although FedAA exhibits a slightly increased operational time in comparison to Ditto, the enhanced performance by 5.5% justifies the additional duration, which is deemed manageable.

IPM attack and partial participation. To enhance the assessment of the proposed FedAA, we incorporate a more potent adversarial attack method known as Inner Product Manipulation (IPM) (Xie, Koyejo, and Gupta 2020), which is crafted specifically to target Krum-based aggregation schemes. In Table 2, C=100% signifies the participation of all clients in the aggregation phase, whereas C=50% indicates that only a subset, specifically half, of the clients are involved in the process, characteristic of a partial participation FL framework.

Our findings demonstrate that the IPM presents a formidable challenge to the FedAvg method, though its impact is mitigated against more sophisticated defensive strate-

Dataset	Methods	clear	IPM	
			10%	20%
CIFAR10	FedAvg	0.377(0.048)	0.105(0.045)	0.114(0.039)
	Ditto	0.867(0.019)	0.857(0.022)	0.849(0.023)
	lp-proj-1	0.867(0.021)	0.864(0.029)	0.857(0.031)
	FedAA(C=100% M=30%)	0.873(0.015)	**0.870(0.017)**	0.861(0.018)
	FedAA(C=50% M=30%)	**0.878(0.013)**	0.869(0.016)	**0.862(0.018)**
MNIST	FedAvg	0.904(0.014)	0.425(0.006)	0.415(0.006)
	Ditto	0.972(0.005)	0.937(0.011)	0.933(0.012)
	lp-proj-1	0.971(0.007)	0.969(0.008)	0.968(0.008)
	FedAA(C=100% M=30%)	0.977(0.002)	**0.977(0.002)**	0.975(0.004)
	FedAA(C=50% M=30%)	**0.984(0.001)**	0.976(0.002)	**0.977(0.003)**

Table 2: Comparison of FedAA and baseline methods under inner product manipulation (IPM) (Xie, Koyejo, and Gupta 2020) attack, where C indicates client participation ratio, across CIFAR-10 and MNIST datasets.

gies. In every examined scenario, the FedAA consistently outpaces the benchmark models. Furthermore, the FedAA exhibited remarkable adaptability to scenarios with only partial client participation in the aggregation phase, and in certain cases, it even delivered superior performance.

Method	Datasets		
	CIFAR-100	Tiny-iamgenet	AGnews-100
FedAA	**0.524(0.006)**	**0.260(0.021)**	**0.955(0.126)**
Ditto	0.239(0.007)	0.203(0.029)	0.950(0.047)
lp-proj-1	0.478(0.000)	0.224(0.000)	0.946(0.038)
lp-proj-2	0.470(0.000)	0.219(0.000)	0.946(0.039)
fedavg	0.019(0.000)	0.095(0.000)	0.312(0.142)

Table 3: Comparative analysis of three more challenging datasets.

Challenging datasets. To fully explore the capabilities of FedAA, we introduce three additional challenging datasets. The first, CIFAR-100 (Krizhevsky, Hinton et al. 2009), comprises 60,000 color images evenly distributed across 100 categories. The second, Tiny-ImageNet (Le and Yang 2015), is a compact version of the ImageNet dataset, configured with 200 classes. Lastly, AG-News (Zhang, Zhao, and LeCun 2015), offers a corpus exceeding one million news articles categorized into four distinctive sections. Each serves to benchmark text classification and machine learning models adept in natural language processing. As demonstrated in Table 3, FedAA surpasses Ditto's performance by approximately 30%, 6%, and 0.5% on these datasets, respectively.

In Table 4, we conduct comparative experiments to more effectively assess the performance of the held-out validation set. Starting with an example from Cifar-10, we modify the size of the set from 10 to 100 images for each category. In a specific scenario involving an unfair dataset, which includes 100 images each for categories 'airplane' and 'automobile' (labels 0 and 1, respectively), and only 10 images for all other categories. It is noted that an increase in the size of the dataset led to a modest improvement in accuracy. Crucially, the variance observed in this unfair dataset scenario is markedly higher compared to that in the other three scenarios.

Datasets	Size	Accuracy
CIFAR10	10	0.842(0.023)
	50	0.854(0.015)
	100	**0.875(0.024)**
	unfair	0.855(0.043)
MNIST	10	0.976(0.001)
	50	0.976(0.001)
	100	**0.978(0.001)**
	unfair	0.978(0.001)
FASHIONMNIST	10	0.972(0.025)
	50	0.974(0.022)
	100	**0.975(0.038)**
	unfair	0.975(0.047)

Table 4: Comparative analysis of different size of held-out validation datasets.

