Active Gradient Manipulation for Privacy Breaching in Vertical Federated Learning

Tre' R. Jeter – Minh N. Vu, Raed Alharbi, Jung Taek Seo, and My T. Thai*

***Abstract*—Federated Learning (FL) has emerged as a promising approach for privacy-preserving collaborative machine learning. Specifically, vertical FL (vFL) allows various devices in multi-agent systems to collectively train models on vertically partitioned data while safeguarding sensitive information. Recent research on vFL privacy analysis primarily explores passive settings where attackers adhere to the FL protocol. This perspective may underestimate the threats posed by vFL, as practical adversaries can deviate from the protocol to enhance their attack capabilities. In response, this work proposes two novel *active* data reconstruction attacks to compromise data privacy. Each attack induces gradient manipulation during the training phase to breach data privacy. Including an *Active Inversion Network* (AIN), our first attack exploits a subset of known data in the training set to make passive parties train an auto-encoder (AE) to reconstruct their private data. The second attack introduces an *Active Generative Network* (AGN) that relies only on the data distribution to train a conditional generative adversarial network (C-GAN) for private feature reconstruction. Our experiments demonstrate the effectiveness of both attacks in three real-world datasets: MNIST, CIFAR10, and USCensus. Additionally, we provide valuable insights and guidelines for enhancing the security of vFL systems through the application of calibrated noise via Local Differential Privacy (LDP).**

***Impact Statement*—Vertical Federated Learning (vFL) enables organizations to collaboratively train machine learning models on vertically partitioned data while preserving privacy. For instance, healthcare providers and insurance companies often need to collaborate on patient cases but cannot share overlapping information. Healthcare providers retain medical records, imaging data, and clinical notes, while insurance companies hold claims, policy details, and payment history. Personally Identifiable Information (PII) such as names, social security numbers, and financial details are never shared. However, existing research has underestimated the capabilities of real-world attackers in reconstructing PII from shared data. To address this, we introduce two novel *active* attack scenarios consisting of an Active Inversion Network (AIN) and an Active Generative Network (AGN), highlighting the critical need for robust defenses in vFL systems beyond the passive scope.**

This work was supported in part by the National Science Foundation under grants CNS-1935923 and IIS-2416606. This work was also supported in part by the Korea Institute of Energy Technology Evaluation and Planning (KETEP) grant funded by the Korea government (MOTIE) (RS-2023-00303559, A Study on Development of Cyber-Physical Attack Response System and Security Management System for Maximizing Availability of Real-Time Distributed Resources).

Tre' R. Jeter, Minh N. Vu, and My T. Thai are with the Department of Computer and Information Science and Engineering at the University of Florida in Gainesville, FL 32611. (Email: {t.jeter, minhvu}@ufl.edu and mythai@cise.ufl.edu)

Raed Alharbi is with the Department of Computer Science at Saudi Electronic University in Riyadh, Saudi Arabia. (Email: ri.alharbi@seu.edu.sa)

Jung Taek Seo is with the Department of Smart Security at Gachon University in Seongnam-daero 1342, Seongnam-si, Republic of Korea. (Email: seojt@gachon.ac.kr)

* Corresponding author.

The first two authors contributed equally to this work.

***Index Terms*—Data Privacy, Active Data Reconstruction Attacks, Local Differential Privacy, Vertical Federated Learning, Gradient Manipulation**

I. Introduction

Federated Learning (FL) allows multiple parties to jointly train machine learning models without sharing their sensitive data. Vertical FL (vFL) is a variant of FL that allows multiple parties to jointly train while using different features of the same dataset [1], [2]. vFL's primary objective is to enable collaborative model training that fully leverages vertically partitioned and distributed data while preserving the privacy of sensitive information. The adoption of vFL in Large-Scale Multi-Agent Systems has experienced a notable increase in recent years. This rise is credited to its ability to address challenges associated with data privacy, training efficiency, and decentralized control. These advantages have been exemplified in diverse sectors such as local banks, insurance companies, and healthcare facilities [3]–[5].

Given that vFL involves multiple parties, not all of whom can be fully trusted, it is crucial to prevent malicious participants from compromising sensitive data. While prior research has suggested that data protection is enhanced when distributed parties use their own datasets and avoid sharing data, the potential for malicious participants to exploit privacy vulnerabilities in vFL remains a concern. In particular, vFL has proven to be susceptible to label inference [6], [7], data reconstruction [8], [9], and property inference attacks [9], during training and inference phases. To counter, provisional defense strategies have been introduced [10], [11]. However, both attack and defense strategies have not fully conveyed vFL's privacy risks since they under-exploit the realistic capabilities of adversarial participants. In fact, all previous studies consider the *passive* setting, in which adversarial participants still abide by the system protocol. Previous threat models overlook the privacy risks posed by *active* adversaries who can deviate from the protocol without detection [12]–[14].

To our knowledge, this work introduces the first investigation on *active* privacy attacks conducted by an *actively* malicious party. Note that "active party" and "passive party" refer to the participants in vFL who own or does not own the labels, respectively. On the other hand, the "active" or "passive" denotation of our privacy attacks indicates whether the attacker deviates from the protocol. Under the active setting, we investigate how effectively an active party can reconstruct the private training data of passive parties by deviating from the protocol. Differing from previously examined passive privacy breaches, our attacks manipulate the

gradients computed during training and transmits them to the passive parties. This manipulation allows the adversary to subsequently exploit these gradients to reveal the data of each passive participant during the inference phase.

Our primary contributions include:

- We introduce a novel data reconstruction attack, consisting of an *Active Inversion Network* (AIN), which manipulates gradients during training by leveraging a small portion of publicly available data. This attack deceives passive parties into training an auto-encoder (AE) that reconstructs their private features.
- We propose a second reconstruction attack, consisting of an *Active Generative Network* (AGN), designed for scenarios where the attacker lacks direct knowledge of private features. The AGN exploits the training data distribution by guiding passive parties to train a conditional generative adversarial network (C-GAN) that reconstructs private features based on known attributes.
- Extensive experiments on three real-world datasets—MNIST [15], CIFAR10 [16], and USCensus [17]—demonstrate that our attacks significantly improve private data reconstruction compared to passive attacks in vFL.
- Additionally, we evaluate the effectiveness of our attacks on data protected by Local Differential Privacy (LDP), analyzing the trade-offs between model performance and privacy across different tasks and datasets.

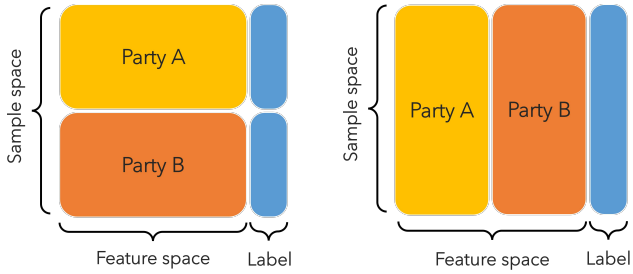
Organization. The manuscript is structured as follows: Section II covers relevant background on traditional FL, vFL, data reconstruction attacks in vFL, and the LDP defensive mechanism. Section III describes notations adopted in this paper and our threat models in the active setting. Section IV provides the descriptions of our proposed active data reconstruction attacks consisting of an Active Inversion Network (AIN) and Active Generative Network (AGN). A thorough experimental evaluation is discussed in Section V. Section VI discusses possible defense strategies to mitigate the proposed attacks. Section VII concludes the paper.

II. Background and Related Works

This section covers relevant background information and related works. We start with brief descriptions of FL and vFL. Then, we summarize well-known data reconstruction attacks executed while training neural networks in vFL. Next, we provide a brief introduction to LDP. Finally, we discuss the limited research within the active attack setting and explain the additional contributions made in this work.

