PS-FedGAN: An Efficient Federated Learning Framework with Strong Data Privacy

Achintha Wijesinghe, Songyang Zhang, *Member, IEEE* and Zhi Ding, *Fellow, IEEE*

***Abstract*—Federated Learning (FL) has emerged as an effective paradigm for distributed learning systems owing to its strong potential in exploiting underlying data characteristics while preserving data privacy. In cases of practical data heterogeneity among FL clients in many Internet-of-Things (IoT) applications over wireless networks, however, existing FL frameworks still face challenges in capturing the overall feature properties of local client data that often exhibit disparate distributions. One approach is to apply generative adversarial networks (GANs) in FL to address data heterogeneity by integrating GANs to regenerate anonymous training data without exposing original client data to possible eavesdropping. Despite some successes, existing GAN-based FL frameworks still incur high communication costs and elicit other privacy concerns, limiting their practical applications. To this end, this work proposes a novel FL framework that only applies partial GAN model sharing. This new PS-FedGAN framework effectively addresses heterogeneous data distributions across clients and strengthens privacy preservation at reduced communication costs, especially over wireless networks. Our analysis demonstrates the convergence and privacy benefits of the proposed PS-FEdGAN framework. Through experimental results based on several well-known benchmark datasets, our proposed PS-FedGAN demonstrates strong potential to tackle FL under heterogeneous (non-IID) client data distributions, while improving data privacy and lowering communication overhead.**

***Index Terms*—Federated Learning (FL), generative adversarial networks (GAN), heterogeneous users. machine learning, privacy.**

I. Introduction

FEderated Learning (FL) presents an exciting framework for distributed and collaborative learning while preserving user data privacy [1], [2]. As an emerging field of artificial intelligence, the full potential of FL must effectively address a myriad of challenges, including heterogeneity of data distribution, data privacy consideration, and communication efficiency, especially for edge devices in Internet-of-Things (IoT) systems. With the widespread deployment of IoT devices such as smart wearables, mobile devices, and personal digital assistants, copious data are generated every second, enabling collaborative machine learning for accurate utility [4]. Today, many global machine learning models such as classifiers readily benefit from ample user-generated data and are implemented on many platforms in IoT systems. On the other hand, continual (machine) learning for global model updates from access to new data also represents an important learning aspect in IoT systems. Due to bottlenecks such as communication efficiency and privacy of user data, direct application of standard machine learning techniques for continual updating of global models is not straightforward. As a solution, FL is widely applied in such IoT systems [5].

Despite successes, existing FL methods require more care to handle non-independent identically distributed (IID) heterogeneous data among different participating FL clients (users) instead of IID user data. The problem of handling data heterogeneity still remains an interesting research topic. Existing works mainly share common datasets, modified algorithms, or generative models to handle data heterogeneity. Among these FL schemes to handle data heterogeneity, generative adversarial networks (GANs) [6] based approaches have recently drawn substantial interest owing to their ability to regenerate data statistics without sharing raw data. In previous works, a GAN model is trained locally to capture the local user's data distribution. Subsequently, the locally trained GAN model is shared with a centralized server for model aggregation. The model aggregation can rely on generating more data to update a target global model or a customized GAN model for each user. Despite reported successes, GAN-based FL also exhibits several shortcomings, such as poor privacy protection and heavy communication redundancies, caused by full model or synthetic data sharing [7], [8].

Many studies, e.g., [9], [10], underscore the susceptibility of traditional FL methods to security risks, and reveal the potential risks of sensitive information leakage. These concerns extend to GAN-based FL, as highlighted in [11], where the release of GAN models or synthetic data raises serious privacy issues. Specifically, GANs face threats from reconstruction attacks and membership inference attacks, as detailed in [12], where adversaries seek to recreate data samples and ascertain the utilization of specific data samples, respectively. Efforts to address privacy issues in GAN-based FL have considered the exploration of differential privacy (DP) [13], as addressed in works such as [7], [14]. However, the adoption of DP highlights a significant challenge: balancing the tradeoff between privacy and utility. This specific challenge is exacerbated by the necessity to allocate a privacy budget, a parameter crucial for balancing the performance of the learning model versus privacy preservation. Paradoxically, to enhance the performance of downstream tasks, practitioners often opt for an infinite privacy budget, effectively nullifying any privacy

 This material is based upon work supported by the National Science Foundation under Grants 2029027, 2029848, and 2332760.

A. Wijesinghe and Z. Ding are with the Department of Electrical and Computer Engineering, University of California, Davis, CA, 95616. (E-mail: achwijesinghe@ucdavis.edu, and zding@ucdavis.edu).

S. Zhang is with the Department of Electrical and Computer Engineering, University of Louisiana at Lafayette, Lafayette, LA, 70504. (E-mail: songyang.zhang@louisiana.edu).

safeguards.

Moreover, the communication link efficiency of GAN-based FL is impeded by the substantial size of the shared models and the limited communication resources in various real-world networking scenarios. This communication inefficiency poses a practical obstacle, limiting the seamless exchange of model updates and the collaborative learning process in federated environments. Consequently, addressing both privacy concerns and communication efficiency emerges as a pivotal challenge in advancing the robustness and applicability of GAN-based FL methods.

To summarize, the existing GAN-related FL suffers from heavy communication overhead and severe privacy leakage, due to the full GAN sharing. On the other hand, existing privacy-preserving approaches may result in utility degradation when protecting data security. To this end, we re-examine the GAN sharing strategy in FL and propose a novel GAN publishing mechanism, named **PS-FedGAN** (**P**artially **S**hared **Fed**erated **GAN**), to address practical cases of non-IID data heterogeneity among FL client users. Through our proposed FL framework, we reconstruct separate generators at the server from partially shared GAN models trained locally at client users, in which each client only shares its discriminator with the server. The proposed PS-FedGAN significantly reduces the communication network overhead of model sharing and provides better data privacy during communication rounds. Furthermore, it bridges the gap between utility and privacy. Fig. 1 highlights the distinction between existing full GAN-sharing approaches and our proposed PS-FedGAN over an untrustworthy network. We summarise our contributions as follows:

- We propose **PS-FedGAN**, a novel GAN-based FL learning framework to cope with non-IID data among FL client users. More specially, we train a generator at the server to capture the underlying data distribution of local user GAN by only sharing an individual discriminator. The proposed framework can significantly lower communication costs and improve data privacy.
- We provide analytical insights into the convergence of generator training at the client end and at the cloud server. We investigate the convergence of the common discriminator training based on our proposed **PS-FedGAN** to establish the benefit of communication cost reduction by only sharing discriminators. To the best knowledge, we are the first to provide a systematic theoretical analysis of FL based on partially shared GAN.
- We present an interpretable result on the privacy of the proposed **PS-FedGAN**, which is further corroborated by our theoretical analysis and extensive experiments.
- Our experimental results against several well-known benchmark datasets further demonstrate the efficacy and efficiency of our **PS-FedGAN**, in terms of utility, privacy, and communication cost.

The subsequent sections of this paper are structured as follows. In Section II, we first provide an overview of the existing literature on Federated Learning (FL) and Generative Adversarial Network (GAN) models. Section III is dedicated to introducing our problem formulation and the details of the proposed novel partial GAN-sharing method. We analyze the theoretical aspects of PS-FedGAN in the section V. The section VI encompasses the presentation of experimental results. Finally, we present conclusions and future works in Section VII.

II. Related Work

This section introduces existing FL frameworks for IID and non-IID user distributions and also summarizes relevant details of GANs.

A. The Concept of Federated Learning

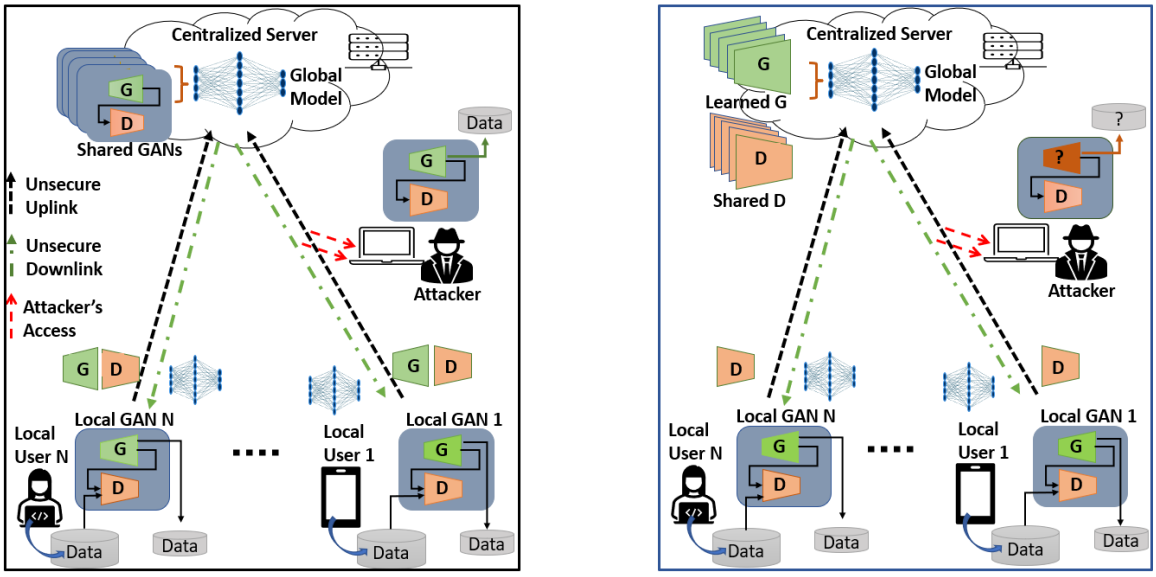
FL stands out as a straightforward and effective approach to privacy preservation in distributed learning scenarios. In the realm of basic FL, local client datasets are often heterogeneous, unbalanced, and non-IID (non-identically distributed). The FL is over a network under constraints of limited communication bandwidth [1]. To mitigate communication overhead and preserve privacy, the Federated Average (FedAvg) algorithm emerges as a solution. FedAvg aggregates gradient updates from various users by consolidating multiple local updates in a single round of communication [2].