Conclusion and Discussion

In this paper, we model each communication round in the FL as an MDP and propose a simple framework FedAA, that seamlessly integrates DDPG into distributed learning with the capability of continuous action control. In addition, we reveal the tension between robustness and fairness at the server level. FedAA can simultaneously deliver superior performance, robustness, and comparable fairness. To attain this objective, we propose a novel client selection algorithm and offer the tradeoff by regulating the number M of clients participating in the aggregation.

In future work, the integration of DRL into frameworks similar to Ditto and lp-proj could yield more robust models. Furthermore, through careful design of states, actions, and rewards, DRL can be applied to problems that are challenging for traditional approaches to handle. In addition, DRL may require a substantial number of transitions for training to unleash its optimal performance. Therefore, future endeavors could explore the pre-training of a robust DRL model capable of accurately identifying malicious clients, assigning them lower aggregation weights. Simultaneously, by leveraging the distance relationships between models, adaptive weight allocation can be achieved to yield fairer results. Therefore, the trade-off between the resulting overhead and performance enhancement is a noteworthy consideration for further contemplation.

Acknowledgments

This research was supported by National Key Research and Development Program of China under Grant 2022ZD0115301.

References

Biggio, B.; Nelson, B.; and Laskov, P. 2012. Poisoning Attacks against Support Vector Machines. In *Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Edinburgh, Scotland, UK, June 26 - July 1, 2012*.

Blanchard, P.; El Mhamdi, E. M.; Guerraoui, R.; and Stainer, J. 2017. Machine learning with adversaries: Byzantine tolerant gradient descent. *Advances in neural information processing systems*, 30.

Cao, X.; Fang, M.; Liu, J.; and Gong, N. Z. 2021. FLTrust: Byzantine-robust Federated Learning via Trust Bootstrapping. In *28th Annual Network and Distributed System Security Symposium, NDSS 2021, virtually, February 21-25, 2021*. The Internet Society.

Chen, Y.; Su, L.; and Xu, J. 2017. Distributed statistical machine learning in adversarial settings: Byzantine gradient descent. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 1(2): 1–25.

Cheng, Y.; Liu, Y.; Chen, T.; and Yang, Q. 2020. Federated learning for privacy-preserving AI. *Communications of the ACM*, 63(12): 33–36.

Dai, Y.; and Li, S. 2023. Chameleon: Adapting to Peer Images for Planting Durable Backdoors in Federated Learning. In *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, 6712–6725. PMLR.

Ezzeldin, Y. H.; Yan, S.; He, C.; Ferrara, E.; and Avestimehr, A. S. 2023. Fairfed: Enabling group fairness in federated learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, 7494–7502.

Fang, M.; Cao, X.; Jia, J.; and Gong, N. 2020. Local model poisoning attacks to {Byzantine-Robust} federated learning. In *29th USENIX security symposium (USENIX Security 20)*, 1605–1622.

Hu, Z.; Shaloudegi, K.; Zhang, G.; and Yu, Y. 2022. Federated learning meets multi-objective optimization. *IEEE Transactions on Network Science and Engineering*, 9(4): 2039–2051.

Jagielski, M.; Oprea, A.; Biggio, B.; Liu, C.; Nita-Rotaru, C.; and Li, B. 2018. Manipulating machine learning: Poisoning attacks and countermeasures for regression learning. In *2018 IEEE symposium on security and privacy (SP)*, 19–35. IEEE.

Kaissis, G. A.; Makowski, M. R.; Rückert, D.; and Braren, R. F. 2020. Secure, privacy-preserving and federated machine learning in medical imaging. *Nature Machine Intelligence*, 2(6): 305–311.

Konda, V.; and Tsitsiklis, J. 1999. Actor-critic algorithms. *Advances in neural information processing systems*, 12.

Krizhevsky, A.; Hinton, G.; et al. 2009. Learning multiple layers of features from tiny images.

Kulkarni, V.; Kulkarni, M.; and Pant, A. 2020. Survey of personalization techniques for federated learning. In *2020 Fourth World Conference on Smart Trends in Systems, Security and Sustainability (WorldS4)*, 794–797. IEEE.

Le, Y.; and Yang, X. 2015. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7): 3.

Li, B.; Wang, Y.; Singh, A.; and Vorobeychik, Y. 2016. Data poisoning attacks on factorization-based collaborative filtering. *Advances in neural information processing systems*, 29.