A. Federated Learning

Initially introduced by Google [18], FL is designed for scenarios involving cross-device collaboration, where many devices work together to train a deep learning model with the assistance of a central server. Particularly, FL is an iterative learning framework and the most common versions use gradient descent [19]. The central server randomly initializes global model parameters at the start of the process. In each iteration of training, a random subset of clients is selected to participate. Selected users obtain the global parameters from the server to compute their local gradients based on their local dataset. The computed gradients are uploaded to the server for aggregation into a new global model. The server redistributes the global model parameters to a new randomly selected subset of clients. This iterative process continues until the global model converges.



(a) Horizontal Federated Learning (b) Vertical Federated Learning

Figure 1: Two types of Federated Learning.

B. Vertical Federated Learning

Influenced by how data is divided among users, FL typically takes two primary forms: horizontal and vertical. In the horizontal setting (hFL), all participants share the same feature space while working with distinct data samples (Figure 1a). Conversely, in the vertical setting (vFL), each participant maintains different sets of features while sharing the same set of data samples (Figure 1b) [2]. In this context, labels are regarded as special features owned by a participant known as the *active party*. All other participants are referred to as *passive parties*. Each party in vFL maintains its own local model. Both training and inference phases in vFL necessitate the sharing of intermediate results (signals) among parties.

The need for vFL has increased considerably across various industries. For instance, companies and institutions with limited and fragmented data have actively sought partnerships in advanced artificial intelligence technology to optimize data utilization [20], [21]. At the same time, the growing global concerns about data leakages and breaches have raised the bar for privacy and security. As a result, numerous privacy-preserving initiatives supporting vFL have emerged and garnered significant attention [3]–[5], [22].

The general training procedure for vFL comprises two phases: Entity Alignment (EA) and Privacy-Preserving Training (PPT). During the EA phase, the objective is to match features from the same samples to facilitate collaborative training. In practice, EA typically utilizes Private Set Intersection (PSI) methods to identify shared sample IDs while keeping the unaligned dataset private. Once the alignment process is complete, the participating parties can begin the PPT phase using the aligned samples. PPT most commonly employs gradient descent techniques [19]. In this approach, instead of sharing their local data, the involved parties send their local model outputs and the corresponding gradients. The attacks described in this work occur during the training phase and a more comprehensive description of the vFL training procedure is provided in Section III.

C. Data Reconstruction Attacks on Neural Networks in vFL

In recent years, several feature inference attacks have been introduced to deduce features from neural networks. These attacks typically focus on scenarios in which the active party acts as an attacker seeking to infer the private features of passive parties. An attacker may or may not know the passive party's model architecture and parameters, which, correspond to the *white-box* and *black-box* settings, respectively. White-box settings primarily employ two methods: Model Inversion (MI) and Gradient Inversion (GI). White-box MI methods [8], [9], [23] generally optimize for inferred features that yield model outputs closely resembling the actual outputs. In contrast, Jin et al. present CAFE [10], a GI attack that aims to find the feature with gradients similar to the actual gradients. In the black-box setting, Ye et al. [11] proposed a Binary Feature Inference (BFI) attack to reconstruct binary features from passive parties, assuming that the local models are made up of just one fully-connected layer. In black-Box MI attacks [8], [23], an attacker first trains a shadow model $\hat{f}_i$ mimicking the local model f_i. Then, the attacker substitutes f_i for $\hat{f}_i$ and carries out the attack as in the white-box setting. When the attacker is allowed to query the passive parties, He et al. in [8] demonstrate that a direct inversion model g' can be learned to reconstruct the input features from the intermediate features. Table I provides a summary of existing attacks. Note that each of these methods only *observe* the gradients or the intermediate signals of vFL without maliciously changing them to breach privacy; thus, they belong to the class of passive attacks.

TABLE I: Data Reconstruction Attacks on Neural Networks in vFL. Auxiliary Requirements for the attacks is denoted as Aux. Req., Binary Features is denoted as Bin. Feat., Auxiliary Training Data is denoted as Aux. Data, and Data Dist. indicates the need for knowledge about the data distribution.

Attack	Setting	Type	Aux. Req.
CAFE [10]	White-box	Passive	–
White-Box MI [8], [9], [23]	White-box	Passive	–
Black-Box MI [8]	Black-box	Passive	Aux. Data
BFI [11]	Black-box	Passive	Bin. Feat.
AIN (Ours)	Black-box	Active	Aux. Data
AGN (Ours)	Black-box	Active	Data Dist.

D. FL with Local Differential Privacy and Other Defenses

Local Differential Privacy (LDP) [24], [25] is a privacy-enhancing technique designed to protect individuals' sensitive information while still extracting useful insights from the data. LDP operates in a decentralized manner, where each data point is perturbed with carefully crafted random noise before it is shared. As such, LDP has been recognized as a solution to alleviate the privacy risks associated with gradient sharing in FL systems. ε-LDP conditioned on an algorithm is defined as:

Definition 1. *[ε-LDP]: A randomized algorithm $\mathcal{M}$ fulfills ε-LDP, if for any two inputs x and x', and for all possible outputs $\mathcal{O} \in \text{Range}(\mathcal{M})$, we have: $Pr[\mathcal{M}(x) = \mathcal{O}] \leq e^{\varepsilon} Pr[\mathcal{M}(x') = \mathcal{O}]$, where ε is a privacy budget. We use the notation $\mathcal{M}^{\varepsilon}$ to refer to an algorithm that satisfies ε-LDP.*

Intuitively, a smaller privacy budget ε implies a higher privacy guarantee and a lower model performance as the distortion between $\mathcal{M}^{\varepsilon}(x)$ and x is higher.

However, specialized defense strategies have also been introduced to more efficiently thwart specific privacy attacks. Jin et al. introduce Random Fake Gradients [10] as a defense against CAFE that replaces the true gradients with randomly generated ones. Ye et al.'s Masquerade Defense [11] defeats BFI attacks by guiding attackers to prioritize randomly generated features instead of the true features.

E. Additional Contributions

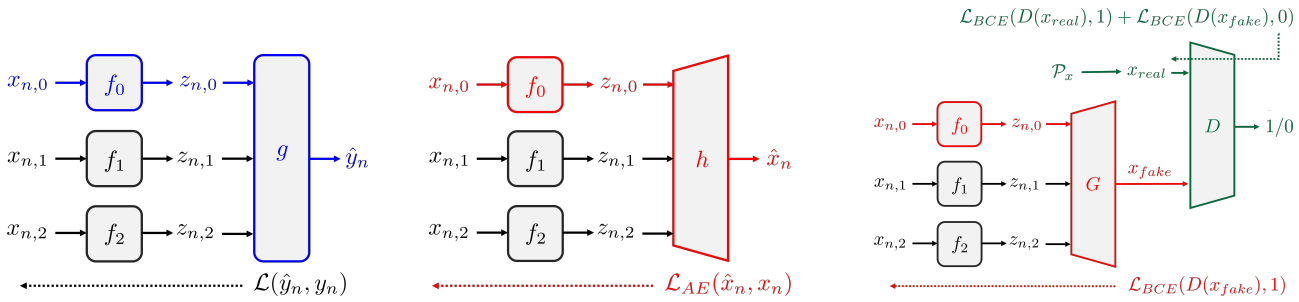
Vu et al. introduced the foundational concepts and initial attack models for the Active Inversion Network (AIN) and Active Generative Network (AGN) in vFL [26], primarily within a two-party system, as seen in [8]. This paper extends that work by adapting the two-party vFL framework to a more realistic multiparty setting, assessing the impact of active attacks across varying numbers of passive participants. Section V demonstrates that these attacks remain highly effective across three real-world datasets. Additionally, we expand the discussion on defenses by conducting an experimental analysis of Local Differential Privacy (LDP) via BitRand [27], examining its trade-offs between model performance and privacy across different tasks and applications.