While the convergence of FedAvg for non-IID data has been studied [16], it has been revealed in [17] that FedAvg can incur an accuracy loss of up to $\sim 55\%$ in non-IID setups for certain datasets. Effectively addressing non-IID data heterogeneity among diverse clients remains an open FL question. FedProx [18] has been introduced as a generalized version of FedAvg against heterogeneous networks. Another extension of FedAvg is SCAFFOLD, which integrates predictive variance reduction techniques to enhance performance [19]. In [20], the authors propose a new approach involving classifier calibration with virtual representation, leveraging a Gaussian Mixer Model to sample virtual features.

Adopting an alternative strategy, the authors of [21] advocate for raw data sharing within a cooperative learning mechanism. Additionally, the suggestion in [16] is to incorporate a small fraction (e.g. 5%) of initial training data to accommodate non-IID data using traditional FL algorithms. Several other methods for dealing with non-IID data include GAN sharing [7], [22], [23], synthetic data sharing [24], [25], and global sub-dataset sharing [17]. These diverse approaches collectively contribute to the ongoing exploration of effective solutions for addressing the challenges posed by non-IID client data in the context of FL.

B. Federated Learning for IoT

FL may apply to a wide range of IoT services. These services range from IoT data sharing, data offloading, and caching to IoT privacy and attack detection [5]. Similar to FL in any paradigm, the main bottleneck of FL in IoT lies in the imbalanced and statistically heterogeneous user data [26] and the need to enhance system security. In [27], authors have addressed the cyber security aspect of FL in IoT. Another critical FL problem involves the resource-constrained IoT



(a) Classical FL with full GAN sharing.

(b) Proposed PS-FedGAN discriminator sharing

Fig. 1: Comparison of full GAN sharing v.s. our proposed PS-FedGAN discriminator sharing for N users. In both cases, a local user trains its GAN model with a generator (G) and a discriminator (D). In traditional FL with fully shared GAN, full GAN models are communicated (i.e. both G and D are shared with the server simultaneously.), whereas PS-FedGAN only shares D. In full GAN sharing, an attacker may eavesdrop over an unsecured channel to gain access to shared G and D. As a result of sharing G and D with the server by clients, the passive attacker can generate synthetic data which may approximate user data distribution. Furthermore, membership inference [15] could also become viable in this setup [12]. In contrast, PS-FedGAN prevents eavesdropping of G and denies attacker access to both true GAN model G and original data distributions.

devices [28]. On the other hand, GAN models can be used to attack IoT-based FL systems as discussed in [29], [30]. At the same time, GAN-based FL approaches are common in IoT applications to enhance performances and personalized solutions [23].

C. GAN-based FL for IID and Non-IID Clients

The practice of training Generative Adversarial Networks (GANs) locally before disseminating the trained models to a centralized server has gained momentum as an effective strategy for managing both IID and non-IID data. In the study by [31], a conditional GAN (cGAN) is employed, and both the local classifier and generator are shared with the central server. The central server trains a global classifier and generator, guiding the actions of local users. Expanding on this approach, the authors of [32] suggest incorporating model splitting by preserving a portion of the cGAN, specifically the discriminator, and a segment of the hidden classifier, while sharing the generator and a global classifier.

Similarly, the notion of sharing the entire local GAN with the server and generating a synthetic local dataset to train a global GAN is proposed in [7], [23]. This approach, known as full GAN sharing (FGS), has also been explored by [22], where only shared generators exhibiting the maximum mean discrepancy are aggregated. Although these methods show promise, effectively addressing privacy concerns in GAN-based FL remains an ongoing challenge. To this end, Differential Privacy (DP) [13] as a mitigating measure has been introduced in GAN-based FL, with applications shown in works such as [7], [14]. However, incorporating a privacy budget in DP introduces a delicate balance between privacy and utility. The pursuit for effective ways to reduce the size of shared models (i.e., communication cost) while preserving privacy in GAN-based FL persists as an open research challenge and warrants continued exploration and innovations.

D. Generative Adversarial Networks

Generative Adversarial Network (GAN) [6] features a minimax game between two neural networks, namely the generator (G) and the discriminator (D). The concept of GAN is to learn and capture sufficient information about a given data distribution to generate data samples that follow the original data distribution. In a GAN, G tries to learn a mapping from the noise space to the data space by starting with a random noise to generate an image to fool D. Whereas, D learns to distinguish samples from in/out of the data distribution. If the generator learns a distribution p_g over data $\mathbf{x}$ by mapping a prior noise distribution $p_z(\mathbf{z})$ to data space. The following objective function can be used to train the GAN model:

$$\min_G \max_D \quad V(D,G) = \mathbb{E}_{x\sim p_{\text{data}}(\mathbf{x})}[\log D(\mathbf{x})] + \mathbb{E}_{z\sim p_z(\mathbf{z})}[\log(1 - D(G(\mathbf{z})))], \qquad (1)$$

where $\mathbb{E}[\cdot]$ refers to statistical expectation.

The downside of traditional GAN is that it provides no control over the reconstruction, i.e., the generated output. To gain control over the generation, the conditional GAN model (cGAN) is introduced. If the generator and discriminator are

TABLE I: Key notations and abbreviations

Notation/ Abbreviation	Description
FL	Federated learning
DP	Differential Privacy
GAN	Generative Adversarial Network
G	Generator
D	Discriminator
G_u	User generator
D_u	User discriminator
G_s	Server generator
D_s	Server discriminator
Cl	Classification model
$\mathcal{M}$	User side publishing mechanism
$\mathcal{M}_p$	PS-FedGAN publishing mechanism
N	Number of updates
z	Noise vector
l	labels
$\mathcal{A}_R$	Reconstruction attacks
$\mathcal{A}_1$	Attacker type I
$\mathcal{A}_2$	Attacker type II
θ	Neural network parameters

conditioned on some extra information $\mathbf{I}$ [33]. Then, the objective function could be set as

$$\min_G \max_D \quad V(D,G) = \mathbb{E}_{x\sim p_{data}(\mathbf{x})}[\log D(\mathbf{x}|\mathbf{I})] + \mathbb{E}_{z\sim p_z(\mathbf{z})}[\log(1 - D(G(\mathbf{z}|\mathbf{I})))]. \quad (2)$$

In this work, we utilize cGAN models. Conditional GAN models provide the freedom of data (e.g. image) generation by conditioning the class label. As a result, there is no need to implement an auxiliary classifier to classify the generated output of cGAN models.

III. Problem Formulation

In this work, we aim to develop a novel GAN-based FL framework for a global/common task in a distributed learning setup. For convenience, we will use image classification as an illustrative example henceforth. Consider a reliable central server with limited access to client training data to achieve a desirable accuracy on the global task. Here, we assume non-IID data distributions and vulnerable communication links among users, a common scenario in practice. For example, in smart healthcare, one learning task could be to train neural networks for the detection of a specific disease. One clinic may have a unique set of brain images from a specific patient group whereas other hospitals may also have access to several types of similar diseases with a plethora of examples. A global model based on all distributed data to detect these diseases could benefit all participating hospitals and providing better service for patients without releasing actual patient data. In such a collaborative system, local data privacy is a critical concern. Also, adding artificial noises to the dataset to hide sensitive information may lead to unwanted information distortion and/or learning deficiencies. In this work, we address how to preserve local privacy while preserving the original data statistics.

Motivated by existing GAN-based FL, we develop a novel GAN sharing/publishing mechanism for FL with privacy preservation and communication efficiency. Consistent with existing GAN-related FL works, we assume a system with a centralized server in the cloud and multiple distributed clients/users in communication with the server. Each client has access to enough resources to train a local GAN model. Note that, although we apply conditional GANs (cGAN) [33] to alleviate the need for label detection via pseudo-labeling, the principle of our proposed scheme generally applies to all types of generative models.

