Cooperative Decentralized Backdoor Attacks on Vertical Federated Learning

Seohyun Lee[ID], *Student Member, IEEE*, Wenzhi Fang[ID], *Graduate Student Member, IEEE*, Anindya Bijoy Das[ID], *Member, IEEE*, Seyyedali Hosseinalipour[ID], *Senior Member, IEEE*, David J. Love[ID], *Fellow, IEEE*, and Christopher G. Brinton[ID], *Senior Member, IEEE*

Abstract—**Federated learning (FL) is vulnerable to backdoor attacks, where adversaries alter model behavior on target classification labels by embedding triggers into data samples. While these attacks have received considerable attention in horizontal FL, they are less understood for vertical FL (VFL), where devices hold different features of the samples, and only the server holds the labels. In this work, we propose a novel backdoor attack on VFL which (i) does not rely on gradient information from the server and (ii) considers potential collusion among multiple adversaries for sample selection and trigger embedding. Our label inference model augments variational autoencoders with metric learning, which adversaries can train locally. A consensus process over the adversary graph topology determines which datapoints to poison. We further propose methods for trigger splitting across the adversaries, with an intensity-based implantation scheme skewing the server towards the trigger. Our convergence analysis reveals the impact of backdoor perturbations on VFL indicated by a stationarity gap for the trained model, which we verify empirically as well. We conduct experiments comparing our attack with recent backdoor VFL approaches, finding that ours obtains significantly higher success rates for the same main task performance despite not using server information. Additionally, our results verify the impact of collusion on attack performance.**

Index Terms—**Vertical federated learning (VFL), variational autoencoder (VAE), metric learning, backdoor attack, privacy.**

I. INTRODUCTION

FEDERATED Learning (FL) [1] has emerged as a popular method for collaboratively training machine learning models across edge devices. By eliminating the need for communication of raw data across the network, FL proves especially valuable in scenarios where data privacy is critical. However, the decentralized nature of FL introduces new security challenges, as individual devices may lack the robust

security measures of a centralized system, thereby increasing the risk of adversarial attacks that can compromise the integrity of training.

The two prevalent frameworks of FL, Horizontal Federated Learning (HFL) and Vertical Federated Learning (VFL), both face significant vulnerabilities to adversaries [2]. In HFL, the training data is partitioned by sample data points, with each device holding different subsets of the overall dataset. Most of the existing literature on adversarial attacks in FL has concentrated on HFL, with the goal to tackle vulnerabilities such as data poisoning attacks [3], [4], model inversion attacks [5], [6], and backdoor attacks [7]. Conversely, VFL [8], [9] involves local devices that share the same samples but hold different features of the samples. In this setup, one node, referred to as the active party or server, holds the labels and oversees the aggregation process, while the other devices function as passive parties or clients, constructing local feature embeddings and periodically passing them to the server. For instance, in a wireless sensor network (WSN) [10], each sensor may collect readings from its local environment (e.g., video feeds) which collectively form a full sample for the fusion center's learning task (e.g., object detection) at a point in time. Recent research has begun to study the impact of attacks on VFL, including feature inference attacks [11], [12], label inference attacks [13], and attribute inference attacks [14]. In this paper, we focus on *backdoor attacks* [7] for VFL. A backdoor attack aims to alter the behavior of an FL model on a particular label (called the *target*) when the model encounters data samples for the label that an adversary has implanted with an imperceptible trigger pattern. Addressing backdoor attacks is crucial in both HFL and VFL because these attacks can lead to severe security breaches without easily detectable impacts on overall model performance [15]. For example, in the WSN object detection use-case, a backdoor adversary could implant triggers in a sensor's local sample view of a car to force the system to misclassify the entire sample as a truck.

In this paper, we consider an underexplored scenario where there are multiple adversaries, and these adversaries have the capability of colluding over a graph topology to execute a coordinated backdoor attack. In this setting, the adversaries gain control of a set of client nodes and establish an ad-hoc local network among themselves for cooperation. Prominent examples can be found in defense settings. For instance, in a contested region, attackers could take control of a few scattered military assets responsible for e.g., communicating front-line

conditions, such as drones or tactical mobile devices [16]. Such control could be achieved through a variety of means, for example, exploiting vulnerabilities over control links to hijack drones [17] or injecting malicious malware onto the devices [18]. Taking advantage of this, the adversaries could begin colluding to analyze and transmit data across a local adversary-formed network topology. To form this graph, the adversaries could employ device-to-device (D2D) communication protocols, which have proliferated in 5G wireless and have shown benefit in distributed learning, allowing devices to synchronize model parameters and/or gain a better estimate of the overall data distribution [19], [20], [21]. This decentralized topology allows for the exchange of information between adversaries that might not be readily available at an individual level, i.e., a more complete view of each sample. By pooling their individual observations, they can potentially infer sensitive information about the battlefield–such as troop numbers, logistics capabilities, or defensive positions. Moreover, this making it easier for them to mislead the central decision-making system via a backdoor attack (e.g., inducing the command center to misclassify an enemy fighter as a benign node).

Still, the adversaries might be unable to engage in maximal information exchange (i.e., forming a full mesh graph) due to e.g., resource constraints, geographical distances, and channel conditions preventing formation of certain D2D links. We thus need to understand how adversarial collusion impacts the attack potency, and the role played by adversarial connectivity. Moreover, following the common assumption in the literature that the adversaries have full control over the compromised node(s) [22], it is important to consider the full range of adversarial capabilities. By doing so, we do not underestimate the adversary's capabilities and insights, which could otherwise lead to overconfidence in the system's security.

A. Related Work

Extensive research has been conducted on backdoor attacks in HFL. In these cases, adversaries send malicious updates to the server, causing the model to misclassify data when a trigger is present without impacting the overall performance of the FL task [23], [24], [25]. In this domain, [7] proposed a scale-and-constrain methodology, in which the adversary's local objective function is modified to maximize attack potency without causing degradation of the overall FL task. Reference [26] explored trigger embeddings that take advantage of the distributed nature of HFL, by dividing the trigger into multiple pieces. In addition to the various attacks, defenses for these vulnerabilities in HFL have also been studied, e.g., [27] and [28]. Another significant issue with the effectiveness of backdoor attacks in HFL is the presence of non-i.i.d. data distributions across local clients, resulting in slower convergence of the global model [29]. In this regard, an HFL backdoor methodology [30] has been developed to work with a popular HFL algorithm called SCAFFOLD [31] by utilizing Generative Adversarial Networks (GANs) [32].

In our work, we focus on backdoor attacks for VFL, which have not been as extensively studied. The VFL scenario introduces unique challenges: devices do not have access to sample labels or a local loss function, and must rely on gradients received from the server to update their feature embedding models. Thus, unlike in the HFL scenario, where attackers can utilize a "dirty-label" backdoor by altering the labels on local datapoints [33], an attack in VFL must be a "clean-label" backdoor attack, since only the server holds the training labels.