III. Notations and Active Threat Models

Prior research primarily concentrated on scenarios in which attackers aimed to extract the victim's information while adhering to the system protocol. The server in this scenario is typically described as *honest-but-curious* or *semi-malicious*. However, this perspective does not fully encompass the potential vulnerabilities of the vFL system, as a participant can diverge from the protocol to mount more aggressive privacy attacks [12]–[14]. In this paper, we primarily focus on the training of neural networks within a vFL framework, where the complete model is divided among various parties. We specifically explore the concerns related to the *active* party, i.e., the one who owns the labels and has the capability to modify the training gradients (Figure 2a). Our investigation reveals that by manipulating these gradients, the malicious active party can coerce passive parties into training models capable of effectively inferring their private features. Consequently, this amplifies the attack's capabilities compared to passive scenarios. Our study focuses on the black-box setting where the attacker has no knowledge of the passive parties' models or parameters. We opt for the black-box setting to better illustrate the advantages active adversaries have over passive ones.

A. Protocol and Related Notations

We consider a vFL system where $K+1$ parties collaborate to train a neural network for a classification task. The training dataset $\mathcal{D} = \{(x_n, y_n)\}_{n=1}^{N}$ is stored locally without sharing, where n is the data index. Each party is associated with a unique feature set. In particular, a sample $x_n \in \mathcal{D}$ is



(a) Normal vFL (b) AIN with AE (c) AGN with C-GAN

Figure 2: Illustrations of a three-party vFL system under normal operation and our proposed attacks. The portions the active party controls are shown in colors. The forwarding and back-propagating directions are shown in solid and dashed arrows, respectively. The green and red in 2c represent two different training phases of the attacker. While the generator G is trained to recover the input, the discriminator D learns to differentiate generated inputs from original data.

partitioned into $[x_{n,0}, \cdots, x_{n,K+1}]$, where $x_{n,k}$ is the k-th partition of the n-th sample. The only participant that has the label, the *active party*, is associated with index 0. The other participants, the *passive parties*, are referred to by the remaining indices.

As illustrated in Figure 2a, since only the active party has the labels, it handles the aggregation of intermediate signal $z_{n,k}$ sent from passive users k. By denoting the encoder of user k by f_k, we can express the intermediate outputs as $z_{n,k} = f_k(x_{n,k})$, and the final output of the forwarding computation of the vFL model as:

$$\hat{y}_n = g\left([f_0(x_{n,0}), \cdots, f_{K+1}(x_{n,K+1})]\right) \tag{1}$$

where g denotes the aggregating model owned by the active party. In a normal training round, the loss on a local dataset $\mathcal{D}$ is given as:

$$\mathcal{L}(\theta; \mathcal{D}) = \frac{1}{N}\sum_{n=1}^{N} l\left(\theta; x_n, y_n\right) + \lambda \sum_{k=0}^{K} \gamma(\theta) \tag{2}$$

where θ, l, γ and λ are the vFL model's parameters, training loss, regularizing function, and the controlling hyperparameter for regularization, respectively. The active party updates the parameters of its encoder f_0 and the aggregation model g, denoted by θ_0 and θ_g, based on the gradients $\nabla_{\theta_0}\mathcal{L}(\theta; \mathcal{D})$ and $\nabla_{\theta_g}\mathcal{L}(\theta; \mathcal{D})$, respectively. The shorthand is written as $\nabla_{\theta_0}\mathcal{L}$ and $\nabla_{\theta_g}\mathcal{L}$, respectively. The final activity during one iteration of training is sending the gradients $\nabla_{\theta_k}\mathcal{L}$ from the active party to the passive user k so that the user can locally update θ_k without sharing its parameters and data.

B. Settings of Active Attacks

In active attacks, the active party can tamper with the distributed gradients to manipulate the models of the passive parties. For instance, as depicted in Figures 2b and 2c, our proposed attacks use an auto-encoder (AE) reconstruction loss $\mathcal{L}_{AE}$ and a generative binary cross-entropy loss $\mathcal{L}_{BCE}$ to compute the gradients sent to other parties. As described in both figures, the active attacks can also modify the later components of the vFL models to improve their ability to infer private information. In each scenario, the passive parties lack knowledge of the encoder f_0 of the active user and the later layers of the training model so, they have no clear means of detecting whether the gradients have been tampered.

We consider two attackers in two different scenarios. In the first scenario, the attacker knows a subset of the training data, denoted as auxiliary data $\mathcal{D}_{\text{AUX}} \subset \mathcal{D}$. This can also be considered as the case of the *limited-query* threat model [8] where the active party can send inference queries toward the passive parties. In that context, the number of inference queries is equivalent to the size of $\mathcal{D}_{\text{AUX}}$. This scenario suggests the attacker knows both $x_{n,k}$ and $f_k(x_{n,k})$ for x_n in the auxiliary data $\mathcal{D}_{\text{AUX}}$ and only knows $f_k(x_{n,k})$ for x_n in the rest of the data $\mathcal{D} \setminus \mathcal{D}_{\text{AUX}}$. From the practical perspective, the assumption depicts the situation when the active party collaboratively trains a model with participants who possess exclusive values of a set of specialized features. While only the local owners have the values of those features; the active party might have those values for some samples used during training. An example of this scenario is when a central hospital collaboratively develops a deep learning model with a specialized healthcare center. The inputs of the model are patients' records with some features only accessible by the specialized healthcare center. However, the central hospital might have the values of those features for some records before training.

In the second scenario, we consider an attacker that **only** knows the data distribution ($|\mathcal{D}_{\text{AUX}}| = 0$). This model applies to real-world scenarios where the active party has a sufficient amount of data to represent the feature distribution [28] or where the attacker can sample data directly from the data distribution. For instance, this situation is similar to the earlier hospital example where a central hospital maintains a comprehensive database of patient records containing all the features used by the model. Even if these data samples are not utilized in the vFL training process, the attack considered in this scenario can be executed by the central hospital as long as the training data stems from the same distribution.

Although our threat models are applicable to vFL systems with more than two participants, we adhere to the approach outlined in [8] and focus on two-party systems, consisting of one active party and one passive party. Expanding these attacks to multi-party systems can be accomplished by treating

all passive parties as a single entity. The key distinction in this generalization lies in the architecture of the combined models among the passive parties. For instance, in a multi-party setting, the model f_2 does not have access to features such as $x_{n,1}$. We do, however, provide an experiment involving a multi-party system as a preliminary result in Section V. The result indicates that there are minimal differences among settings (Table IV).

IV. Active Data Reconstruction Attacks

This section provides the descriptions for our proposed attacks. They are designed for cases in which the adversarial active party does not know the architecture and the parameters of local models f_k. Liu et al. [29] theoretically showed that, without additional information, the private features $x_{n,k}$ cannot be exactly recovered and established the fundamental challenge of inference attacks on black-box models. From a practical viewpoint, not knowing the local models prevents the attacker from computing the intermediate signals z_n; thus, thwarting MI and GI attacks from solving for features resulting in z_n or gradients based on z_n. However, our proposed attacks depend on an under-explored capability of the active party in vFL. In our attacks, the active party has control over the gradients sent toward the passive users. As discussed in Section III-B, both of our attacks operate on a minimal assumption regarding information about the private training data. Another key difference in our proposed active attacks, compared to previous passive attacks, is the inclusion of the passive users' local models. While passive attacks consider the local models determined, our attacks exploit the gradients to manipulate them to enhance privacy breaches.