To assess the efficacy of our framework in preserving privacy, we consider adversarial attackers that passively eavesdrop. In order to model potential attacks from these quiet adversaries, we assume that an attacker has the capability to eavesdrop on the communication channels connecting local users and the server without being detected, as depicted in Figure 1. The attacker's objective is to estimate the data distribution of the local user through reconstruction attacks.

As illustrated in Figure 1, in conventional GAN-based FL methods, such attackers may have unrestricted access to the entire GAN model. However, our proposed PS-FedGAN is specifically designed to address the security vulnerabilities associated with fully shared GAN models.

Mathematical Formulation: Let ψ be the parameters of the global/common model F that need to be optimized (minimized) for a given objective function $\mathcal{L}$ over data points that follow certain distribution p_{data}. Consider a distributed learning setup with N number of clients. Suppose that the ith user u_i has access to data from a distribution p_{datai} and uses publishing mechanism $\mathcal{M}_i$, then, we aim to optimize the distributed optimization as follows:

$$\min_{\psi}\{F(\psi) \triangleq \sum_{i=1}^{N} p_i f_i(\psi)\}, \quad (3)$$

where $\sum_{i=1}^{N} p_i = 1$. Here f_i is defined as

$$f_i(\psi) \triangleq \frac{1}{n_i}\sum_{j=1}^{n_i} \mathcal{L}(\psi; x_{ij}). \quad (4)$$

where x_{ij} is sampled from a data distribution $\hat{p}_{datai}$ that approximates p_{datai}, and $\hat{p}_{datai}$ is learned from $\mathcal{M}_i(.)$ and using the Eq. (2).

IV. Methodology

In this section, we introduce the overall framework of our proposed PS-FedGAN and subsequently describe the attacker models in our system.

A. PS-FedGAN

We now delve into the structures of the proposed PS-FedGAN. To tackle the privacy issues arising from full GAN sharing, we introduce a partial sharing approach within our PS-FedGAN framework. Specifically, we train two generators: generator G_u at the local user's end and generator G_s at the server side for each user. In order to bridge the training of two generators, i.e., one at a local user and the other at the server connected by a communication link, we share a common discriminator D_u trained only at the local user.

To visually depict the update process of PS-FedGAN, we present the user-server communication flow leading up to a

single-step update of the global classification model (C_l) in Figure 2. In the case of a single user, the training process begins with the local user training a local GAN model consisting of generator G_u and discriminator D_u. After completing a single batch training at the local user's end, the trained discriminator D_u is shared with the server through the PS-FedGAN publishing mechanism $\mathcal{M}_p$. The details of this publishing scheme will be further elaborated in the next section. In our study, we consider an untrustworthy communication link where attackers may potentially gain access to the PS-FedGAN publishing mechanism $\mathcal{M}_p$. On the server side, we initialize a separate generator G_s for each user, following the guidelines of PS-FedGAN Algorithm 1. Subsequently, the server updates its corresponding G_s based on information received through $\mathcal{M}_p$.

Before updating the global classification model C_l, we wait for a specific number of updates from the user side, denoted as N. In this context, N can represent an epoch for the local user. Consequently, we carry out N updates on both the local user's GAN and the corresponding G_s in parallel. The generated data from all the updated G_s models, corresponding to different local users, is combined with the server-available data to update the global model C_l. This cycle of updates and combination of generated data is repeated until C_l converges to the desired point.

When deploying the FL models, we assume each local user and the server have agreed on a secret seed value and the same architecture for G_u and G_s. The secret seed serves to initiate the weights of G_u and its corresponding G_s. Note that each user is assigned its own dedicated generator on the cloud server. Also, we impose no constraint on D_u. PS-FedGAN follows a batch-wise training process where we update G_s, G_u, and D_u at each step. Once G_u and G_s are initiated, local GAN training commences with D_u. In the following subsections, we provide a detailed explanation of the PS-FedGAN publishing mechanism $\mathcal{M}_p$ followed by an elaboration of the training process for both the local users and the cloud server in each communication round.

1) PS-FedGAN Publishing Mechanism $\mathcal{M}_p$: In PS-FedGAN, let z_t be the noise vector used to train G_u with random labels l_t at step/batch t at the user side. Let $\mathcal{M}(\theta)$ be a user-side publishing mechanism used to send trained parameters θ to the server. The PS-FedGAN publishing mechanism is defined as $\mathcal{M}_p$: $\mathcal{M}(D_u, z_t, l_t)$, where the local user releases D_u, z_t, and l_t to the server after each training step of D_u. We shall show that $\mathcal{M}_p$ preserves privacy and lowers communication cost compared to the full GAN releasing methods $\mathcal{M}(G_u, D_u)$ via theoretical analysis and experimental results in Section V and Section VI, respectively.

2) PS-FedGAN Local User Training: As discussed earlier, D_u is trained at the local user end after initializing G_u. The local user has the flexibility to choose any architecture for D_u, as long as it is trainable with G_u. Following the conventional GAN training, D_u is initially trained using both real images and images generated by G_u at each step. To handle the randomness of G_u in a deterministic manner, the parameters of D_u are updated during backpropagation (BP).

Depending on the convergence requirements of the architecture, multiple iterations of D_u training can be performed. Subsequently, random vectors z_t and corresponding random labels l_t are generated for training G_u. Before we train G_u, we release $\mathcal{M}_p$ to the server to minimize latency. Locally, G_u is trained using z_t and the corresponding l_t. Similar to general GAN training principles, the randomness in D_u is handled deterministically, to enable the training of GAN G_u. The parameters of G_u are then updated through BP.

3) PS-FedGAN Server Training: At the server, we maintain a dedicated generator (G_s) corresponding to each user initialized with the shared secret seed. Upon receiving the parameters $\mathcal{M}_p$ from the respective user, the training process for each G_s starts. During this process, it is not necessary to wait for other users to communicate with the server. Similar to the training of G_u, we train G_s using the received z_t and l_t from its corresponding user. To ensure consistency and avoid randomness in D_u, we employ the same techniques used by the corresponding user during training.

Next, we apply BP to update the parameters of G_s. Upon reception of updates from all users, we proceed to update the global classifier C_l. To update C_l at each iteration, we generate a preset number of samples using each G_s and combine them with a small amount of available real data pre-fetched by the server. Note that this small pre-fetched real data is the only advantage of the server over the potential eavesdroppers. This approach allows us to generate sufficient training samples that capture multiple private client datasets. We present the PS-FedGAN training algorithm in Algorithm 1.

Algorithm 1 PS-FedGAN: Training Algorithm. Minibatch GAN training for distributed GANs.

for each user i **do**
  Initial G_{u_i} and G_{s_i} using secret seed key_i
  Initiate the discriminator: D_{u_i}
end for
Initiate the global model C_l (classifier) and train using the available cloud data.
for each communication round, until C_l is converged **do**
  for number of training iterations **do**
    for each step t **do**
      # User side:
      $z_t, l_t \leftarrow$ random vectors, random labels
      Train D_{u_i}: $D_{u_i}^{(t+1)} \leftarrow \mathcal{F}(G_{u_i}^{(t)}(z_l, l_t), \mathcal{D}_{u_i}^{(t)})$, where $\mathcal{F}$ represent forward/back prop and weight updating
      Share $(\mathcal{D}_{u_i}^{(t+1)}, z_t, l_t) : \mathcal{M}_p(\mathcal{D}_{u_i}^{(t+1)}, z_t, l_t)$
      Train G_{u_i}: $G_{u_i}^{(t+1)} \leftarrow \mathcal{F}(G_{u_i}^{(t)}(z_t, l_t), D_{u_i}^{(t+1)})$
      # Server side:
      Train G_{s_i}: $G_{s_i}^{(t+1)} \leftarrow \mathcal{F}(G_{s_i}^{(t)}(z_t, l_t), D_{u_i}^{(t+1)})$
    end for
  end for
  # Server side:
  Train C_l with cloud data and synthetic data : $C_l^{(t+1)} \leftarrow \mathcal{F}(C_l^{(t)})$
end for