In this context, a few recent techniques have been developed to carry out backdoor attacks in VFL through different methods for inferring training sample labels. These include using gradient similarity [34] and gradient magnitude [35] comparisons with a small number of reserve datapoints the adversary has labels for. In this domain, [13] created a local adaptive optimizer that changes signs of gradients inferred to be the target label. In a similar vein, other works have exploited the fact that the indices of the target label in the cross-entropy loss gradients will have a different sign, provided that the model dimension matches the number of classes [36], [37]. Other works have also considered gradient substitution alignment to conduct the backdoor task with limited knowledge about the target label [38]. Moreover, researchers have explored the implications of backdoor attacks in different settings, such as with graph neural networks (GNNs) [39]. Further, [40] considered training an auxiliary classifier to infer sample labels based on server gradients.

Despite these recent efforts, a major limitation of the existing approaches is that *they rely on information sent from the server to conduct the backdoor attack, in addition to using it for VFL participation*. This dependency can enable the server to implement defense mechanisms, particularly during the label inference phase, which can significantly limit the effectiveness of the attack [41]. While research on bypassing the use of server-received information in backdooring VFL exists, it is limited to only *binary classification tasks* [42]. Additionally, the aforementioned studies [34], [35], [36], [37], [42] *concentrate on the classic two-party VFL scenario with a single adversary*, which fits them into the cross-silo FL context [9], leading to a limited understanding of VFL backdoors in networks where multiple adversaries may collude to carry out the attack. In particular, unlike cross-silo FL, which typically involves a few participants such as large organizations, cross-device FL often encompasses numerous distributed devices collaborating to construct a global model, thereby significantly increasing the potential for security malfunctions [1], [7].

B. Research Questions and Approach Overview

These limitations lead us to investigate the following two research questions (RQs) in this work:

- **RQ1:** Can an adversary successfully implement a backdoor injection into the server's VFL model using only locally available information for label inference?
- **RQ2:** How can multiple adversaries collude with limited sharing to construct a backdoor injection in cross-device VFL, and what is the impact of their graph connectivity?

Overview of approach. We develop a novel backdoor VFL strategy that addresses the above questions. To answer RQ1, we introduce a methodology for an adversary to locally infer and generate datapoints of the target label for attack. Our approach leverages Variational Autoencoders (VAE) [43] and triplet loss metric learning [44] to determine which samples

should receive trigger embeddings, avoiding leveraging server gradients. To answer RQ2, we employ the graph topology of adversarial devices to conduct cooperative consensus on which samples should be implanted with triggers. In this regard, we show both empirically (Fig. 5d & Fig. 6) and theoretically (Sec. V) that the effectiveness of the attack and gradient perturbation is dependent on this graph topology. We also develop an intensity-based triggering scheme and two different methods for partitioning these triggers among adversaries, leading to a more powerful backdoor injection than existing attacks.

C. Outline and Summary of Contributions

- We propose a novel collaborative backdoor attack on VFL which does not rely on information from the server. Our attack employs a VAE loss structure augmented with metric learning for each adversary to independently acquire its necessary information for label inference. Following the local label inference, the adversaries conduct majority consensus over their graph topology to agree on which datapoints should be poisoned (Sec. III-A&IV-A).
- For trigger embedding, we develop an intensity-based implantation scheme which brings samples closer to the target without compromising non-target tasks. Attackers employ their trained VAEs to generate new datapoints to be poisoned that are similar to the target label, forcing the server to rely more on the embedded trigger. Adversary collusion is facilitated via two proposed methods for trigger splitting, either subdividing one large trigger or embedding multiple smaller triggers (Sec. III-B&IV-B).
- We conduct convergence analysis of cross-device VFL under backdoor attacks, revealing the degradation of main task performance caused by adversaries. Specifically, we show that the server model will have a stationarity gap proportional to the level of adversarial gradient perturbation (Sec. V). We provide an interpretation for this gradient perturbation as an increasing function of the adversary graph's algebraic connectivity and average degree, which we further investigate empirically (Sec. VI). We are unaware of prior works with such convergence analysis on VFL under backdoor attacks.
- We conduct extensive experiments comparing the performance of our attack against the state-of-the-art [34], [35] on five image classification datasets. Our results show that despite not using server information, we obtain a 30% higher attack success rate for comparable main task performance. We also show an added advantage of our decentralized attack in terms of improved robustness to noising defenses at the server. We also demonstrate that higher adversarial graph connectivity yields improved attack success rate with our method, thus corroborating our theoretical claims (Sec. VI).

II. SYSTEM MODEL

A. Vertical Federated Learning Setup

We consider a network of K nodes within the vertical federated learning (VFL) setup collected in the set $\mathcal{K} = $

TABLE I
SUMMARY OF MAIN NOTATIONS EMPLOYED THROUGHOUT THE PAPER

Notation	Description
k	Any client, excluding the server K
m	An adversary client
$\mathcal{K}$	The set of all clients, including the server
$\mathcal{A}$	The set of all adversary clients
G	The graph formed amongst adversary clients $m \in \mathcal{A}$
$\mathcal{A}_m$	A subset of $\mathcal{A}$, the neighbors to adversary m in graph G
$x_k^{(i)}$	The feature partition belonging to local client k for sample datapoint $x^{(i)}$
$X_m^{(i)}$	The concatenated samples for adversary m received over edges in G
$\widetilde{x}_m^{(i)}$	Sample generated from adversary m's VAE
$\widehat{x}_m^{(i)}$	$\widetilde{x}_m^{(i)}$ implanted with the trigger pattern subportion corresponding to adversary m
$z_m^{(i)}$	Latent variable produced from VAE encoder
$\mathcal{D}$	The overall dataset without being partitioned amongst clients
$\mathcal{D}_m$	A subset of $\mathcal{D}$ containing concatenated datapoints $X_m^{(i)}$
$\widehat{\mathcal{D}}_m$	A subset of $\mathcal{D}_m$, concatenated samples of only those where the label is known
$\widehat{\mathcal{D}}_m^{\text{target}}$	A subset of $\widehat{\mathcal{D}}_m$, concatenated datapoints belonging only to target label out of known datapoints
$\mathcal{D}_m^{(p)}$	Locally inferred datapoints for adversary m
$\mathcal{D}_g^{(p)}$	Collaborative inference set based off $\{\mathcal{D}_m^{(p)}|m \in \mathcal{A}\}$
$\widetilde{\mathcal{D}}_m^{(p)}$	m's feature partition slice of $\mathcal{D}_g^{(p)}$
f_k	The local model of client k, producing feature embeddings to be sent to server K
ϕ_K	The server model for server K
θ_k	The parameters for local models f_k. For server K, its server model is parameterized as θ_K
ϕ_m^{VAE}	Adversary m local VAE
μ	The mean vector of an adversarial VAE
$\phi_{m,\mu}$	Auxiliary classifier for adversary m
ζ	The poisoning budget
y_t	The target label
ρ	The connectivity of the graph G
$\delta(\rho)$	The gradient perturbation from adversaries for connectivity ρ
$\widehat{\kappa}$	The margin value of the triplet loss
a	Anchor datapoint for the triplet loss
p	Positive datapoint for the triplet loss
n	Negative datapoint for the triplet loss

$\{1, 2, \ldots, K\}$, where $k = 1, \ldots, K - 1$ are the clients and $k = K$ is the server. We assume a black-box VFL scenario, where the clients do not have any direct knowledge about the server and global objective, e.g., the model architecture, loss function, etc. We denote the overall dataset as $\mathcal{D}$, and the total number of datapoints is $N = |\mathcal{D}|$. Each client contains a separate disjoint subset of datapoint features. We represent the ith datapoint of $\mathcal{D}$ as $x^{(i)} = \{x_1^{(i)}, \ldots, x_{K-1}^{(i)}\}$, where $x_k^{(i)}$ belongs to the local data of client $k = 1, 2, \ldots, K - 1$. Note that only the server K holds the labels $\mathcal{Y} = \{y_1, \ldots, y_N\}$ associated with the corresponding dataset.