A. Active Inversion Network

Our first attack includes an *Active Inversion Network* (AIN) aiming to train an inverse network h to recover private features (Figure 2b). The attacker manipulates the returned gradients, thereby deceiving passive users into jointly training the adversarial inverse network. Specifically, the attacker leverages a batch of auxiliary data points known before the training and utilizes a loss function such as Mean-Squared-Error (MSE), to compute corresponding gradients. These gradients are then transmitted to passive users. Upon updating their local parameters with these gradients, the passive users' local models undergo training for the adversarial reconstruction task. As a result, intermediate signals $z_{n,k}$ produced by these models post-adversarial training become more effective in reconstructing private input features.

Training the Active Inversion Network. Algorithm 1 shows the pseudocode of the AIN when given the intermediate signal from passive users during one training iteration of vFL, which consists of three main tasks: data filtering for the computation of the reconstruction loss (lines 2-8), updating the attacker's local parameters (lines 9-10), and sending the manipulated gradients to the passive users (line 12). The adversarial reconstruction loss to train the inverse network is:

$$\mathcal{L}_{AE}(\hat{X}, X) = \frac{1}{|X|} \sum_{n=1}^{|X|} \|\hat{x}_n - x_n\|^2 \tag{3}$$

Algorithm 1 AIN Training Step

Input: Intermediate activations $\{z_{n,k}\}_{n=1,k=1}^{n=B,k=K+1}$
Parameter: Reconstruction loss $\mathcal{L}_{AE}$, learning rate μ
Given: Intermediate activations $\{z_{n,0}\}_{n=1}^{B}$, x_n if $x_n \in \mathcal{D}_{\text{AUX}}$
Output: None

1: $c = 0$, $\hat{X}$ = empty array, X = empty array
   \\ *Filtering training batch B in $\mathcal{D}_{\text{AUX}}$*
2: **for** $n = 1$ to B **do**
3: **if** $x_n \in \mathcal{D}_{\text{AUX}}$ **then**
4: $c = c + 1$
5: $\hat{x}_n = h([z_{n,0}, \cdots, z_{n,K+1}])$
6: $X[c] = x_n$, $\hat{X}[c] = \hat{x}_n$
7: **end if**
8: **end for**
   \\ *Updating active party's parameters*
9: $\theta_h^{(t+1)} = \theta_h^{(t)} - \mu \nabla_{\theta_h^{(t)}} \mathcal{L}_{AE}(\hat{X}, X)$
10: $\theta_{f_0}^{(t+1)} = \theta_{f_0}^{(t)} - \mu \nabla_{\theta_{f_0}^{(t)}} \mathcal{L}_{AE}(\hat{X}, X)$
   \\ *Computing and sending adversarial gradients*
11: **for** user $k = 1$ to $K + 1$ **do**
12: Send $\nabla_{z_{n,k}} \mathcal{L}_{AE}(\hat{X}, X)$ to user k
13: **end for**

where X is the set of inputs that appear in both training and auxiliary batches of data. $\hat{X}$ is the output of h on the intermediate signal z_n resulting from those $x_n \in X$. With that, the AIN training can be formalized as:

$$\text{Active User: } \arg\min_{\theta_0, h} \mathcal{L}_{AE}(\hat{X}, X) \tag{4}$$

$$\text{Passive User } k\text{: } \arg\min_{\theta_k} \mathcal{L}_{AE}(\hat{X}, X) \tag{5}$$

While the local updating serves as the attacker's optimization (4), the gradient-sending is for the passive parties optimization (5). The gradients of the passive parties are given by:

$$\nabla_{\theta_k} \mathcal{L}_{AE} = \frac{\partial \mathcal{L}_{AE}}{\partial \theta_k} = \sum_n \frac{\partial \mathcal{L}_{AE}}{\partial z_{n,k}} \frac{\partial z_{n,k}}{\partial \theta_k} \tag{6}$$

Reconstructing Data with the Active Inversion Network. When the training of h is complete, the active party is able to recover any private inputs from the intermediate signals, given that it can compute $x_n \approx h(z_n)$. Note that the more data in $\mathcal{D}_{\text{AUX}}$, the better the reconstruction capability and the stronger the attack. As the access to $\mathcal{D}_{\text{AUX}}$ of the active user is hardly ever noticed in practice, our AIN-based attack signifies the importance in designing a more secure vFL protocol.

B. Active Generative Network

Our second attack, comprises an *Active Generative Network* (AGN), tailored for scenarios where no auxiliary data is available. The AGN draws inspiration from Generative Adversarial Networks (GANs) [30], comprising of a *generator* and a *discriminator*. In a GAN setup, the generator produces fake data samples mimicking real samples from the distribution. The discriminator aims to distinguish between the *real* data samples from the distribution and the *fake* samples generated by the generator. However, GANs typically generate samples

Algorithm 2 AGN Training Step

Input: Intermediate activations $\{z_{n,k}\}_{n=1,k=1}^{n=B,k=K+1}$
Parameter: BCE loss $\mathcal{L}_{BCE}$ (as specified in Eq. 7 and 8), discriminator learning rate μ_D, generator learning rate μ_G,
Given: Input features $\{x_{n,0}\}_{n=1}^{B}$ and intermediate activations $\{z_{n,0}\}_{n=1}^{B}$
Output: None

1: X_{real} = empty array, X_{fake} = empty array
\\ *Generating training batch B for discriminator D*
2: **for** $n = 1$ to B **do**
3: Sample x_{real} from $\mathcal{P}_x$
4: $x_{fake} = G([z_{n,0}, \cdots, z_{n,K+1}])$
5: $x_{fake,0} = x_{n,0}$
6: Include x_{real} to X_{real} and x_{fake} to X_{fake}
7: **end for**
\\ *Updating the adversarial discriminator D*
8: $\theta_D^{(t+1)} = \theta_D^{(t)} - \mu_D \mathcal{L}_{BCE}(D(X_{real}), 1) - \mu_D \mathcal{L}_{BCE}(D(X_{fake}), 0)$
\\ *Updating adversarial generator G*
9: $\theta_G^{(t+1)} = \theta_G^{(t)} - \mu_G \nabla_{\theta_G^{(t)}} \mathcal{L}_{BCE}(D(X_{fake}), 1)$
\\ *Computing and sending adversarial gradients*
10: **for** user $k = 1$ to $K + 1$ **do**
11: Send $\nabla_{z_{n,k}} \mathcal{L}_{BCE}(D(X_{fake}), 1)$ to user k
12: **end for**

via random noise and are not inherently designed for reconstruction or inference tasks. To address this limitation, the AGN takes advantage of the information known to the active party about the target sample, i.e., $x_{n,0}$ and potentially y_n, to guide the GAN generator in returning x_n rather than an arbitrary sample from the data distribution. In this regard, the AGN exhibits similarities with the conditional generative adversarial network (C-GAN) [31], a modified GAN that conditions data generation based on additional information.