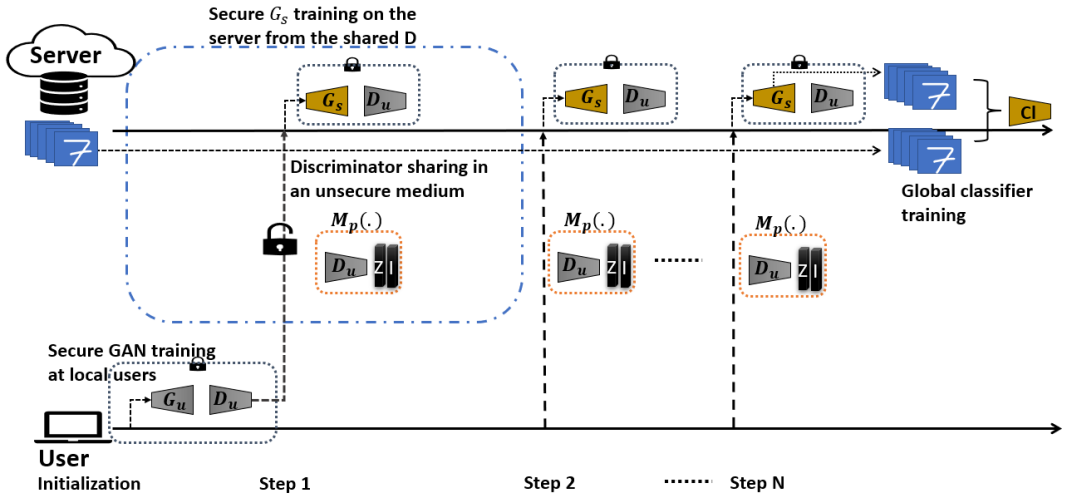


Fig. 2: User-server communication process until one update of the global classifier (C_l) shown for one user using the PS-FedGAN method: The local user first starts training a local cGAN (generator G_u and discriminator D_u). After a single batch training at the local user, trained D_u is shared with the server using $\mathcal{M}_p$. $\mathcal{M}_p : \mathcal{M}(D_u, z, l)$, where $\mathcal{M}$ is a publishing mechanism. z and l are noise vectors and corresponding fake labels used to train G_u at current step, respectively. At the server, we initiate a separate generator G_s for each user. For every received user update, we update G_s by one step. After N steps, we update the C_l using synthetic data generated from each G_s and cloud-available data. We continue this process till C_l converges.

B. Attacker Models

To assess the performance of PS-FedGAN against potential attackers, we focus on reconstruction attacks specifically. These attackers are denoted as $\mathcal{A}_R$. For any $\mathcal{M}(\theta)$, an attacker $\mathcal{A}_R$ attempts to reconstruct the training samples and assume $\mathcal{I}$ represents a reconstructed sample (e.g. image), i.e.,

$$\mathcal{A}_R : \mathcal{M}(\theta) \mapsto \mathcal{I}. \tag{5}$$

In our experimental setups, we consider two types of attackers: $\mathcal{A}_1$ and $\mathcal{A}_2$. The bottleneck faced by any attacker against our model lies in the hidden generator model which is more prone to privacy leakage. Therefore, the process of reconstructing synthetic data helps preserve privacy in terms of the initial weights and network architecture, as further elaborated in Section V. Predicting the exact architecture has become challenging owing to the many advances in deep learning techniques.

For an attacker to obtain information, it is crucial to eavesdrop from the very beginning and ensure they do not miss any round of communication exchanges. Such requirements prove difficult for potential privacy attackers. Additionally, the attacker must acquire knowledge of the initial weights of the generator, to be discussed in Section V. However, in practice, it is typically challenging, and often impossible, for an attacker to accurately estimate the generator's structure due to various practical constraints such as power, latency, and hardware capabilities. The complexity and variability of generator architectures could play a significant role in preserving privacy. The diverse range of architectures available for generators, coupled with their complexity, makes it highly uncertain for an attacker to accurately infer the precise model structure. Such inherent difficulty further enhances the preservation of privacy in the system. By utilizing complex and variable generator architectures, PS-FedGAN adds an additional layer of protection against potential privacy breaches.

Therefore, to define new attacks, we first introduce a multiplicative factor $r \in [0, 1]$. We assume that attackers $\mathcal{A}_1$ and $\mathcal{A}_2$ have the same architecture of G_u and the same weights and bias terms of G_u in every deep learning network layer except the first. Let w_1 and b_1 be the layer 1 weight and bias term of G_u. Layer 1 weight w_{a11} and bias term b_{a11} of $\mathcal{A}_1$ are set to $w_{a11} = rw_1$ and $b_{a11} = b_1$, respectively. Similarly for $\mathcal{A}_2$, we set layer 1's weight unperturbed as $w_{a21} = w_1$, and set layer 1's bias to $b_{a21} = rb_1$. Clearly, these two attackers' knowledge of G_u parameters are only slightly different from the true parameters with a multiplicative factor r on Layer 1's weights and biases, respectively.

V. Analytical Results

This section focuses on a convergence analysis of the proposed method and provides insights into privacy preservation with detailed proofs.

A. Convergence of D_u

We first show the convergence of two generators and a discriminator trained according to **Algorithm 1**. Suppose that the GAN trained at a local user u consists of a generator G_u capturing a probability distribution of p_{gu}, and a discriminator D_u. Let the local user's data distribution be $p_{data}(x)$. The local user (client) trains G_u with $z \sim p_z$. We have the following properties on model convergence.

Proposition 1. Two generators G_u and G_s trained on **Algorithm 1** with a shared discriminator D_u lead to the convergence to the same optimal discriminator D_u^* as in [6], i.e.,

$$D_u^* = \frac{p_{data}(x)}{p_{data}(x) + p_{gu}(x)} \tag{6}$$

which uniquely corresponds to the same given G_u.

Proof: In the proposed PS-FedGAN, **Algorithm 1** trains D_u using G_u while G_s has no influence on D_u. Therefore, as illustrated in [6], the discriminator training is valid with the value function $V(G_u, D_u)$ denoted by,

$$V(G_u, D_u) = \int_x p_{data} \log(D_u(x))\, dx + \int_z p_z(z) \log(1 - D_u(G_u(z)))\, dz. \tag{7}$$

Radon-Nikodym theorem leads to the following conclusion:

$$E_{z \sim p_z(z)} \log(1 - D_u(G_u(z))) = E_{x \sim p_{gu}(x)} \log(1 - D_u(x)). \tag{8}$$

Then, the value function can be recalculated as

$$V(G_u, D_u) = \int_x p_{data} \log(D(x)) + p_{gu}(x) \log(1 - D(x))\, dx. \tag{9}$$

Let $g(x) = a \log(x) + b \log(1 - x)$ for which

$$\frac{\partial g(x)}{\partial x} = \frac{a}{x} - \frac{b}{1 - x}, \tag{10}$$

and

$$\frac{\partial^2 g(x)}{\partial x^2} = -\frac{a}{x^2} - \frac{b}{(1 - x)^2}. \tag{11}$$

The derivatives give the unique maximizer $x = \frac{a}{a+b}$ with $\frac{\partial^2 g(x)}{\partial x^2} < 0$ for $a, b \in (0, 1)$. Hence, the optimal D_u^* can be calculated by

$$D_u^* = \frac{p_{data}(x)}{p_{data}(x) + p_{gu}(x)}. \tag{12}$$

Here, the unique maximizer implies the uniqueness of D_u^* for a given G_u.

■

In Proposition 1, we see that the D_u in PS-FedGAN converges to the same discriminator if we train G_u and D_u without G_s. That is, training G_s in the cloud does not hamper the convergence or performance of the local GAN training. On the other hand, we have a unique D_u given G_s. This property drives the convergence of G_s to G_u which is characterized in the following Propositions regarding the generator models.

B. Convergence of G_u and G_s

Proposition 2. Two generators G_u and G_s trained according to **Algorithm 1** with a shared D_u would converge to a unique $G^* = G_u^* = G_s^*$ which captures p_{data}.

Proof: Following **Proposition 1**, **Algorithm 1** converges to D_u^*. Therefore, in the training of the generator, we have the following optimization formulations, i.e.,

$$G_u^* = \arg\min_{G_u} V(G_u, D_u^*), \tag{13}$$

and

$$G_s^* = \arg\min_{G_s} V(G_s, D_u^*). \tag{14}$$

According to **Algorithm 1**, after each epoch of training in G_u and G_s, we have $G_u^{'} = G_s^{'}$. This is because the input to both networks and initial weights are the same, leading to the same loss with the same gradients to be updated in the backpropagation. Therefore, the training of G_s can be viewed as the traditional GAN training with G_s and D_u. Since z is shared in the communication, we have

$$V(G_s, D_u) = \int_x p_{data}(x) \log(D_u(x))\, dx + \int_z p_z(z) \log(1 - D_u(G_s(z)))\, dz. \tag{15}$$

Moreover, if G_s captures a distribution p_{gs}, from Eq. (15), we have

$$V(G_s, D_u) = \int_x p_{data}(x) \log(D_u(x))\, dx + \int_z p_z(z) \log(1 - D_u(G_u(z)))\, dz, \tag{16}$$

which could be further calculated via

$$V(G_s = G_u, D_u) = \int_x p_{data}(x) \log(D(x)) + p_{gs}(x) \log(1 - D(x))\, dx. \tag{17}$$