Each client locally trains its feature encoder on its data partition, and the server is responsible for coordinating the aggregation process. In particular, we adopt a Split Neural Network-based (SplitNN) VFL setup [9], where the clients send their locally produced feature embeddings (sometimes called the bottom model) to the server. The server then updates its global model (the top model) and returns the gradients of the loss with respect to the feature embeddings back to the clients.
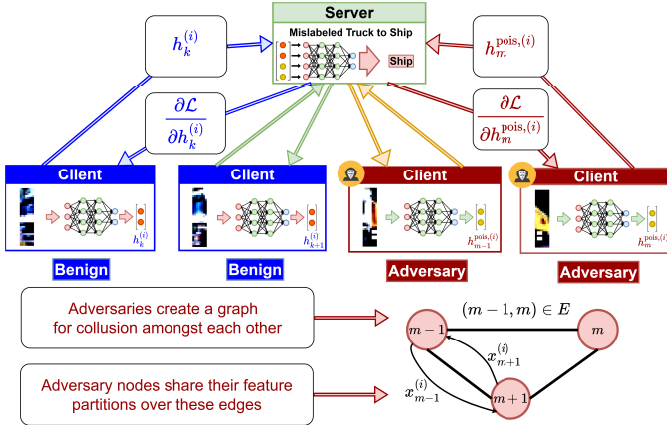
Fig. 1. Client-server sharing of embeddings and gradients in VFL. An example of a feature-partitioned datapoint undergoing a backdoor trigger implantation is shown. The adversaries send up their poisoned embeddings, which are then concatenated by the server and cause misclassification. Moreover, adversaries form a graph amongst each other, sharing their feature partitions to enhance insights on the samples they wish to poison.

Mathematically, the optimization objective of the VFL system can be expressed as

$$\min_{\theta_1,\ldots,\theta_K} F(\theta) := \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}\left(y_i, \phi_K\left(\{f_1(x_1^{(i)}; \theta_1),\right.\right.$$
$$\left.\left. f_2(x_2^{(i)}; \theta_2), \ldots, f_{K-1}(x_{K-1}^{(i)}; \theta_{K-1})\}; \theta_K\right)\right), \quad (1)$$

where f_k denotes the embedding mapping function of client $k \in \mathcal{K} \setminus \{K\}$ parameterized by θ_k, ϕ_K is the server model parameterized by θ_K, $\theta = \{\theta_1, \ldots, \theta_K\}$, and $\mathcal{L}$ denotes the loss function of the learning task.

To solve (1), in each VFL training round t, the server selects a set of mini-batch indices. Across clients, the full mini-batch set is $\mathcal{B}^{(t)} = \{(x_1^{(i)}, \ldots, x_{K-1}^{(i)})|x_k^{(i)} \in \mathcal{B}_k^{(t)}\} \subset \mathcal{D}$. Each client k needs to update its own local model θ_k on its mini-batch subset $\mathcal{B}_k^{(t)}$. However, different from HFL, the gradients of the clients' loss function models depend on information from the server, while the model update at the server also depends on the mapping computed by the clients. Thus, during round t, each client k first computes local low-dimensional latent feature embeddings $H_k^{(t)} = \{h_k^{(i)} = f_k(x_k^{(i)}; \theta_k^{(t)})|x_k^{(i)} \in \mathcal{B}_k^{(t)}\}$. After the embeddings $H_k^{(t)}$ are obtained, as seen in Fig. 1, they are sent to the server. The server computes the gradient $\frac{\partial \mathcal{L}}{\partial \theta_K}$ to update the top model θ_K via gradient descent: $\theta_K^{(t+1)} \leftarrow \theta_K^{(t)} - \eta_K^{(t)} \frac{\partial \mathcal{L}}{\partial \theta_K}$. In addition, the server computes $\frac{\partial \mathcal{L}}{\partial h_k^{(i)}}$, which is sent back to client k for the computation of gradients of the loss function with respect to the local model $\theta_k^{(t)}$ as $\frac{\partial \mathcal{L}}{\partial \theta_k} = \frac{1}{|\mathcal{B}_k^{(t)}|} \sum_{h_k^{(i)} \in H_k^{(t)}} \frac{\partial \mathcal{L}}{\partial h_k^{(i)}} \frac{\partial h_k^{(i)}}{\partial \theta_k}$. The client then updates its model via gradient descent: $\theta_k^{(t+1)} \leftarrow \theta_k^{(t)} - \eta_k^{(t)} \frac{\partial \mathcal{L}}{\partial \theta_k}$.

B. Backdoor Attacks in VFL

In this work, we investigate backdoor attacks on VFL, where each adversary is a client in the system (i.e., compromised client). The goal of the adversaries is to modify the server model's behavior on data samples of a target label (i.e., the label that adversaries want to induce misclassification on) via an implanted backdoor trigger on these samples. Importantly, however, the model should still perform well on clean data for which the trigger is not present. While modifying the objective function is a common method for backdoor attacks in the HFL case [7], this is not feasible in VFL because only the server can define the loss, and hence the adversaries must follow the server's loss function. Therefore, we consider attacks where an adversary m implants a trigger σ_m into a selected datapoint i *inferred* to be of the target label, i.e., producing $x_m^{(i)} + \sigma_m$.

In addition, we assume that multiple adversarial clients $\mathcal{A} \subset \mathcal{K}$ can collude to plan the attack. In this vein, we consider a connected, undirected graph $G = (\mathcal{A}, E)$ among the adversaries, where E denotes the set of edges. For adversary $m \in \mathcal{A}$, we denote $\mathcal{A}_m = \{m' : (m, m') \in E\}$ as its set of neighbors. Adversaries will employ G to conduct collaborative label inference, as will be described in Sec. III-A&IV-A.

III. ATTACK METHODOLOGY

To execute the backdoor attack in VFL, adversaries need to (i) identify datapoints belonging to the target label (Sec. III-A) and (ii) implant triggers on the corresponding datapoints to induce misclassification (Sec. III-B). We present the methodology for these processes in this section, and give more specific algorithmic procedures in Sec. IV.

A. Label Inference

Our label inference methodology is summarized in Fig. 2:

1) Feature Sharing: As in existing work [13], [35], we assume that the adversaries possess labels for a small (e.g., $< 1\%$) set of the datapoints. Even so, when numerous clients each hold a small feature partition of the samples (Feature partition block of Fig. 2), extracting meaningful information without employing gradients from the server (which we aim to avoid, as discussed in Sec. I) becomes challenging. This difficulty arises due to the presence of irrelevant features within sample partitions, e.g., a blank background.

To address this issue, the adversaries utilize their collusion graph G discussed in Sec. II-B, through which they exchange feature partitions of their datapoints with their one-hop neighbors (Feature Sharing phase of Fig. 2). Each adversary m concatenates the partitions as $X_m^{(i)} = \bigcup_{m' \in \mathcal{A}_m \cup m} x_{m'}^{(i)}$. We denote this dataset as $\mathcal{D}_m \subset \mathcal{D}$, a further subset $\widehat{\mathcal{D}}_m \subset \mathcal{D}_m$ of which is for known labels.