Training the Active Generative Network. Algorithm 2 outlines the pseudocode for the AGN in a single vFL training iteration, comprising four key steps. First is the generation of the batch data to train the discriminator (lines 2-7). The real data is sampled from the known distribution $\mathcal{P}_x$ while the fake data is generated from the intermediate activations. To enforce the knowledge of the active party on the target sample x_n, the portion of the original sample $x_{n,0}$ that the active party knows is updated onto the fake sample (line 5). Note that our current implementation and algorithm only exploits $x_{n,0}$, not y_n. The second step is the update of the discriminator which is the same as in the standard GAN training. In particular, the discriminator is updated with the Binary Cross-Entropy (BCE) loss (line 8) whose description is:

$$\mathcal{L}_{BCE}(Y, 1) = \frac{1}{|X|} \sum_n \log(y_n) \tag{7}$$

$$\mathcal{L}_{BCE}(Y, 0) = \frac{1}{|X|} \sum_n \log(1 - y_n) \tag{8}$$

Next is updating the generator. In contrast to the discriminator which is trained to recognize the fake sample with the loss $\mathcal{L}_{BCE}(D(X_{fake}), 0)$, the generator aims to generate data from the distribution $\mathcal{P}_x$ with the loss $\mathcal{L}_{BCE}(D(X_{fake}), 1)$ (line 9). The final step of this attack is to send the manipulated gradients $\nabla_{z_{n,k}} \mathcal{L}_{BCE}(D(X_{fake}), 1)$ to the passive users. This makes the users jointly train the generator G.

Reconstructing Data with the Active Generative Network. During the inference phase, the attacker reconstructs the private data using the generator:

$$\hat{x}_n = G([z_{n,0}, \cdots, z_{n,K+1}]), \quad \hat{x}_{n,0} = x_{n,0}$$

where $[z_{n,0}, \cdots, z_{n,K+1}]$ are the received intermediate signals. This is also the computation of x_{fake} at lines 4 and 5 of Algorithm 2. As the generator is trained to produce $\hat{x}n$ that closely resembles the training data and its 0-th partition $\hat{x}n, 0$ aligns with that of the original input x_n, the generator tends to recover $\hat{x} \approx x$. Intuitively, the more $x_{n,0}$ are known by the attacker, the higher the likelihood that $\hat{x}_n$ becomes similar to x_n. We conduct an experiment to illustrate this claim in Section V (Table III).

The key advantage of the AGN compared to the AIN is that it does not need to know any features used for training, which makes the attack much stealthier. Given the swift expansion of real-world public data, assuming the active party can access or cheaply sample data from the distribution is increasingly realistic. As a result, the AGN-based attack serves as a direct illustration of real-world privacy threats.

V. Experiments

This section presents the results of our experiments, demonstrating the efficacy of our proposed attacks in reconstructing real-world private data within a vFL framework. Our primary objective is to underscore the inherent privacy threats an *active* attacker can attain by circumventing the vFL protocol.

A. Experimental Settings

We implement our experiments with Python 3.8. Each experiment is conducted on a single GPU-assisted compute node installed with a Linux 64-bit operating system. Our testbed resources include 36 CPU cores with 60GB of RAM and 2 threads per core. Our allocated node is also provisioned with 8 GPUs with 80GB of VRAM per GPU.

Datasets. Our experiments are conducted on three real-world datasets: MNIST [15] (hand-written image dataset), CIFAR10 [16] (image dataset), and USCensus taken from the UCI Machine Learning Repository [17] (tabular dataset). The two image datasets are commonly used in vision tasks while the USCensus contains a one percent sample with 68 features of the Public Use Microdata Samples records. The features of the image datasets are normalized into a range between 0 and 1. The 68 categorical features of the USCensus dataset are preprocessed into 396 binary features. Table II shows the general information of our experiments reported in this work. The D_{AUX} and D_{TRAIN} indicate the sizes of the auxiliary and training datasets for the respective attacking methods.

Simulating the Passive Parties. In practice, the model's components for passive users are chosen based on the applications at hand. In our evaluation, we use Multi-layer Perceptron (MLP) and Convolutional (Conv.) layers for the image datasets

TABLE II: Configurations and Parameters of our experiments.

Dataset	Dimension	D_{AUX}	D_{TRAIN}	Attacks	Victim Module (f_i)	Attacker Modules
MNIST	$1 \times 28 \times 28$	50-3000	N/A	MLP AIN	2-layer MLP encoder	2-layer MLP decoder
		50-3000	N/A	Conv. AIN	2-layer MLP encoder	2-layer CNN decoder
		N/A	50000	AGN	2-layer MLP encoder	4-layer CNN generator and 4-layer CNN discriminator
CIFAR10	$3 \times 32 \times 32$	50-2000	N/A	MLP AIN	3-layer CNN encoder	2-layer MLP decoder
		50-2000	N/A	Conv. AIN	3-layer CNN encoder	2-layer CNN decoder
		N/A	50000	AGN	1-layer MLP encoder	2-layer CNN generator and 4-layer CNN discriminator
USCensus	68	50-3000	N/A	AIN	3-layer MLP encoder	3-layer MLP decoder
		N/A	100000	AGN	1-layer MLP encoder	3-layer MLP generator and 3-layer MLP discriminator

and only MLPs for the tabular dataset. The hyperparameters of the layers are selected based on the common tasks conducted on the datasets: image classifications (MNIST/CIFAR10) and data clustering (USCensus). Table II also provides a brief description of the hyperparameter configurations.

Attacker Configurations. The model's components on the active party's side are directly under the control of the attacker in the threat models. Hence, it can select different architectures as well as fine-tune their parameters for better-reconstructed signals. In our AIN-based attack on MNIST and CIFAR10, we use two different architectures, namely MLP and Conv. in which the decoder's architectures use MLP and Conv. layers, respectively. All other experiments use MLPs.

Evaluation Metrics. We quantify our results in image datasets with the Peak Signal-to-Noise Ratio (PSNR), which measures the pixel level recovery quality of the image and is defined as:

$$PSNR(\hat{x}, x) = 20 \log_{10} (\max(x)/MSE)$$

where $\max_i(x)$ is the maximum possible value of pixels in the original image x and MSE is the reconstruction loss between the original image x and the recovered image $\hat{x}$. Since the features are categorical for the USCensus dataset, we also use accuracy to evaluate the attacking results. All reported results are obtained with at least 10 runs.

Benchmark. Although no prior work exists on *active* data reconstruction attacks in vFL, black-box attacks have also proven to be limited (Table I). Therefore, we compare our proposed attack methods with the black-box MI method in [8] and refer to it as the *benchmark* in our results. Compared to our methods, the benchmark has two main differences. First, it is a passive attack conducted during the inference phase instead of the training phase. Second, since the attack is during the inference phase, it does not involve the update and manipulation of the model's components of the passive parties. The configuration of the model architecture used for the benchmark is an MLP as in the original paper. Since models for tabular data are not provided in [8], our results on tabular data exclude the benchmark.

B. Experimental Results

General Performance. Figures 3 and 4 visually illustrate the effectiveness of our reconstruction attacks on MNIST and CIFAR10. For both figures, the top row indicates the target inference in the test set, and the remaining rows are recovered images from different attack methods. Since we consider a two-party vFL system, each party holds half of the images. In these experiments, we consider an active attacker that holds the lower half of the images and its goal is to reconstruct the upper half. Therefore, the resulting PSNRs are calculated only on the upper half. However, Figures 3 and 4 show the reconstructed images in their entirety for more intuitive visualizations.

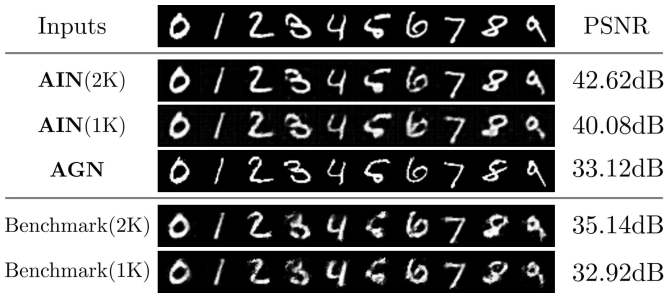


Figure 3: Reconstructed samples from MNIST. Our attack methods are highlighted in bold. The notations 1K and 2K refer to the size of the auxiliary dataset $\mathcal{D}_{\text{AUX}}$.