The uniqueness of D_u^* in Eq. (9) and Eq. (17) leads to $p_{gu} = p_{gs}$. Thus, G_s captures the same distribution as G_u. Hence, we can train two generators to capture the same distribution by only sharing a discriminator, without sharing original data explicitly. If G_u achieves G_u^*, we have $G_u^* = G_s^*$. Then the optimization problems in Eq. (13) and Eq. (14) reduce to

$$G^* = G_u^* = G_s^* = \arg\min_{G_u} V(G_u, D_u^*). \tag{18}$$

In order to prove that G_s^* captures p_{data}, we refer to [6]. For G^*, from the viewpoint of game theory, D_u fails to distinguish between true and fake data. Then, we have

$$D_u^* = \frac{p_{data}}{p_{data} + p_{gu}} = \frac{1}{2}, \tag{19}$$

which leads to $p_{gu} = p_{data}$. Now, we have $p_{gs} = p_{data}$. Thus, the server generator also captures the data distribution of the corresponding local user. Hence, G^* captures p_{data}. The uniqueness of G^* shall follow the same conclusion from [6]. Following the GAN training in [6], the global minimum of $M(G) = \max_D V(G, D)$ is gained if and only if $p_{gu} = p_{data}$. To prove both the above theorem and the uniqueness of G^*,

we first assume that $p_{gu} = p_{data}$. Then the value in Eq. (17) can be calculated as

$$\begin{aligned} V(G, D_u^*) &= \int_x p_{\text{data}}(x) \log(D_u^*) + p_{gs}(x) \log(1 - D_u^*)\, dx \\ &= \int_x p_{data}(x) \log(0.5) + p_{gs}(x) \log(0.5)\, dx \\ &= (\int_x (p_{data}(x) + p_{gs}(x))\, dx) \log(0.5) \\ &= (1+1) \log(0.5) \\ &= -\log(4). \end{aligned} \tag{20}$$

This provides us with the global minimum. On the other hand, for any G and D_u^*, For $M(G)$, we have,

$$\begin{aligned} M(G) &= \max_D V(G, D) \\ &= \int_x p_{data}(x) \log(D_u) + p_{gs}(x) \log(1 - D_u)\, dx \end{aligned} \tag{21}$$

$$\begin{aligned} &= \int_x p_{data}(x) \log(\frac{p_{data}}{p_{data} + p_{gu}}) \\ &\quad + p_{gs}(x) \log(1 - \frac{p_{data}}{p_{data} + p_{gu}})\, dx \\ &= \int_x p_{data}(x) \log(\frac{p_{data}}{p_{data} + p_{gu}}) \\ &\quad + p_{gs}(x) \log(\frac{p_{gu}}{p_{data} + p_{gu}})\, dx. \end{aligned} \tag{22}$$

After some manipulations, $M(G)$ can be calculated as

$$\begin{aligned} M(G) &= -\log(4) + \int_x p_{data}(x) \log(\frac{p_{data}}{(p_{data} + p_{gu})/2}) \\ &\quad + p_{gs}(x) \log(\frac{p_{gu}}{(p_{data} + p_{gu})/2})\, dx \\ &= -\log(4) + 2\text{JSD}(p_{data} | p_G), \end{aligned} \tag{23}$$

where JSD is the Jenson-Shannon Divergence which is non-negative. This yields $-\log(4)$ as the global minimum. Finally, it gives us $p_{data} = p_{gu}$, and the uniqueness of G^*.

■

B 1.3 Divergence of any G other than G_u or G_s

This proposition establishes that two distributed generators trained using PS-FedGAN converges to the same generative model. Moreover, this model is the same as the optimized model that one can obtain via classical GAN training. Another vital observation is that both G_u and G_s capture the user-side data distribution.

Proposition 3. Any other generator $G_{\mathcal{A}}$ failing to capture the weights and architectures of G_s or G_u, either in the initial state or in any single communication round, would fail to characterize the data distribution p_{data}.

Proof: From **Proposition 1** and **Proposition 2**, we have $p_{gu} = p_{data}$ and $p_{gs} = p_{data}$ which are made possible only by a unique G^* and D_u^*. As shown before, to obtain $G^* = G_u^* = G_s^*$, we need to have $G_u^{'} = G_s^{'}$ at each step, implying that G needs to capture all communication rounds. Therefore, these conditions restrict G to capture p_{data}.

■

This proposition provides insights into the capacity required by an attacker. To attack the proposed model, an attacker would need to predict a generator G using the information obtained from $\mathcal{M}_p$. However, to successfully carry out this attack, the attacker would need to possess precise knowledge of the generator's architecture, initial weights of either G_s or G_u, and would need to monitor and capture every round of communication. In our experiments, numerical results shall substantiate the need for these requirements and further demonstrate the difficulty an attacker would face when attempting to breach the privacy of the PS-FedGAN model.

C. DP property of $\mathcal{M}_p$

We now discuss the DP properties of the proposed methods. Let us denote the discriminator by $D = f(data)$ and the generator by $G = g(D, z)$, where z represents noise. From the post-processing property in [34], $g(f(D))$ is DP if $f(data)$ is DP. Thus, the generator G is DP whenever D is DP. Furthermore, if the training process is based on original data, FL-GAN follows DP [35].

In practical scenarios of PS-FedGAN, accessed models of D with quantized channel noise is DP [36] on the original weights of D, i.e., W_D. Since an attacker can only gain access to discriminator D during the communication round in the proposed PS-FedGAN framework, any generators reconstructed by the attackers from the hacked D are also DP on W_D.

Proposition 4. Any generator G reconstructed by the attacker shall be DP in a communication channel with quantization noise or channel-induced error.

This proposition shows that the attackers' estimated GAN model is DP on the original model weights W_D sent from a client to the server. Considering the mutual information of shared models and original data, we have $I(data, W_D) \leq I(data, data)$. Therefore, if we preserve privacy on W_D, we also preserve some privacy on the original data.

Proposition 5. $\mathcal{M}_p$ preserves privacy compared to full data sharing.

Clearly, full data sharing poses the most privacy leaks. Based on the foregoing propositions, we empirically evaluate the privacy-preserving capabilities of $\mathcal{M}_p$ next.

VI. Experiments

In this section, we present the test results of the proposed PS-FedGAN under non-IID user data distributions. In the first subsection, we evaluate performance when utilizing the proposed method and compare performance with existing FL methods. We then present privacy measures and also evaluate the associated communication cost. Our experiments use several well-known benchmark datasets, including MNIST [37], Fashion MNIST [38], SVHN [39], and CIFAR10 [40]. Experiments are implemented via torch 1.13 on a server with NVIDIA TITAN V GPUs with an Intel Xeon Silver 4110 CPU (32 cores) with a memory size of 64 GB.

To design the global classifier for the MNIST and Fashion MNIST datasets, our architecture utilizes a structure of two convolution layers, each of which uses batch normalization and ReLU activation followed by a fully connected layer with a dropout layer. It is then connected to two fully connected layers for output. For SVHN and CIFAR10 datasets, the global classifier consists of six convolution layers followed by two fully connected layers. Each odd-numbered convolution layer has a batch normalization layer, whereas even-numbered convolution layers have a max pool followed by a dropout layer. The final layer includes a batch for output.

A. Evaluation of Utility

Our study considers three distinct scenarios (Split 1, Split-2, and Split-3) of heterogeneous user cases, which are based on the work presented in [7]. Each case involves a total of 10 users and the following details:

- **Split-1**: In this case, training data is divided into 10 shards, each containing samples from a single class. Each user is randomly assigned one distinct shard.
- **Split-2**: This case generates 20 training data shards, each consisting of samples from a single class. Two shards are randomly assigned to each user without overlap.
- **Split-3**: This case generates 30 shards, each containing samples from a single class. Three shards are randomly assigned to each user without overlap.

As illustrated above, the experimental setup for accuracy evaluation consists of three different settings. From Split-1 to Split-3, we reduce the skewness in data by increasing the number of classes per user while keeping the total number of data points per user constant. Specifically, we test PS-FedGAN performance against extreme skewness by considering users with a single class in Split-1. Subsequently, we degrade the skewness by increasing the number of classes by two and three in Split-2, and Split-3 respectively.

As a utility measure, we select the classification accuracy of the global model in a supervised setup. We compare our results against several existing FL alternatives: FedAvg (FA) [1], FedProx (FP) [18], SCAFFOLD (SD) [19], Naivemix (NM) [41], FedMix (FM) [41], and SDA-FL (SL) [7] as baselines. We also compare our partially-shared PS-FedGAN (PS) with the fully-shared GAN (FG) as a performance benchmark. The results of Split-1/2/3 are shown in Table II.

Table II shows the superior performance of the proposed PS-FedGAN over most existing methods except for the FG. More specially, we see a significant improvement in PS-FedGAN in Split-1 for the CIFAR10 dataset. Compared with full-GAN (FG) sharing which is the best possible benchmark, our PS-FedGAN achieves similar performance but at significantly reduced communication cost and substantial privacy loss as further shown in Table VI. Note that in the above test, the SDA-FL method requires an infinite privacy budget to achieve the desired utility. This indicates that SDA-FL fails to balance privacy and utility. Additionally, we note that other alternatives such as [7] are able to access more real data for new classes seen from Split-1 (10% real data in the respective received class) to Split-3 (3.33% real data per class and 3 such classes), whereas our proposed PS-FedGAN only maintains a small unchanged amount (1%) of real data on the server.