2) VAE and Metric-based Label Inference: Next, using $\mathcal{D}_m$, each adversary will conduct local label inference (VAE and Metric-based Label Inference phase of Fig. 2). We propose leveraging Variational Autoencoders (VAE) as a framework for this, deploying one VAE model ϕ_m^{VAE} on each adversary device. Unlike their AE counterparts [45], VAEs are simpler to use for generative purposes, as a variable z sampled from the VAE's latent space can be fed through the decoder to generate new datapoints [43]. To do this, we assume that each datapoint $X_m^{(i)}$ is generated from latent variables $z_m^{(i)}$ following a distribution $p(z_m^{(i)})$, which usually is a standard normal distribution, $\mathcal{N}(0, I)$. Therefore, the goal of the VAE's decoder model is to learn its parameters to maximize $p_{m,d}(X_m^{(i)}|z_m^{(i)})$.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

LEE et al.: COOPERATIVE DECENTRALIZED BACKDOOR ATTACKS ON VERTICAL FEDERATED LEARNING
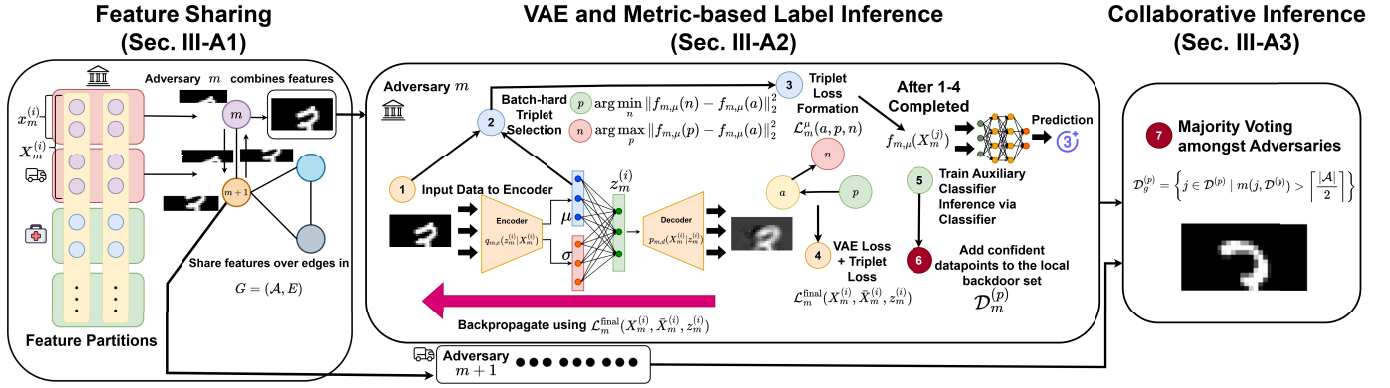
5

Fig. 2. Label inference methodology with modified VAE architecture. Initially, the adversaries share their feature partitions. At the μ layer of the VAE, triplet margin loss is employed to conduct metric learning via the known label datapoints. After training the VAE, the μ vectors are used to perform a classification task for inference of the target label, with the results of local inference being used in a majority voting scheme for a final collaborative inference of the indices.

However, $p(X_m^{(i)}) = \int p_{m,d}(X_m^{(i)}|z_m^{(i)})p(z_m^{(i)})dz_m^{(i)}$ is computationally intractable, making it unrealistic to calculate the term directly. Therefore, rather than maximizing $p_{m,d}$ directly, the VAE employs its encoder $q_{m,e}$ as an approximate model which outputs a mean μ and standard deviation σ, reducing the latent space to a univariate Gaussian $\mathcal{N}(\mu, \sigma^2)$. The error can be captured in a KL divergence-based objective [46] which measures the difference between two probability distributions, $q_{m,e}(z_m^{(i)})$ and $p(z_m^{(i)})$, denoted as $D_{\mathrm{KL}}(q_{m,e}(z_m^{(i)})\|p(z_m^{(i)}))$. This term, when included in the loss function, can encourage the latent space to be closer to a standard normal distribution, allowing for random sampling from the latent space for datapoint generation.

Additionally, the VAE aims to optimize its reconstruction loss. We adopt the mean-squared error (MSE) metric $\mathcal{L}_m^{\mathrm{rec}}(x, \bar{x}_m(x)) = \|x - \bar{x}_m(x)\|_2^2$, where $\bar{x}_m(x)$ is the output reconstructed by the decoder for input x. Combining these together, a typical VAE trains for any datapoint x on the objective function

$$\mathcal{L}_m^{\mathrm{VAE}}(x, \bar{x}_m(x), z) = \lambda \cdot \mathcal{L}_m^{\mathrm{rec}}(x, \bar{x}_m(x))$$
$$+ (1-\lambda) \cdot D_{\mathrm{KL}}(q_{m,e}(z)\|p(z)), \quad (2)$$

where $0 < \lambda < 1$ captures the importance weight of each individual term.

A key advantage of a VAE is its ability to utilize the latent space to learn separable embeddings. Additionally, existing work has hypothesized that applying metric learning to the μ vector can enhance embedding alignment within the latent space [47]. We leverage this by training a joint triplet margin loss [44] objective alongside the standard VAE, given by

$$\mathcal{L}_m^\mu(a, p, n) = \max\left(d_{m,\mu}^2(a, p) - d_{m,\mu}^2(a, n) + \widehat{\kappa}, 0\right), \quad (3)$$

where $d_{m,\mu}(a, r) = \|f_{m,\mu}(a) - f_{m,\mu}(r)\|_2$. Here, a is the anchor datapoint, p is a datapoint with the same label (called the *positive*), n is a datapoint belonging to a different label (called the *negative*), $\widehat{\kappa}$ is the margin hyperparameter, and $f_{m,\mu}(\cdot)$ is the function induced by the μ vector of the VAE. The triplet margin loss creates embeddings that reduce the distance between the anchor a and the positive p in the feature

space while ensuring the negative n is at least a distance $\widehat{\kappa}$ from p.

Now, using the labeled dataset $\widehat{\mathcal{D}}_m$, the positives and anchors are the set of datapoints belonging to the target label, and negatives are from the other labels. However, as outlined in [44], careful triplet selection is required for a good embedding alignment. Therefore, we employ the "batch-hard" method of online triplet selection [48], where the "hardest" positive and negative are chosen. These include the farthest positive and the closest negative to the anchor embedding, given by $\widehat{p}_{m,\mu}(a) = \arg\max_p \|f_{m,\mu}(p) - f_{m,\mu}(a)\|_2^2$ and $\widehat{n}_{m,\mu}(a) = \arg\min_n \|f_{m,\mu}(n) - f_{m,\mu}(a)\|_2^2$ respectively (Steps 2 and 3 of Fig. 2). Now, combining these all together, we can formulate the final loss each adversary VAE trains on as

$$\mathcal{L}_m^{\mathrm{final}}(X_m^{(i)}, \bar{X}_m^{(i)}, z_m^{(i)}) = \mathcal{L}_m^{\mathrm{VAE}}\left(X_m^{(i)}, \bar{X}_m^{(i)}, z_m^{(i)}\right)$$
$$+ \widehat{\lambda} \cdot \mathcal{L}_m^\mu\left(X_m^{(i)}, \widehat{p}_{m,\mu}(X_m^{(i)}), \widehat{n}_{m,\mu}(X_m^{(i)})\right), \quad (4)$$

which is shown as Step 4 in Fig. 2. Our experiments in Sec. VI will demonstrate the benefit of this hybrid VAE and metric learning approach, i.e., training on $\mathcal{L}_m^{\mathrm{final}}$ versus $\mathcal{L}_m^{\mathrm{VAE}}$.