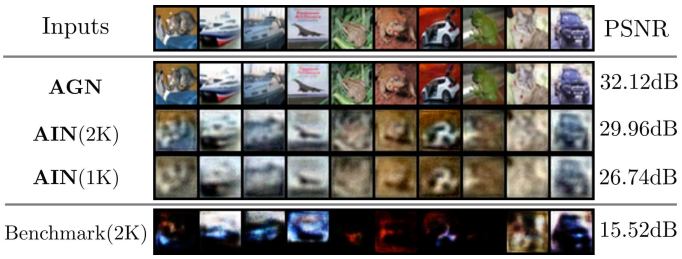


Figure 4: Reconstructed samples from CIFAR10. Our attack methods are highlighted in bold. The notations 1K and 2K refer to the size of the auxiliary dataset $\mathcal{D}_{\text{AUX}}$.

Our attack methods are highlighted in bold. For both datasets, the AIN refers to the Conv. AIN in which the decoder uses Convolutional layers for the decoder h (Figure 2b). We can see that our adversaries can accurately recover the images with competitive performance. While the AIN approach can be better than the AGN with just the knowledge of 1000 training samples in MNIST, the AGN approach proves to be much better in the more complex CIFAR10 dataset. It is also clear that, with the same amount of auxiliary data, the AIN-based attack performs significantly better than the benchmark in terms of PSNR. In fact, even with 2000 samples, the recovered CIFAR10 images of the passive benchmark are hardly recognizable. These results clearly demonstrate the gain of *active* attackers over *passive* ones.

Impact of the Size of the Auxiliary Dataset $\mathcal{D}_{\text{AUX}}$. Figure 5 provides thorough evaluations of different attacks with different sizes of auxiliary data. The results on image

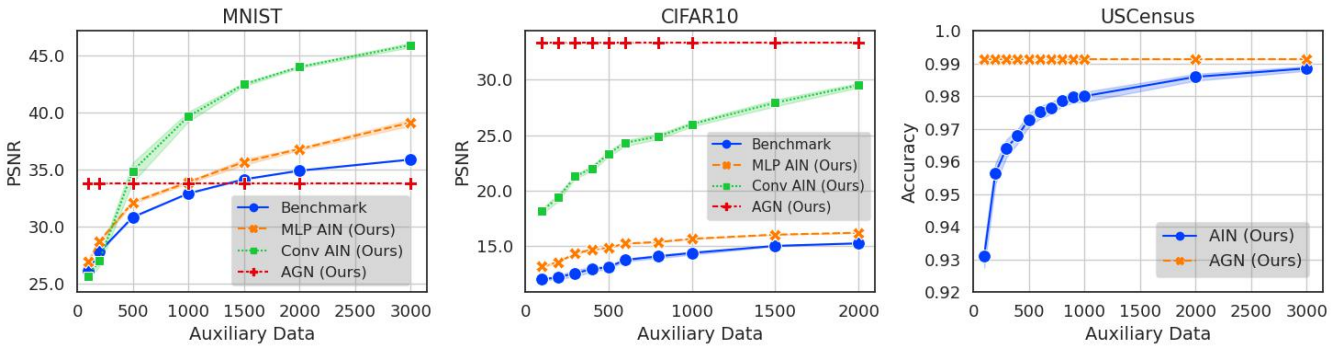


Figure 5: PSNR (dB) of reconstructed data samples in MNIST and CIFAR10 along with accuracy of reconstructed features in USCensus. The AGN-based attack does not require $\mathcal{D}_{\text{AUX}}$ so, its results are plotted in a straight line for convenient comparison.

TABLE III: Accuracy and PSNR (dB) of our AGN-based attack on the USCensus dataset with varying numbers of known features. The dataset contains a total of 68 features.

No. of Known Features	50	46	38	30	22	14
Accuracy	99.73 ± 0.10	99.36 ± 0.06	99.28 ± 0.24	98.69 ± 0.40	97.65 ± 0.55	95.90 ± 0.34
PSNR (dB)	52.55 ± 3.05	44.18 ± 0.70	43.52 ± 2.51	38.20 ± 2.31	33.00 ± 1.93	27.98 ± 0.65

datasets show that by manipulating the later layers of the training (Conv. AIN), the attacker can significantly improve the reconstructed signal. Notably, the AGN approach outperforms all methods in CIFAR10. This observation can be attributed to the increased complexity of CIFAR10 compared to MNIST, indicating that our AIN methods and the benchmark would require a larger number of samples to match the performance of the generative approach.

For the USCensus dataset, our attacks aim to infer at least half (34/68) of the private features of the input samples. While the AGN can consistently achieve $\approx 99\%$ of inference, the AIN needs ≈ 3000 data samples to reach the same performance. Since there is no available passive attack for tabular data, we cannot include the passive benchmark. However, the results have demonstrated that our methods work competitively for both image and tabular data.

Impact of the Number of Known Features. The more features the attacker knows, the easier to infer the remaining private features. In fact, Table III reports the results of our AGN-based attack on the USCensus dataset with different numbers of known features and reflects the above statement. The table includes both the accuracy and the PSNR of the reconstructed features for better illustration. The PSNR is computed by considering the tabular features as vectors with components between 0 and 1. Note that our AIN-based attack does not rely on the known features, thus the analysis is only meaningful for our second attack comprised of the AGN. It is important to note that with just 14 features, our AGN approach is capable of inferring the remaining 54 features with almost 96% accuracy. This observation suggests that even limiting the number of known features might not be an effective approach to mitigate this *active* attack.

Impact of the Number of Passive Parties. When the number of parties increases, the neurons of different parties are more disconnected. This means that, compared to the case of the single passive party, the multi-party vFL settings do not have trainable weights among neurons belonging to different parties. Analyzing the impact of this modification on the attack is not trivial. On one hand, it may decrease the expressive power of the AIN approach due to the absence of cross-user weights. On the other hand, it may also help the AIN-based attack avoid over-fitting as there is a reduction in the number of total parameters.

TABLE IV: Accuracy and PSNR (dB) of our attacks on the USCensus dataset varying different numbers of passive users.

	No. of Passive Users	**1**	**2**	**4**
AGN	Accuracy	98.81	98.30	98.11
	PSNR (dB)	38.96	35.73	34.86
AIN	Accuracy	95.69	86.42	86.54
	PSNR (dB)	27.17	26.87	28.52

We implement three vFL systems using the USCensus dataset with 1, 2, and 4 passive users to heuristically study the impact of the number of passive parties on both of our attacks. The number of features owned by each user in those systems is 40, 20, and 10, respectively. Each user's module f_i is a 1-layer MLP. Table IV reports the PSNR and the accuracy of our AGN and AIN-based attacks with the described configurations. We can see that there is no clear indicator of which number of users is more susceptible to the attacks even though 1 passive user seems to result in a higher reconstructed accuracy. One meaningful observation is that changing the number of participants might not help to prevent active data reconstruction attacks.

VI. Discussion of Defenses

This section discusses possible defense strategies to mitigate the attacks studied in this paper.