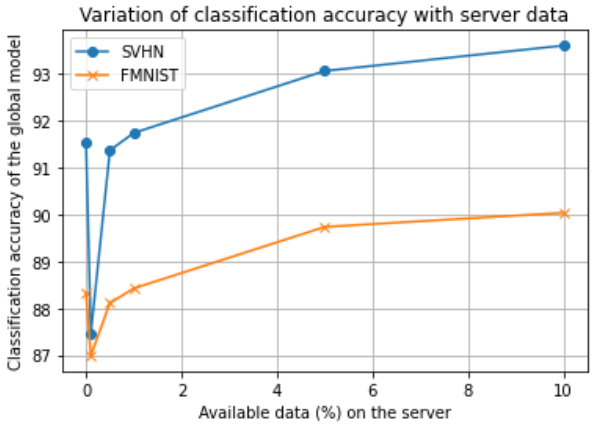


Fig. 3: The variation of the classification accuracy with the percentage of available data in the server.

B. Ablation Study of the Effect of the Server Data

We now provide an ablation study to analyze the effect of the server data. We first present the accuracy compared to other state-of-the-art (SOTA) approaches without any server data (0%), as shown in Table III. From the results, our proposed PS-FedGAN still achieved the best performance, even without any server data, which validates the power of our proposed framework. Furthermore, to analyze the effect of server data, we analyze the variation of the accuracy against the data availability on the server using the same Split-1 for the datasets, i.e., FMNIST and SVHN. Generally, as illustrated in Figure 3, the number of server data and classification accuracy of our model are positively correlated. When some server data is first introduced (below 0.1%), we can observe an accuracy drop, around 0.1% of server data. There are several possible reasons. First, the number of samples in 0.1% data samples is tiny and cannot well represent the learned distribution. Secondly, a better data combination algorithm, such as Mixup [42], can be used. Nevertheless, as shown in Figure 3, even with a very small ratio of server data, such as 0.1%, our proposed PS-FedGAN could achieve a better performance against SOTA methods, such as SL [7] trained on 10% real data (skewed) using the Mixup.

C. Privacy Evaluation

We now evaluate the privacy of PS-FedGAN with respect to reconstruction attacks, which are often viewed as one of the most dangerous attacks. Here, we utilize the MNIST dataset. To evaluate the efficacy against reconstruction attacks, we use several proxies to measure the degree of privacy leakage, including normalized mean square error (NMSE), structural similarity index (SSIM) [43], and classification accuracy. We consider two different non-IID user setups:

- **Setup-1** includes three users. User-1 has access to classes {0,1}, user-2 has access to classes {2,3,4}, and user-3

TABLE II: Classification accuracy (best) of existing FL methods and full GAN sharing compared with PS-FedGAN in Split-1, Split-2, and Split-3. Some of the results are from [7]. We assume 1% of the training data of each dataset available in the cloud for FG and PS-FedGAN.

	Split-1				Split-2				Split-3			
Method	MNIST	FMNIST	SVHN	CIFAR10	MNIST	FMNIST	SVHN	CIFAR10	MNIST	FMNIST	SVHN	CIFAR10
FA	83.44	16.50	14.05	18.36	97.61	73.50	81.11	61.28	98.42	82.47	84.18	79.33
FP	84.17	57.14	17.53	11.24	97.55	75.76	86.28	63.16	98.38	83.43	92.15	79.54
SD	25.39	56.80	11.64	12.81	94.17	70.82	73.34	60.78	96.89	77.68	80.13	79.35
NM	84.35	66.62	14.35	14.39	84.35	79.54	84.64	64.39	98.11	82.09	92.30	78.92
FM	90.96	72.11	16.78	13.57	90.96	82.41	86.61	65.76	98.46	84.65	92.61	79.49
SL	98.19	85.70	88.46	37.70	98.26	86.87	90.70	67.89	98.50	87.06	93.16	84.56
FG	98.41	88.77	91.92	66.98	98.41	89.01	92.61	69.91	98.46	89.26	93.05	82.09
PS	98.31	88.42	91.73	66.96	98.44	89.02	92.30	69.89	98.54	89.28	93.23	82.05

TABLE III: Accuracy of different methods compared in the experiment setting Split-1

Method	MNIST	FMNIST	SVHN	CIFAR10
FA	83.44	16.50	14.05	18.36
FP	84.17	57.14	17.53	11.24
SD	25.39	56.80	11.64	12.81
NM	84.35	66.62	14.35	14.39
FM	90.96	72.11	16.78	13.57
SL	98.19	85.70	88.46	37.70
PS (no server data)	97.25	**88.32**	**91.52**	**58.60**

has access to remaining classes {5,6,7,8,9}. Also attacker model $\mathcal{A}_1$ described in Section IV-B is used.

- In **Setup-2**, we consider 10 users with data splitting in Split-1, where attacker model $\mathcal{A}_2$ is applied here in reconstruction attacks.

Classification Accuracy: To evaluate the attacker's classification accuracy on reconstructed images, we first consider **Setup-1**. In this setup, we assume that the attacker has access to all the elements released by the releasing mechanism $\mathcal{M}p$. Furthermore, we assume that the attacker can accurately guess the exact generator architecture. Note that at the beginning of training, the generator weights are different for $\mathcal{A}_1$ ($w_{a11} = rw_1$), from the cloud generator as mentioned earlier. The generators are trained simultaneously at the user, the server, and the attacker.

We then select an attacking classifier trained on the original MNIST training set and infer data generated by the attacker's generator. Here, the attacker guesses correctly that the user works with MNIST data. Table IV illustrates the attacker's performance in different r. From Table IV, we see that the attacking gains in accuracy with increasing r. In Table IV, we evaluate the classification accuracy at the cloud using the same attacking classifier, which reflects the potential of an ideal attacker and the utility at the cloud. Table IV shows that an attacker with some disparity in the initial weights only performs like a random guess. Note that in this scenario, user-1 has 2 classes, user-2 has 3 classes, and user-3 has 5 classes. These results suggest that an attacker must acquire extraordinarily accurate information on architecture and initial model weights to achieve higher and nontrivial inference accuracy. Since such requirements are improbable and impractical to achieve, our results establish the robustness of our proposed method against classification attacks.

TABLE IV: Classification accuracies (best) on attacker's generator and cloud's generator: How attacker's performance (classification accuracy %) varies as r varies on $\mathcal{A}_1$ on Setup-1. Here $r_1 = 1-1\times10^{-4}$,$r_2 = 1-1\times10^{-7}$,$r_3 = 1-1\times10^{-15}$, and $r_4 = 1 - 1 \times 10^{-21}$

r	Attacker's Models			Cloud's models		
	User1	User2	User3	User1	User2	User3
r_1	0.4955	0.3390	0.2048	0.9911	0.9902	0.9722
r_2	0.4933	0.3323	0.1945	0.9983	0.9727	0.9610
r_3	0.4941	0.3299	0.1990	0.9980	0.9677	0.9690
r_4	0.5027	0.3341	0.2056	0.9981	0.9746	0.9754

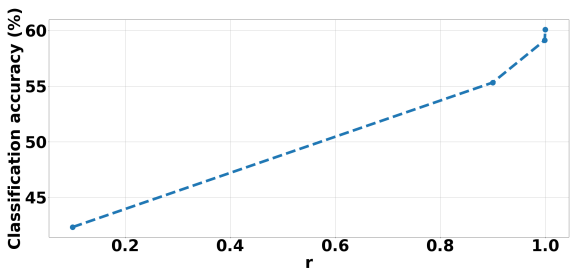


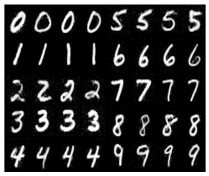
Fig. 4: Reconstruction attacks: Different r from 0.1 to 0.999999 evaluated under $\mathcal{A}_2$.

Next, we consider $\mathcal{A}_2$'s classification accuracy on reconstructed images. For this attacker, the difference between its generator and the cloud generator lies in the first layer bias term ($b_{a21} = rb_1$). We evaluate $\mathcal{A}_2$'s performances on **Split-1**. Figure 4 illustrates the attacker's performance for different r. Figure 4 shows that the attacking gains better classification with increasing r. Similar to the first type of attackers in $\mathcal{A}_1$, these results suggest that an attacker from $\mathcal{A}_2$ also needs to have extremely accurate architecture knowledge and the initial weights. This test case further establishes the robustness of PS-FedGAN robust against classification attacks.

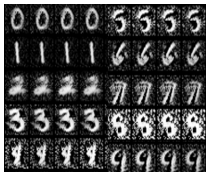
Reconstruction Quality and Similarity As presented in the paper [44], reconstruction quality and similarity can be used as a measure of privacy leakage. For this, we consider 2 metrics, i.e., SSIM and NMSE, in **Setup-1**. Table V compares the corresponding generations of the attacker and the cloud based on the similarity between each generated image. From Table V, we can see that the NMSE values achieved by

TABLE V: Reconstruction attacks: Attacker $\mathcal{A}_1$ reconstruction quality on 3 different users in Setup-1. For NMSE lower values are better and for SSIM higher values are better.