After training the VAE, we introduce an auxiliary classifier $\phi_{m,\mu}$, which is trained in a supervised manner using the latent embeddings from the μ vector (Step 5 of Fig. 2), denoted by $f_{m,\mu}(\cdot)$. We use the cross-entropy loss $\mathcal{L}(y, \hat{y}) = -\sum_{c \in \mathcal{C}} y_c \log(\hat{y}_c)$ as the objective function, where y_c is an indicator for whether the data point is from class c, $\hat{y}_c$ is the softmax probability for the cth class, and $y, \hat{y}$ are the corresponding vectors. This is trained on $\{f_{m,\mu}(X_m^{(i)})|X_m^{(i)} \in \widehat{\mathcal{D}}_m\}$. In this way, the embeddings are trained to be separable, associating label positions in the latent space with their corresponding labels [49]. We can then employ this to construct the set of locally inferred target datapoints from adversary m, $\mathcal{D}_m^{(p)}$ (Step 6 of Fig. 2), as will be described in Sec. IV-A.

3) Collaborative Inference: Upon completing the local inference phase, the adversaries utilize their locally inferred labels to reach a consensus over the local graph G on which datapoints are from the target label (Collaborative Inference phase of Fig. 2). This consensus can be reached in several potential ways, e.g., through a leader adversary node perform-

ing a Breadth-First Search (BFS) [50] traversal on the graph, followed by a majority voting scheme (Step 7 of Fig. 2). This is the method we will employ in Sec. IV-A.

Each of these label inference processes, e.g., feature sharing, VAE training/inference, and collaborative inference, take place before the VFL training process begins. This emphasizes one of our contributions: developing a backdoor attack where gradients from the server are not utilized for label inference. As, we will see in Sec. VI-B, this enhances robustness against server-side gradient noise injection defenses. Further, by minimizing the degree to which adversary behavior will deviate from the VFL training protocol, the possibility of the system detecting the attack (through e.g., anomalous communication, computation, or energy consumption) becomes small.

B. Trigger Embedding

In the next step, after the target data points have been inferred, the VFL training process starts, during which adversaries embed triggers into inferred data points. One of the primary challenges for the adversaries conducting a backdoor attack is to ensure that both the primary task and the backdoor task perform effectively. To this end, adversaries poison a specific subset of the inferred data points, defined as $\mathcal{D}^{\text{pois}}$, according to a poisoning budget $\zeta = \frac{|\mathcal{D}^{\text{pois}}|}{|\mathcal{D}|}$. A smaller budget also helps prevent detection of the malicious operation.

Formally, during training iteration t, if any datapoint i from $\mathcal{D}^{\text{pois}}$ is present in the minibatch $\mathcal{B}^{(t)}$, adversary m will implant a trigger σ_m into its local $x_m^{(i)}$ as $\widehat{x}_m^{(i)} = x_m^{(i)} + \sigma_m$. Here, $\widehat{x}_m^{(i)}$ is the datapoint embedded with the trigger, and the adversary aims for the server to learn and associate this trigger pattern with the target label. Unlike most existing works, our setup considers more than one adversary. Therefore, in Sec. IV-B, we will propose two different methods for generating σ_m across adversaries $m \in \mathcal{A}$ as a subpartition or smaller version of the trigger σ that would be embedded by a single adversary.

Our aim is to make the server rely on the trigger while still learning features relevant to the target label, by leveraging the VAE's generative properties. This involves the following steps:

- **Data Generation:** Using adversary m's VAE, we generate datapoints $\widetilde{x}_m^{(i)}$ by sampling vector $z_m^{\text{gen}} \sim \mathcal{N}(0, I)$ from the latent space and passing them through the decoder $p_{m,d}$.
- **Data Substitution and Selective Poisoning:** The adversary swaps the original datapoints $x_m^{(i)}$, similar to [34], with the newly generated ones $\widetilde{x}_m^{(i)}$, and embeds them with the trigger. This is performed on a subset of the inferred data points during training, according to the poisoning budget ζ.

As a result, the server learns more variations of the target label, which intuitively leads it to rely more on the trigger for classification. Additionally, the generated samples will still follow the general structure of the target label, to prevent misclassification of labels not involved in the backdoor attack.

In designing the trigger, we aim for the poisoned datapoints to produce embeddings that are as close as possible to embeddings of non-poisoned embeddings. This can be thought of mathematically as looking for triggers that will minimize

$\|\mathbb{E}[h_m^{\text{pois}}] - \mathbb{E}[h_m^{\text{target}}]\|_2$, where $\mathbb{E}[h_m^{\text{pois}}]$ is the expectation over feature embeddings produced from datapoints i implanted with the trigger and $\mathbb{E}[h_m^{\text{target}}]$ is the expectation over feature embeddings produced from clean datapoints i' belonging to the target label. If adversary m had access to the server's loss function, it may be possible to incorporate (an approximation of) this norm difference directly into the adversary's local VFL update. However, we are considering a black-box scenario. Hence, to emulate the desired trigger behavior, we complement the data substitution and selective poisoning process with an *intensity-based triggering scheme*. Detailed in Sec. IV-B, this scheme enhances the background value of the trigger by an adversary-defined intensity value γ, so that the trigger becomes more prominent within the datapoint. Thus, on the one hand, $\widetilde{x}_m^{(i)}$ presents the server with more variations of the target label, causing the server to rely more heavily on the consistent trigger pattern. Then, combining these harder-to-identify background features with the γ-enhanced trigger induces heavier reliance of the server on the trigger while minimizing main task performance degradation (when the trigger is not present).

The VFL system relies on the bottom models' outputted latent embeddings for classification, with updates to these local models aiming to optimize performance. By embedding the target datapoints with an intensified trigger, we can pull the embeddings h_m^{pois} of the backdoored samples closer to embeddings h_m^{target} of the target label, making it harder for the server to distinguish between them.

IV. ALGORITHM DETAILS

We now provide specific algorithms for implementing the label inference (Sec. IV-A) and trigger embedding (Sec. IV-B) methodologies from Sec. III.