Deeper Client Modules or Specific Client's Architectures? Previous studies on passive attacks recommend increasing the depth of client-side modules and running fully connected layers before sending out results [8]. These recommendations

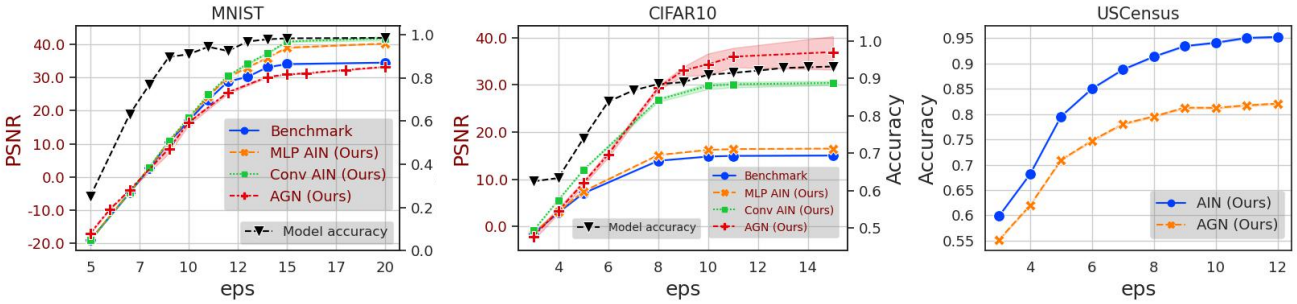


Figure 6: Accuracy and PSNR (dB) of reconstructed data samples from MNIST and CIFAR10 along with accuracy of reconstructed features from the USCensus dataset under the BitRand [27] LDP-mechanism.

are primarily tailored for passive settings, assuming that the active party cannot manipulate the weights of the client's modules. However, in *active* settings, where these modules participate in adversarial inference during training, these modifications provide limited protection. For example, the attacker can counteract a deeper client module by reducing the depth of their own components. Additionally, deeper networks increase the computational demands on the client-side. In practice, the client or device might lack the necessary computational power, storage capacity, or battery resources to implement these recommendations effectively.

Avoid Public Data During Training? Given that our AIN-based attack exploits public data used during the training phase, one apparent strategy to counter these attacks is to abstain from using public data altogether. Conversely, the AGN approach relies on data distribution, and in certain specialized circumstances, offering minimal information about the data might enhance privacy. Nonetheless, it is important to note that neither of these approaches can be considered entirely reliable because information about private features and data distribution can still be obtained through side channels.

Defense with Local Differential Privacy. Acknowledged as a solution to address the privacy concerns arising from the sharing of gradients, LDP operates on the principle of *randomized response* [32]. Figure 6 reports our experiments evaluating our attacks when BitRand [27] is applied to the input data.

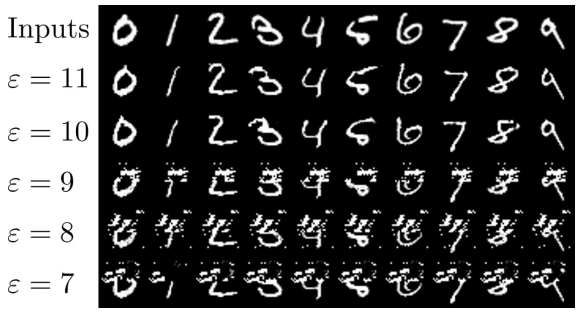


Figure 7: Samples of reconstructed data by the AGN attack on the MNIST dataset with different privacy budgets ε. Note that the attacker has the bottom half of the input.

For the image datasets, we provide the black curves indicating the test set accuracy of a neural network trained in the classification tasks with respect to different privacy budgets ε. From the model's performance perspective, we can see that the model performance in MNIST begins to drop when $\varepsilon = 10$, and significantly drops when $\varepsilon < 8$. From the attacker's point of view, we provide Figure 7 showing the reconstructed signal of our AGN method for MNIST samples. We can see that only when $\varepsilon \leq 9$, the attack starts to fail to reconstruct the inputs. Therefore, the result of MNIST demonstrates clearly the trade-off between model performance and data privacy. We can also observe that trade-off in experiments of CIFAR10. For example, if we want the model to maintain an accuracy $> 90\%$, the privacy budget ε needs to be about 9. However, around that range, the PSNR of the recovered signal is about 30 dB, at which, the reconstructed features are blurry, but still recognizable (Figure 4). Since the USCensus dataset does not have the labels for the classification task, we simply report the accuracy of our attacks in Figure 6. The results indicate the performance of our attacks in LDP-protected tabular data is not that much different from how they perform in image data. Particularly, the attack performance starts to drop significantly around $\varepsilon \approx 8$.

The takeaway here is that implementing LDP-protecting mechanisms can provide some defense against our proposed attack. Nevertheless, it is essential to recognize that there exists an obvious trade-off between performance and privacy. Determining the appropriate privacy budget requires additional examination and assessment based on the specific applications. It is also essential to be aware that the usage of LDP-protecting mechanisms might significantly burden the passive participant's devices in terms of computational complexity and energy consumption.

VII. Conclusion

Our work exposes critical privacy vulnerabilities in vertical Federated Learning (vFL), demonstrating how an active party can manipulate gradients during local training to reconstruct private data. We introduce two novel attack methods—one leveraging a small set of known training samples and the other exploiting the underlying data distribution—both of which reveal significant privacy risks in vFL systems. These findings highlight the urgent need for strengthening existing defenses, such as Local Differential Privacy (LDP), while also addressing the inherent trade-off between model performance and privacy. Beyond identifying these security challenges, our

study provides practical insights for improving vFL protocols to mitigate adversarial threats. By encouraging further research into adaptive privacy mechanisms and strategies for balancing performance, privacy, and security, this work aims to guide the development of more resilient vFL systems for secure deployment in real-world applications.

REFERENCES

[1] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Trans. Intell. Syst. Technol.*, vol. 10, no. 2, jan 2019. [Online]. Available: https://doi.org/10.1145/3298981

[2] Q. Yang, Y. Liu, Y. Cheng, Y. Kang, T. Chen, and H. Yu, *Vertical Federated Learning*. Cham: Springer International Publishing, 2020, pp. 69–81. [Online]. Available: https://doi.org/10.1007/978-3-031-01585-4_5

[3] G. Long, Y. Tan, J. Jiang, and C. Zhang, *Federated Learning for Open Banking*. Cham: Springer International Publishing, 2020, pp. 240–254. [Online]. Available: https://doi.org/10.1007/978-3-030-63076-8-17

[4] Z. Liu, Y. Chen, H. Yu, Y. Liu, and L. Cui, "Gtg-shapley: Efficient and accurate participant contribution evaluation in federated learning," *ACM Trans. Intell. Syst. Technol.*, vol. 13, no. 4, may 2022. [Online]. Available: https://doi.org/10.1145/3501811

[5] D. Cha, M. Sung, and Y.-R. Park, "Implementing vertical federated learning using autoencoders: Practical application, generalizability, and utility study," *JMIR Medical Informatics*, vol. 9, p. e26598, 06 2021.

[6] C. Fu, X. Zhang, S. Ji, J. Chen, J. Wu, S. Guo, J. Zhou, A. X. Liu, and T. Wang, "Label inference attacks against vertical federated learning," in *31st USENIX Security Symposium (USENIX Security 22)*. Boston, MA: USENIX Association, Aug. 2022, pp. 1397–1414.

[7] T. Zou, Y. Liu, Y. Kang, W. Liu, Y. He, Z. Yi, Q. Yang, and Y.-Q. Zhang, "Defending batch-level label inference and replacement attacks in vertical federated learning," *IEEE Transactions on Big Data*, 2022.

[8] Z. He, T. Zhang, and R. B. Lee, "Model inversion attacks against collaborative inference," in *Proceedings of the 35th Annual Computer Security Applications Conference*, ser. ACSAC '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 148–162. [Online]. Available: https://doi.org/10.1145/3359789.3359824

[9] X. Luo, Y. Wu, X. Xiao, and B. C. Ooi, "Feature inference attack on model predictions in vertical federated learning," in *2021 IEEE 37th International Conference on Data Engineering (ICDE)*, 2021.