Li, H.; Sun, X.; and Zheng, Z. 2022. Learning to attack federated learning: A model-based reinforcement learning attack framework. *Advances in Neural Information Processing Systems*, 35: 35007–35020.

Li, T.; Beirami, A.; Sanjabi, M.; and Smith, V. 2021a. Tilted Empirical Risk Minimization. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*.

Li, T.; Hu, S.; Beirami, A.; and Smith, V. 2021b. Ditto: Fair and Robust Federated Learning Through Personalization. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, 6357–6368. PMLR.

Li, T.; Sanjabi, M.; Beirami, A.; and Smith, V. 2020. Fair Resource Allocation in Federated Learning. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*.

Lillicrap, T. P.; Hunt, J. J.; Pritzel, A.; Heess, N.; Erez, T.; Tassa, Y.; Silver, D.; and Wierstra, D. 2016. Continuous control with deep reinforcement learning. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Lin, S.; Han, Y.; Li, X.; and Zhang, Z. 2022. Personalized federated learning towards communication efficiency, robustness and fairness. *Advances in Neural Information Processing Systems*, 35: 30471–30485.

McMahan, B.; Moore, E.; Ramage, D.; Hampson, S.; and y Arcas, B. A. 2017. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, 1273–1282. PMLR.

McMahan, B.; and Ramage, D. 2017. Federated Learning: Collaborative Machine Learning without Centralized Training Data.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A. A.; Veness, J.; Bellemare, M. G.; Graves, A.; Riedmiller, M.; Fidjeland, A. K.; Ostrovski, G.; et al. 2015. Human-level control through deep reinforcement learning. *nature*, 518(7540): 529–533.

Mohri, M.; Sivek, G.; and Suresh, A. T. 2019. Agnostic federated learning. In *International Conference on Machine Learning*, 4615–4625. PMLR.

Rubinstein, B. I.; Nelson, B.; Huang, L.; Joseph, A. D.; Lau, S.-h.; Rao, S.; Taft, N.; and Tygar, J. D. 2009. Antidote:

understanding and defending against poisoning of anomaly detectors. In *Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement*, 1–14.

Sandeepa, C.; Siniarski, B.; Wang, S.; and Liyanage, M. 2024. SHERPA: Explainable Robust Algorithms for Privacy-Preserved Federated Learning in Future Networks to Defend Against Data Poisoning Attacks. In *2024 IEEE Symposium on Security and Privacy (SP)*, 204–204. IEEE Computer Society.

Suciu, O.; Marginean, R.; Kaya, Y.; Daume III, H.; and Dumitras, T. 2018. When does machine learning {FAIL}? generalized transferability for evasion and poisoning attacks. In *27th USENIX Security Symposium (USENIX Security 18)*, 1299–1316.

Tan, A. Z.; Yu, H.; Cui, L.; and Yang, Q. 2022. Towards personalized federated learning. *IEEE Transactions on Neural Networks and Learning Systems*.

Valadi, V.; Qiu, X.; De Gusmão, P. P. B.; Lane, N. D.; and Alibeigi, M. 2023. {FedVal}: Different good or different bad in federated learning. In *32nd USENIX Security Symposium (USENIX Security 23)*, 6365–6380.

Wang, H.; Kaplan, Z.; Niu, D.; and Li, B. 2020. Optimizing federated learning on non-iid data with reinforcement learning. In *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*, 1698–1707. IEEE.

Xia, Q.; Tao, Z.; Hao, Z.; and Li, Q. 2019. FABA: an algorithm for fast aggregation against byzantine attacks in distributed neural networks. In *IJCAI*.

Xie, C.; Koyejo, O.; and Gupta, I. 2018. Generalized Byzantine-tolerant SGD. *CoRR*, abs/1802.10116.

Xie, C.; Koyejo, O.; and Gupta, I. 2020. Fall of empires: Breaking byzantine-tolerant sgd by inner product manipulation. In *Uncertainty in Artificial Intelligence*, 261–270. PMLR.

Zhang, X.; Zhao, J.; and LeCun, Y. 2015. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28.

Zhao, B.; Sun, P.; Wang, T.; and Jiang, K. 2022. Fedinv: Byzantine-robust federated learning by inversing local model updates. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, 9171–9179.

Zhou, Z.; Chu, L.; Liu, C.; Wang, L.; Pei, J.; and Zhang, Y. 2021. Towards fair federated learning. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 4100–4101.

Zhuo, H. H.; Feng, W.; Lin, Y.; Xu, Q.; and Yang, Q. 2019. Federated deep reinforcement learning. *arXiv preprint arXiv:1901.08277*.