A. Label Inference

Alg. 1 summarizes our label inference approach. Recall the goal is to find datapoints of the target label y_t, i.e., to find which datapoints are candidates to be poisoned. This trains the VAE model ϕ_m^{VAE} and auxiliary classifier $\phi_{m,\mu}$ for adversary m, which is then capable of generating datapoints of y_t based off local inference datapoints $\mathcal{D}_m^{(p)}$. As input, the overall algorithm will utilize the set of adversaries $\mathcal{A}$ and concatenated datapoints $\mathcal{D}_m$ for adversary m to infer datapoints of y_t. We detail the steps of Alg. 1 below:

1) Training VAE ϕ_m^{VAE} and Auxiliary Classifier $\phi_{m,\mu}$: As outlined in Sec. III-A, label inference is performed before the standard VFL protocol. Firstly, ϕ_m^{VAE} is trained via (4) and utilizes the "batch-hard" strategy discussed in Sec. III-A (Line 1 of Alg. 1). The VAE's reconstruction loss $\mathcal{L}_m^{\text{VAE}}$ employs $\widehat{\mathcal{D}}_m^{\text{target}} \subseteq \widehat{\mathcal{D}}_m$, where $\widehat{\mathcal{D}}_m^{\text{target}}$ is the set of known concatenated datapoints belonging only to the target y_t. For the triplet loss $\mathcal{L}_m^{\mu}$, positives are selected from the VAE's training batch and the negatives are taken from $\widehat{\mathcal{D}}_m$. Next, $\phi_{m,\mu}$ takes in the μ embeddings on labeled datapoints and trains via cross-entropy loss (Line 2 of Alg. 1).

After training the VAE, $\phi_{m,\mu}$ takes in the μ-embeddings of the datapoints from $\mathcal{D}_m$ with unknown labels as input (Line 3

Algorithm 1 VAE and Metric-Based Label Inference

Input: VAE model ϕ_m^{VAE}, auxiliary model $\phi_{m,\mu}$, adversary index $m \in \mathcal{A}$, target y_t, known concatenated datapoints $\widehat{\mathcal{D}}_m$, all concatenated datapoints $\mathcal{D}_m$

1 Train ϕ_m^{VAE} using $\mathcal{L}_m^{\text{final}}$ from (4), which uses (2) and (3)
2 Train auxiliary classifier $\phi_{m,\mu}$ via $\{f_{m,\mu}(X_m^{(i)}) | X_m^{(i)} \in \widehat{\mathcal{D}}_m\}$
3 **for** *adversary index* $m \in \mathcal{A}$ **do**
4   **for** *datapoint* $X_m^{(i)} \notin \widehat{\mathcal{D}}_m$ *and* $X_m^{(i)} \in \mathcal{D}_m$ **do**
5       $\hat{y} = \phi_{m,\mu}(f_{m,\mu}(X_m^{(i)}))$
6   **if** $y_t = arg\max_c(\hat{y}_c)$ *and* $\max(\hat{y}) \geq \beta$ **then**
7   Insert $x_m^{(i)}$ to $\mathcal{D}_m^{(p)}$
8   Adjust model architecture and retrain ϕ_m^{VAE} on $\mathcal{D}_m^{(p)}$
9 Get indices $\mathcal{D}_g^{(p)} \leftarrow$ CONSENSUS($\mathcal{A}$) from Sec. IV-A
10 **for** $j \in \mathcal{D}_g^{(p)}$ **do**
11   Insert datapoint $x_m^{(j)}$ to $\widetilde{\mathcal{D}}_m^{(p)}$

Output: VAE ϕ_m^{VAE}, inferred adversary dataset $\widetilde{\mathcal{D}}_m^{(p)}$
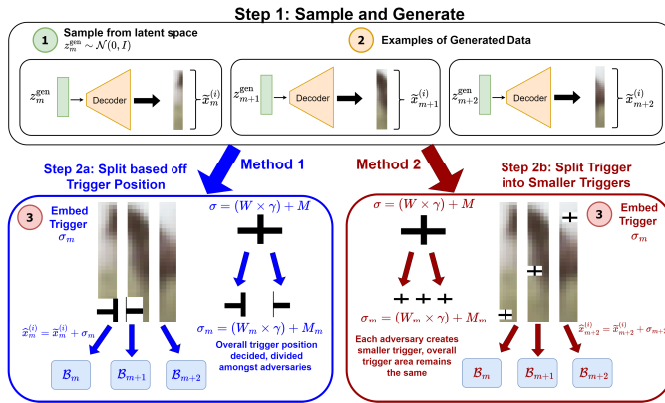

Fig. 3. Image generation and trigger-embedding process. The adversaries can choose one of two methods: (1) constructing a collaborative trigger on some position of the known adversary features, or (2) giving each adversary a smaller trigger. Method 1 may result in some adversaries not possessing any portion of the trigger pattern, i.e., only having the background.

of Alg. 1), and populate $\mathcal{D}_m^{(p)}$, the set of locally inferred target datapoints. $\mathcal{D}_m^{(p)}$ is initially the same as $\widehat{\mathcal{D}}_m^{\text{target}}$ in terms of indices. The datapoints are added to the set if the maximum prediction probability corresponds to the target label y_t and is greater than a confidence threshold β, i.e., $\max(\hat{y}) \geq \beta$ and $y_t = \arg\max_c(\hat{y}_c)$ (Lines 5-7 of Alg. 1). In this way, only datapoints $\phi_{m,\mu}$ is confident about will be considered as targets.

Afterwards, adversary m's VAE is retrained on $\mathcal{D}_m^{(p)}$ (Line 8 of Alg. 1). The retraining allows the target datapoint generation to match the shape of the feature partition.

2) Consensus Amongst Adversaries: After local label inference, collaborative inference begins (Line 9 of Alg. 1) with the following steps:

- **Choosing Leader Node and BFS-traversal:** One adversary in graph G, denoted m^{lead}, is chosen by selecting

---

Algorithm 2 Distributed Trigger Embedding

Input: Server K, client $k \in \mathcal{K} \setminus \{K\}$, adversary set $\mathcal{A}$, adversary VAE ϕ_m^{VAE}, bottom-model parameters θ_k, to-be-poisoned data $\mathcal{D}_m^{\text{pois}}$ when $m \in \mathcal{A}$

1 **for** $t = 0$ *to* $T-1$ **do**
2   **for** *clients* $k \in \mathcal{K} \setminus \{K\}$ **do in parallel**
3   Sample local minibatch $\mathcal{B}_k^{(t)}$
4   **for** *datapoint* $x_k^{(i)} \in \mathcal{B}_k^{(t)}$ **do**
5   **if** $x_k^{(i)} \in \mathcal{D}_m^{pois}$ *and client* $k \in \mathcal{A}$ **then**
6   Adversary generates $\widetilde{x}_k^{(i)} = p_{m,d}(z_k^{\text{gen}})$
7   Add trigger pattern from (5)
8   Adversary replaces original data: $x_k^{(i)} = \widehat{x}_k^{(i)}$
9   Compute $H_k^{(t)} = \{h_k^{(i)} = f_k(x_k^{(i)}; \theta_k^t) | x_k^{(i)} \in \mathcal{B}_k^{(t)}\}$
10   Transmit $H_k^{(t)}$ to server K
11   Server computes $\{\frac{\partial \mathcal{L}}{\partial h_k^{(i)}} | h_k^{(i)} \in H_k^{(t)}\}, k \in \mathcal{K} \setminus \{K\}$, sends them back to client k, computes $\frac{\partial \mathcal{L}}{\partial \theta_K}$, and updates $\theta_K^{(t+1)} \leftarrow \theta_K^{(t)} - \eta_K^{(t)} \frac{\partial \mathcal{L}}{\partial \theta_K}$
12   **for** *each client* $k \in \mathcal{K} \setminus \{K\}$ **do in parallel**
13   Compute $\frac{\partial \mathcal{L}}{\partial \theta_k}$ and update $\theta_k^{(t+1)} \leftarrow \theta_k^{(t)} - \eta_k^{(t)} \frac{\partial \mathcal{L}}{\partial \theta_k}$

the node with the highest degree. m^{lead} will conduct BFS over G to collect the local inference results from each adversary.