[10] X. Jin, P.-Y. Chen, C.-Y. Hsu, C.-M. Yu, and T. Chen, "Cafe: Catastrophic data leakage in vertical federated learning," in *Advances in Neural Information Processing Systems*, M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, Eds., vol. 34. Curran Associates, Inc., 2021, pp. 994–1006.

[11] P. Ye, Z. Jiang, W. Wang, B. Li, and B. Li, "Feature reconstruction attacks and countermeasures of dnn training in vertical federated learning," *arXiv preprint arXiv:2210.06771*, 2022.

[12] F. Boenisch, A. Dziedzic, R. Schuster, A. S. Shamsabadi, I. Shumailov, and N. Papernot, "When the curious abandon honesty: Federated learning is not private," in *2023 IEEE 8th European Symposium on Security and Privacy (EuroS&P)*. IEEE, 2023, pp. 175–199.

[13] T. Nguyen, P. Lai, K. Tran, N. Phan, and M. T. Thai, "Active membership inference attack under local differential privacy in federated learning," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2023, pp. 5714–5730.

[14] L. H. Fowl, J. Geiping, W. Czaja, M. Goldblum, and T. Goldstein, "Robbing the fed: Directly obtaining private data in federated learning with modified models," in *International Conference on Learning Representations*, 2021.

[15] Y. LeCun and C. Cortes, "MNIST handwritten digit database," http://yann.lecun.com/exdb/mnist/, 2010. [Online]. Available: http://yann.lecun.com/exdb/mnist/

[16] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009.

[17] M. Meek, T. Thiesson, and H. Heckerman, "US Census Data (1990)," UCI Machine Learning Repository, DOI: https://doi.org/10.24432/C5VP42.

[18] H. B. McMahan, E. Moore, D. Ramage, and B. A. y Arcas, "Federated learning of deep networks using model averaging," *CoRR*, vol. abs/1602.05629, 2016. [Online]. Available: http://arxiv.org/abs/1602.05629

[19] L. Wan, W. K. Ng, S. Han, and V. C. S. Lee, "Privacy-preservation for gradient descent methods," in *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '07. New York, NY, USA: Association for Computing Machinery, 2007, p. 775–783. [Online]. Available: https://doi.org/10.1145/1281192.1281275

[20] Q. Li, Z. Wen, Z. Wu, S. Hu, N. Wang, Y. Li, X. Liu, and B. He, "A survey on federated learning systems: Vision, hype and reality for data privacy and protection," *IEEE Transactions on Knowledge & Data Engineering*, vol. 35, no. 04, pp. 3347–3366, apr 2023.

[21] L. Li, Y. Fan, M. Tse, and K.-Y. Lin, "A review of applications in federated learning," *Computers & Industrial Engineering*, vol. 149, p. 106854, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0360835220305532

[22] Y. Liu, T. Fan, T. Chen, Q. Xu, and Q. Yang, "Fate: An industrial grade platform for collaborative learning with data protection," *Journal of Machine Learning Research*, vol. 22, no. 226, pp. 1–6, 2021. [Online]. Available: http://jmlr.org/papers/v22/20-815.html

[23] X. Jiang, X. Zhou, and J. Grossklags, "Comprehensive analysis of privacy leakage in vertical federated learning during prediction." *Proc. Priv. Enhancing Technol.*, vol. 2022, no. 2, pp. 263–281, 2022.

[24] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *TCC*, 2006, pp. 265–284.

[25] U. Erlingsson, V. Pihur, and A. Korolova, "Rappor: Randomized aggregatable privacy-preserving ordinal response," in *Proceedings of the 2014 ACM SIGSAC CCS*, 2014, pp. 1054–1067.

[26] M. N. Vu, J. Tre'R, R. Alharbi, and M. T. Thai, "Active data reconstruction attacks in vertical federated learning," in *2023 IEEE International Conference on Big Data (BigData)*. IEEE, 2023, pp. 1374–1379.

[27] P. Lai, H. Phan, L. Xiong, K. Tran, M. Thai, T. Sun, F. Dernoncourt, J. Gu, N. Barmpalios, and R. Jain, "Bit-aware randomized response for local differential privacy in federated learning," 2022.

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

[29] Y. Liu, X. Zhang, Y. Kang, L. Li, T. Chen, M. Hong, and Q. Yang, "Fedbcd: A communication-efficient collaborative learning framework for distributed features," *IEEE Transactions on Signal Processing*, vol. 70, pp. 4277–4290, 2022.

[30] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *Advances in neural information processing systems*, vol. 27, 2014.

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

[32] S. L. Warner, "Randomized response: A survey technique for eliminating evasive answer bias," *Journal of the American Statistical Association*, vol. 60, no. 309, pp. 63–69, 1965.

Minh Vu received his Ph.D. from the University of Florida in 2023 under Dr. My T. Thai. His research focuses on Explainable Machine Learning. Before that, he received a B.Eng. degree (Hons.) in electronics and telecommunications from the Hanoi University of Science and Technology, Hanoi, Vietnam, and an M.Sc. degree in electrical and computer engineering from The University of Akron, Akron, OH, USA, in 2016 and 2018, respectively. His research interests include explainable AI, cybersecurity, optimization, information theory, and wireless communications.

Tre' R. Jeter is a Ph.D. Candidate under Dr. My T. Thai and has been a student at the University of Florida (UF) since the Fall of 2021. He received his Masters in Computer Science from UF in the Spring of 2024. Before this, he received both of his Bachelors of Science in Computer Science and Computer Engineering from Claflin University in April of 2021. His research interests lie at the intersection of security, privacy, and application of AI, specifically in the realm of Federated Learning.

Raed Alharbi received his Ph.D. from the University of Florida (UF) in 2023 under Dr. My T. Thai. He received his Bachelor's degree of Engineering in Computer and Information Systems from Taibah University and his Master's degree in Computer Science from UF. He is currently a Research Assistant Professor in the Department of Computer Science at the Saudi Electronic University. His research interest include explainable AI and security of ML applications in social media.

Jung Taek Seo received his M.S. degree in computer engineering from Ajou University, South Korea, in 2001, and his Ph.D. degree in information security engineering from Korea University, South Korea, in 2006. He is currently a Professor with the Department of Smart Security at Gachon University. His research interests include CPS security, ICS cyber security, smart grid security, nuclear power plant security, smart factory security, smart city security, and automotive cyber security.

My T. Thai is a Research Foundation Professor of Computer & Information Sciences & Engineering and Associate Director of Nelms Institute for the Connected World at the University of Florida. Dr. Thai has extensive expertise in trustworthy AI, billion-scale data mining, and optimization, especially for complex graph data with applications to blockchain, social media, critical networking infrastructure, cybersecurity, and healthcare. The results of her work have led to 7 books and 300+ publications in leading academic journals and conferences, including 2014 IEEE MSN Best Paper Award, 2017 IEEE ICDM Best Papers Award, 2018 IEEE/ACM ASONAM Best Paper Runner Up, 2023 ACM Web Science Test-of-Time Award, and 2023 AAAI Distinguished Paper Award.

In 2009, Dr. Thai was awarded the Young Investigator (YIP) from the Defense Threat Reduction Agency (DTRA) and in 2010, she won the NSF CAREER Award. She was also awarded the UF Research Foundation Professorship in 2016. Dr. Thai is a Fellow of the IEEE.

Dr. Thai has engaged in many professional activities, including being TPC-chairs of many IEEE international conferences and on the editorial board of several journals. She is presently the Editor-in-Chief of Journal of Combinatorial Optimization (JOCO) journal. She is a book series editor of Springer Optimization and its Application.