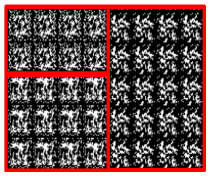
Metric	User1	User2	User3
NMSE	1.4108	1.4585	1.2553
SSIM	0.01956	0.0253	0.0229



(a) Regenerated data samples in the cloud server.



(b) Regenerated samples of $\mathcal{A}_2$ (each # for one user).



(c) Regenerated samples of $\mathcal{A}_1$ (each block for one user).

Fig. 5: Divergence of the attacker: Any attacker fails to initiate with the same parameters as G_u or G_s fails to capture the local user's distribution ($r = 0.9999$ for both $\mathcal{A}_1$ and $\mathcal{A}_2$).

the attackers are high whereas SSIM (maximum 1) is very low. These results indicate that the attacker-generated images cannot capture the true data distribution. We shall further show in a later section that, if the attacker deviates from the actual generator weights, its convergence would be to a trivial point. As a result, no underlying user data properties are captured. The generator generates an informationless image which is enough to pass the discriminator. Hence, the generator stops learning the data distribution. As illustrated by the visual examples in Figure 5(c), the attacker generates single-mode outputs for each user.

D. Evaluation of Communication Cost

Table VI illustrates the number of parameters that need to be shared between each user and the cloud server per communication round for both full GAN and PS-FedGAN. From the table, we can see a significant saving in PS-FedGAN compared to classical full GAN sharing. The advantage would be similar for various different GAN architectures.

E. Convergence of Cloud Generators

As an example to illustrate the generator convergence, we illustrate the cloud generator loss of all the 10 users under **Split-1** in Figure 6. The results show that our proposed PS-FedGAN can converge well for all users, which further demonstrates the practicality of discriminator sharing.

F. Divergence of Attacker's Generator

From Figure 7 we see that so long as the attacker ($\mathcal{A}_1$) deviates slightly from the actual generator weights, the attacker would converge to a trivial point. Hence, almost no underlying user data properties are captured by the attacker.

TABLE VI: Number of parameters communicated at each step for Split-3.

Dataset	Full GAN sharing: $\mathcal{M}(G, D)$	PS-FedGAN: $\mathcal{M}_p(D, z, l)$
MNIST	3 M	1.5 M
FMNIST	5 M	1.5 M
SVHN	2 M	0.7 M
CIFAR10	6.8 M	2.98 M

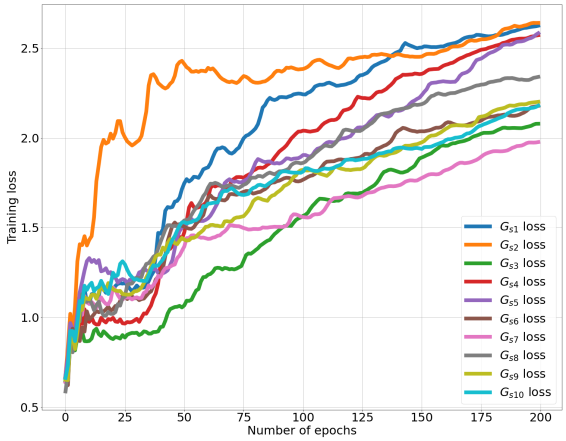


Fig. 6: Generator loss of all the cloud generators in Split-1 for MNIST dataset.

G. $\mathcal{A}_1$ Performances

We next provide more visual results to illustrate the robustness of the proposed methods against the attacks of $\mathcal{A}_1$. Figure 8 shows cloud-generated images compared to $\mathcal{A}_1$ generated images for the SVHN dataset for three users at some intermediate step (epoch 20) while Figure 9 shows the regenerated samples in the FMNIST dataset. From these visualization comparisons, we see that $\mathcal{A}_1$ fails to capture the underlying data statistics as well as to regenerate the meaningful synthetic data samples.

Discussion: Note that, our objective is to develop a privacy-preserved and communication-efficient GAN-based FL for general IoT applications. Thus, we focus our analysis without considering the possibility of imperfect links. However, the case of imperfect channels is also of interest. For example, if packet loss is considered, Automatic Repeat Query (ARQ) [45] or hybrid ARQ allows the transmission of lost packets, which would introduce some additional delay though we can still achieve the same convergence. Different channel noises or fading characteristics would directly lead to different packet loss rate and delay variation. For unrecoverable packet loss, We may investigate its effect on training convergence in future studies.

VII. Conclusions

This work develops PS-FedGAN, a novel GAN-based FL framework that can beneficially preserve local data privacy and

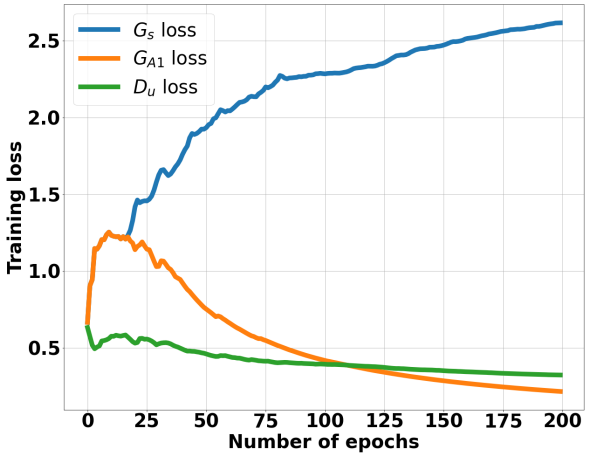


Fig. 7: Divergence of the attacker. Any attacker fails to initiate with the same parameters as the optimal generators, i.e., G_u or G_s, which indicates the attacks would fail to capture the local user's data distribution.

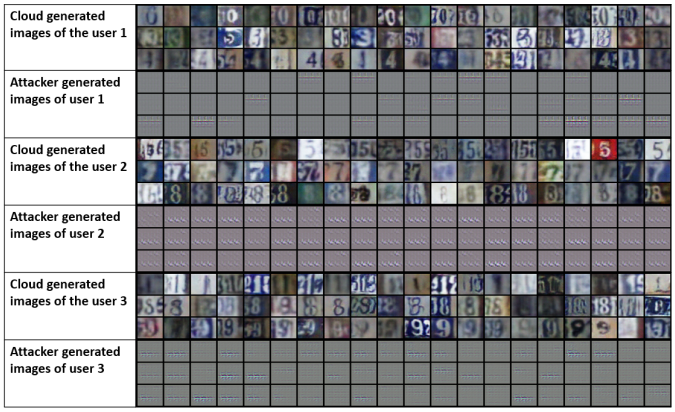


Fig. 8: Cloud generated images compared to $\mathcal{A}_1$ generated images for Split-3 for SVHN dataset.

reduce communication overhead in comparison with existing FL proposals. Our proposed PS-FedGAN achieves learning utility very close to that of fully shared GAN architecture, while providing strong data privacy and significant reduction of communication cost. Empirical results further demonstrate superior results against state-of-the-art GAN-based FL frameworks. This PS-FedGAN principle and architecture can be directly generalized to incorporate other existing GANs. In future work, we plan to further investigate the effect of lossy networking channels and to improve PS-FedGAN's robustness against non-ideal network links. Another promising extension may consider different FL models and data heterogeneity as outlined in [46].

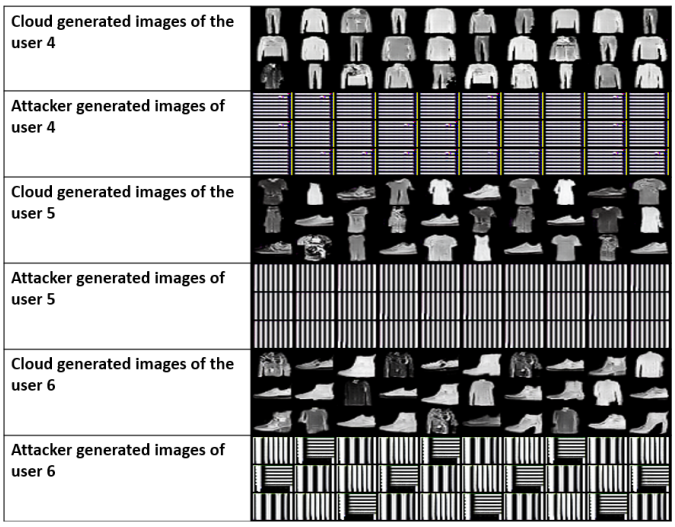


Fig. 9: Cloud generated images compared to $\mathcal{A}_1$ generated images for Split-3 for FMNIST dataset.

REFERENCES

[1] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. 2017 Artificial Intelligence and Statistics*, Ft. Lauderdale, FL, USA, Apr. 2017, pp. 1273–1282.

[2] J. Konecný, H.B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency", *arXiv:1610.05492*, Oct. 2016.

[3] S. Jing, A. Yu, S. Zhang, and S. Zhang, "Fedsc: provable federated self-supervised learning with spectral contrastive objective over non-i.i.d. data", to appear in *2024 International Conference on Machine Learning (ICML)*, Vienna, Austria, Jul. 2024.