- **Consensus Voting:** Next, given a multiset of the local inferred datasets $\mathcal{D}^{(p)} = \{\mathcal{D}_m^{(p)}\}_{m \in \mathcal{A}}$, m^{lead} adopts a simple majority based voting scheme similar to [40]. If an index j appears for more than $\left\lceil \frac{|\mathcal{A}|}{2} \right\rceil$ times in $\mathcal{D}^{(p)}$, it is added to the collaborative inference set $\mathcal{D}_g^{(p)}$. In other words, $\mathcal{D}_g^{(p)} = \left\{ j \in \mathcal{D}^{(p)} \mid m(j, \mathcal{D}^{(p)}) > \left\lceil \frac{|\mathcal{A}|}{2} \right\rceil \right\}$, where $m(j, \mathcal{D}^{(p)})$ is the multiplicity of j in $\mathcal{D}^{(p)}$ for an index j.
- **Sharing Final Results:** Lastly, m^{lead} will conduct BFS again, propagating the final indices of $\mathcal{D}^{(p)}$ to all adversaries, with m's feature partition slice of $\mathcal{D}_g^{(p)}$ is defined as $\widetilde{\mathcal{D}}_m^{(p)}$.

While we adopt the BFS traversal-based method, other graph traversal methods (e.g., DFS, spanning tree) could also be used, as long as all adversaries correctly receive the voting results.

B. Trigger Embedding

Alg. 2 summarizes our trigger embedding algorithm. Recall that after label inference, our goal is to conduct trigger embedding on the inferred datapoints. Overall, during the VAE protocol, we implant a trigger on generated datapoints from VAE ϕ_m^{VAE}. We detail the steps of Alg. 2 below:

1) Poisoning and Trigger Implantation: To maintain a high accuracy in the main task along with the backdoor, a

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

8   IEEE TRANSACTIONS ON NETWORKING

poisoning budget ζ outlined in Sec. III-B is utilized to limit the number of poisoned datapoints. The selected indices to be poisoned are chosen at random from $\mathcal{D}_g^{(p)}$, with the corresponding sub-dataset for adversary m denoted $\mathcal{D}_m^{\text{pois}} \subset \mathcal{D}^{\text{pois}}$.

Now, the trigger implantation follows a two step process, depicted in Fig. 3. Firstly, a to-be-poisoned datapoint in a minibatch is replaced with a datapoint $\widetilde{x}_m^{(i)}$ generated from the decoder of the VAE, $p_{m,d}$ (Line 6 of Alg. 2). Secondly, an intensity-based trigger is formed and distributed among adversaries. This can follow one of two methods, which we consider in the context of image data, where each sample x is a matrix of pixels (or a tensor in the multi-channel case). Starting with Method 1, adversaries collaboratively insert a trigger into a target datapoint at a location specified by centering parameter $\ell = (\ell_x, \ell_y)$. The trigger has a background of 1's with area $W = h \times w$, divided among adversaries based on their feature partitions' proximity to ℓ. The background's value is enhanced by multiplying pixel-wise with an intensity parameter γ, which controls the trigger's prominence. A cross pattern of 0's is added to complete the trigger (see Fig. 3). Overall, the general trigger pattern is similar to the trigger adopted in [40]. Each adversary m receives a portion of the trigger according to their position relative to ℓ. The final datapoint with backdoor implantation is given by

$$\widehat{x}_m^{(i)} = \widetilde{x}_m^{(i)} + ((W_m \times \gamma) + M_m). \tag{5}$$

Here, $\widehat{x}_m^{(i)}$ is created by implanting the trigger into $\widetilde{x}_m^{(i)}$ (Line 7 of Alg. 2), where W_m is the trigger background portion falling within adversary m's local partition, and M_m is the local cross pattern. The trigger pattern is limited by a maximum area budget ϵ to avoid server detection, i.e., $h \cdot w \le \epsilon$.

2) Alternative Trigger Embedding Method: Next, we describe an alternative method for adversaries to implant a trigger, referred to as Method 2 in Fig. 3. In this case, instead of one collaborative trigger, each adversary implants a smaller subtrigger on their feature partition. The smaller subtriggers, when combined, should still have the same total area as the collaborative trigger; for example, all of their individual areas can be $W_m = \frac{h \cdot w}{|\mathcal{A}|}$. The subtriggers are placed randomly within the datapoint, preventing the server from memorizing the trigger pattern by location to enhance generalizability.

V. Convergence Analysis

We now analyze the convergence of VFL in the presence of backdoor attacks. We first make the following assumptions:

Assumption 1 (L-Smoothness): The loss function $F(\theta)$ in (1) is L-smooth, meaning that for any x and y, we have

$$F(y) \le F(x) + \langle \nabla F(x), y - x \rangle + \frac{L}{2} \|y - x\|^2.$$

Assumption 2 (Variance): The mini-batch gradient $\nabla_{\theta_k} \mathcal{L}$ is an unbiased estimate of $\nabla_{\theta_k} F(\theta)$, and

$$\mathbb{E} \|\nabla_{\theta_k} \mathcal{L} - \nabla_{\theta_k} F(\theta)\|^2 \le \Gamma, \forall k \in \mathcal{K},$$

where Γ is the variance of $\nabla_{\theta_k} \mathcal{L}$.

Assumption 3 (Perturbation): There is an upper bound for the gradient perturbation from adversaries, i.e.,

$$\mathbb{E} \|\nabla_{\theta_k}^a \mathcal{L} - \nabla_{\theta_k} \mathcal{L}\|^2 \le \delta(\rho), \forall k \in \mathcal{K},$$

where $\nabla_{\theta_k}^a \mathcal{L}$ denotes the perturbed gradient and ρ is a measure of connectivity for the graph G.

Assumptions 1 and 2 are common in literatures [51] and [52], while Assumption 3 characterizes the perturbation induced by the attack. We expect that $\delta(\rho)$ is an increasing function of ρ: a higher graph connectivity for collusion implies a higher probability that more datapoints will be targeted, leading to a larger perturbation. More specifically, as ρ increases, adversary m's concatenated sample $X_m^{(i)}$ becomes closer to the full feature set for each datapoint i. With more features, m can infer more samples and achieve lower expected loss $\mathcal{L}(y, \hat{y})$ in local label inference. Consequently, adversaries will tend to perturb more samples from the target label, and the expected perturbation in Assumption 3 increases. Here, we can also consider a simple illustrative example: suppose there are multiple compromised nodes in an image classification task, aiming to perturb samples corresponding to the "truck" label. Each adversary holds a small portion of the overall features (blocks of pixels). When connectivity is low (e.g., as with a line graph topology), each adversarial node receives only a small share of features from neighbors. This may leave the adversaries without enough information after exchange to reliably identify samples containing a truck, such as when all of the received features correspond to pixels offset from the object (e.g., the surrounding sky). In contrast, with a higher ρ (e.g., as with a fully connected graph topology), each adversary gains a more complete view of the contents of sample i in its concatenated $X_m^{(i)}$. This gives them a higher chance of identifying presence of trucks, resulting in more samples to poison after the collaborative inference process, and thus a higher δ. We will observe this relationship experimentally in Sec. VI, where ρ is taken as the second smallest eigenvalue of the Laplacian matrix of G (the algebraic connectivity or Fiedler value) [53], which often increases with average node degree.