[4] B. Ghimire and D. B. Rawat, "Recent advances on federated learning for cybersecurity and cybersecurity for federated learning for internet of things," in *IEEE Internet of Things Journal*, vol. 9, no. 11, pp. 8229–8249, Feb. 2022.

[5] D. C. Nguyen, M. Ding, P. N. Pathirana, A. Seneviratne, J. Li, and H. V. Poor, "Federated learning for internet of things: a comprehensive survey," in *IEEE Communications Surveys & Tutorials*, vol. 23, no. 3, pp. 1622–1658, Apr. 2021.

[6] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proc. 2014 Advances in Neural Information Processing Systems*, Montreal, Canada, Dec. 2014, Part 27.

[7] Z. Li, J. Shao, Y. Mao, J. H. Wang, and J. Zhang, "Federated learning with gan-based data synthesis for non-iid clients," in *Proc. 2022 International Workshop on Trustworthy Federated Learning*, Vienna, Austria, Jul. 2022, pp.17-32.

[8] B. Xin, W. Yang, Y. Geng, S. Chen, S. Wang, and L. Huang, "Private fl-gan: Differential privacy synthetic data generation based on federated learning," in *Proc. 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Barcelona, Spain, May 2020, pp. 2927–2931.

[9] M. Fang, X. Cao, J. Jia, and N. Gong, "Local model poisoning attacks to Byzantine-Robust federated learning," in *Proc. 29th USENIX Security Symposium (USENIX Security 20)*, Boston, MA, USA, Aug. 2020, pp. 1605–1622.

[10] L. Lyu, H. Yu, X. Ma, L. Sun, J. Zhao, Q. Yang, and P. S. Yu, "Privacy and robustness in federated learning: attacks and defenses," in *IEEE Trans. on Neural Networks and Learning Systems*, Early Access, Nov. 2022.

[11] B. Hitaj, G. Ateniese, and F. Perez-Cruz, "Deep models under the gan: information leakage from collaborative deep learning," in *Proc. the 2017 ACM SIGSAC Conference on Computer and Communications Security*, New York, NY, USA Oct. 2017, pp. 603–618.

[12] J. Hayes, L. Melis, G. Danezis, and E. De Cristofaro, "Logan: evaluating information leakage of generative models using generative adversarial networks," *arXiv:1705.07663*, vol. 18, May 2017.

[13] C. Dwork, "Differential privacy," in *Automata, Languages and Programming*, M. Bugliesi, B. Preneel, V. Sassone, and I. Wegener, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 1-12. Accessed: Apr. 23, 2024. [Online]. Available: https://link.springer.com/chapter/10.1007/11787006_1.

[14] C. Xu, J. Ren, D. Zhang, Y. Zhang, Z. Qin, and K. Ren, "Ganobfuscator:

Mitigating information leakage under gan via differential privacy," in *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 9, pp. 2358–2371, Feb. 2019.

[15] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *Proc. 2017 IEEE Symposium on Security and Privacy (SP)*, San Jose, CA, USA, May 2017, pp. 3–18.

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

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

[18] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," in *Proc. Machine Learning and Systems*, Austin, TX, USA, Mar. 2020, pp. 429–450.

[19] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh, "SCAFFOLD: stochastic controlled averaging for federated learning," in *Proc. 37th International Conference on Machine Learning*, Virtual, Jul. 2020, pp. 5132–5143.

[20] M. Luo, F. Chen, D. Hu, Y. Zhang, J. Liang, and J. Feng, "No Fear of Heterogeneity: Classifier Calibration for Federated Learning with Non-IID Data," in *Proc. Advances in Neural Information Processing Systems*, Virtual, Dec. 2021, pp. 5972–5984.

[21] N. Yoshida, T. Nishio, M. Morikura, K. Yamamoto, and R. Yonetani, "Hybrid-fl for wireless networks: cooperative learning mechanism using non-iid data," in *Proc. 2020 IEEE International Conference on Communications*, Dublin, Ireland, Jun. 2020, pp. 1-7.

[22] W. Li, J. Chen, Z. Wang, Z. Shen, C. Ma, and X. Cui, "Ifl-gan: improved federated learning generative adversarial network with maximum mean discrepancy model aggregation," in *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–14, Apr. 2022.

[23] X. Cao, G. Sun, H. Yu, and M. Guizani, "Perfed-gan: personalized federated learning via generative adversarial networks," in *IEEE Internet of Things Journal*, vol. 10, no. 5, pp. 3749–3762, May 2023.

[24] W. Hao, M. El-Khamy, J. Lee, J. Zhang, K. J. Liang, C. Chen, and L. Carin, "Towards fair federated learning with zero-shot data augmentation," in *Proc. 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, Nashville, TN, USA, Jun. 2021, pp. 3305-3314.

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

[26] W. Ni, J. Zheng, and H. Tian, "Semi-federated learning for collaborative intelligence in massive iot networks," in *IEEE Internet of Things Journal*, vol. 10, no. 13, pp. 11942-11943, Jul. 2023.

[27] B. Ghimire and D. B. Rawat, "Recent advances on federated learning for cybersecurity and cybersecurity for federated learning for internet of things," in *IEEE Internet of Things Journal*, vol. 9, no. 11, pp. 8229-8249, Jun. 2022.

[28] A. Imteaj, U. Thakker, S. Wang, J. Li, and M. H. Amini, "A survey on federated learning for resource-constrained iot devices," in *IEEE Internet of Things Journal*, vol. 9, no. 1, pp. 1–24, Jul. 2021.

[29] Z. Chen, A. Fu, Y. Zhang, Z. Liu, F. Zeng, and R. H. Deng, "Secure collaborative deep learning against gan attacks in the internet of things," in *IEEE Internet of Things Journal*, vol. 8, no. 7, pp. 5839–5849, Oct. 2020.

[30] K. Wang, N. Deng, and X. Li, "An efficient content popularity prediction of privacy preserving based on federated learning and wasserstein gan," in *IEEE Internet of Things Journal*, vol. 10, no. 5, pp. 3786–3798, May 2022.

[31] H. Zhang, Z. Zhang, A. Odena, and H. Lee, "Consistency regularization for generative adversarial networks," *arXiv:1910.12027*, Oct. 2019.

[32] Y. Wu, Y. Kang, J. Luo, Y. He, and Q. Yang, "Fedcg: leverage conditional gan for protecting privacy and maintaining competitive performance in federated learning," in *Proc. International Joint Conference on Artificial Intelligence*, Virtual, Aug. 2021.

[33] M. Mirza and S. Osindero, "Conditional generative adversarial nets," *arXiv:1411.1784*, Nov. 2014.

[34] C. Dwork and A. Roth, "The algorithmic foundations of differential privacy," in *Found. Trends Theor. Comput. Sci.*, vol. 9, pp. 211–407, Aug. 2014.

[35] B. Xin, Y. Geng, T. Hu, S. Chen, W. Yang, S. Wang, and L. Huang, "Federated synthetic data generation with differential privacy," in *Neurocomputing*, vol. 468, pp. 1–10, Jan. 2022.

[36] S. Amiri, A. Belloum, S. Klous, and L. Gommans, "Compressive differentially private federated learning through universal vector quantization," in *Proc. 2021 AAAI Workshop on Privacy-Preserving Artificial Intelligence*, Virtual, Feb. 2021, pp. 2-9.

[37] L. Deng, "The mnist database of handwritten digit images for machine learning research," in *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 141–142, Nov. 2012.

[38] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," *arXiv:1708.07747*, Aug. 2017.

[39] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, A. Y. Ng et al., "Reading digits in natural images with unsupervised feature learning," in *NIPS workshop on deep learning and unsupervised feature learning*, Granada, Spain, Dec. 2011, p. 7.

[40] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," in *Technical Report 0*, University of Toronto, Toronto, Ontario, 2009.

[41] T. Yoon, S. Shin, S. J. Hwang, and E. Yang, "Fedmix: Approximation of mixup under mean augmented federated learning," *arXiv:2107.00233*, Jul. 2021.

[42] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "Mixup: beyond empirical risk minimization," *arXiv:1710.09412*, Oct. 2017.

[43] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," in *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, Apr. 2004.

[44] D. Dangwal, V. T. Lee, H. J. Kim, T. Shen, M. Cowan, R. Shah, C. Trippel, B. Reagen, T. Sherwood, V. Balntas et al., "Analysis and mitigations of reverse engineering attacks on local feature descriptors," *arXiv:2105.03812*, May 2021.

[45] A. M. Albarrak, "Similarity-aware query refinement for data exploration," PhD Thesis, School of Information Technology and Electrical Engineering, The University of Queensland. [Online]. Available: https://doi.org/10.14264/uql.2018.416.

[46] S. Vahidian, M. Morafah, M. Shah, and B. Lin, "Rethinking data heterogeneity in federated learning: introducing a new notion and standard benchmarks," in *IEEE Transactions on Artificial Intelligence*, vol. 5, no. 3, pp. 1386-1397, Mar. 2024.