δ is thus the parameter that connects the attack performance to the graph connectivity ρ. We demonstrate its impact on the model convergence in the following theorem:

Theorem 1: Suppose that the above assumptions hold, and the learning rate is upper bounded as $\eta_1^{(t)} = \eta_2^{(t)} = \cdots = \eta_K^{(t)} = \eta^{(t)} \le \frac{1}{4L}$. Then, the iterates generated by the backdoored SplitNN and vanilla SplitNN satisfy

$$\min_{t \in \{0, \ldots, T-1\}} \{\mathbb{E} \|\nabla F(\theta^{(t)})\|^2\} \le 4 \frac{F(\theta^{(0)})}{\sum_{t=0}^{T-1} \eta^{(t)}}$$
$$+ 4 \frac{\sum_{t=0}^{T-1} (\eta^{(t)})^2}{\sum_{t=0}^{T-1} \eta^{(t)}} (KL\Gamma + KL\delta(\rho)) + 2K\delta(\rho). \tag{6}$$

proof: The proof is contained in Appendix.

When there is no attack, i.e., $\delta(\rho) = 0$, the bound in Theorem 1 recovers the result of VFL in [54]. Under a learning rate $\eta^{(t)}$ that satisfies $\sum_{t=0}^{T-1} (\eta^{(t)})^2 \to 0$ and $\sum_{t=0}^{T-1} \eta^{(t)} \to \infty$ for $T \to \infty$, the first and second terms in the right-hand side of inequality (6) diminish to zero. The adversarial attack induces a constant term $2K\delta(\rho)$ within the convergence bound, reflecting the convergence degradation due to the adversarial perturbations. Since $\delta(\rho)$ is an increasing function in terms of the connectivity, we see that the gap from

TABLE II
NETWORK ARCHITECTURE AND HYPERPARAMETERS
FOR EACH DATASET

Parameters	Dataset	
	MNIST	FMNIST
Margin κ	0.4	0.25
Poisoning budget ζ	1%	1%
Intensity factor γ	20	30
Confidence threshold β	0.999	0.999
# Auxiliary samples	360	360
% Auxiliary as Target	16%	16%
VAE Latent Dimension	32	64

Parameters	Dataset		
	CIFAR-10	SVHN	CIFAR100-20
Margin κ	0.2	0.2	0.2
Poisoning budget ζ	1%	1%	0.5%
Intensity factor γ	30	20	95
Confidence threshold β	0.9985	0.99995	0.99999
# Auxiliary samples	350	560	487
% Auxiliary as Target	14%	12.5%	15.4%
VAE Latent Dimension	512	256	512

a stationary point induced by the backdoor attack becomes progressively larger.

VI. NUMERICAL EXPERIMENTS

In this section, we evaluate the performance of our proposed approach on various datasets. We compare our performance with two state-of-the-art backdoor VFL attack methods discussed in Sec. I-A: BadVFL [34] and VILLAIN [35].

A. Simulation Setup

We perform experiments on the MNIST [55], Fashion-MNIST (FMNIST) [56], CIFAR-10 [57], Street View House Numbers (SVHN) [58], and CIFAR-100 (CIFAR100-20, coarse label version) [57], [59], [60] datasets. We consider fully-connected VAEs for MNIST and FMNIST, a Convolutional VAE (CVAE) with 4 layers each for the decoder and encoder for CIFAR-10 and CIFAR100-20, and a 4-layer encoder and 3-layer decoder CVAE for SVHN. For the bottom model, MNIST and FMNIST adopt a two layer Convolutional Neural Network (CNN), with CIFAR-10 and CIFAR100-20 having the same architecture as the encoder of the VAE. SVHN has the same bottom model as CIFAR-10. For the top model, we adopt a two-layer fully-connected network, which trains using the Adam optimizer [61]. Further details are given in Table II.

In our experiments, unless stated otherwise, there are 10 clients with 5 adversaries, with the adversaries utilizing trigger Method 1 from Sec. IV-B by default. Moreover, since the baselines do not consider adversary graphs in their methodology, we by default consider a fully-connected graph for fair comparison. In addition, note that both baselines assume only one adversary, and we extend their method to multiple adversaries by adding majority voting and trigger splitting to their label inference and attack processes. All experiments were conducted on a server with a 40GB NVIDIA A100-PCIE GPU and 128GB RAM, using PyTorch [62] for neural network design and training.

B. Competitive Analysis With Baselines

Attack Success Rate (ASR) and Clean Data Acc. (CDA): First, we assess the accuracy of the backdoor attack and the main task and compare the performance of our approach to the BadVFL and VILLAN baselines. We define the accuracy of misclassifying a poisoned datapoint as the target label as Attack Success Rate (ASR) and the accuracy of the regular main task as Clean Data Accuracy (CDA), which is the same notation utilized in [35]. We intend to show a high ASR, indicating a successful backdoor, while keeping the CDA close to the baselines, indicating the main task is not significantly affected.

To measure the accuracy of both the attack (ASR) and the main task (CDA), we evaluate the main task on the test dataset, and select 250 random datapoints from the test set in each communication round that do not belong to the target label to embed the trigger for evaluation of the ASR. While we use a trigger dimension $W = 5 \times 7$ for the proposed method, for BadVFL [34], we follow their method of a white square, and set the trigger size to 9×9. For VILLAIN [35], the trigger embedding method involves poisoning the embedding vector instead of the datapoint, and we poison 35% of the embedding.

Our results are given in Fig. 4. We note the superiority of our approach compared to the baselines: our attack achieves higher ASR across all settings, while the CDA stays relatively constant with the CDAs of the baselines. There are three reasons for this. First, due to better label inference performance (outlined in following section), the adversaries are more accurately poisoning datapoints belonging to the target, making it more likely for the server to draw an association between the target label and the trigger. Secondly, the intensity-based triggers are more easily captured by the bottom-models, making the server more reliant on it for classification. Third, the samples generated by the adversary VAEs follow the same general patterns and features of the target label, meaning that it is still learning the overall structure of the clean features properly when poisoned datapoints are present, keeping both the ASR and CDA high. Note that our findings are consistent on each of the datasets; compared to MNIST, Fashion-MNIST, CIFAR-10, and SVHN, the CIFAR100-20 experiments validate our results on a larger and more complex dataset. This corroborates the efficacy of our proposed method in implanting a successful backdoor on a wide range of learning scenarios.

We also note that our method requires a smaller trigger area compared to BadVFL [34] to successfully carry out the attack. Additionally, BadVFL is sensitive to its initial known datapoint, causing the backdoor to fail during some runs, accounting for the high variation in the averages. As seen in Fig. 4, the ASR average is low for CIFAR100-20 and SVHN due to a poor initial known datapoint being used often. On the other hand, VILLAIN [35] must wait for a period of time before label inference and attacking can begin, since the method requires a well-trained bottom and top model to carry out an attack. Lastly, the baselines, particularly VILLAIN, slightly suffer from catastrophic forgetting, where the back-
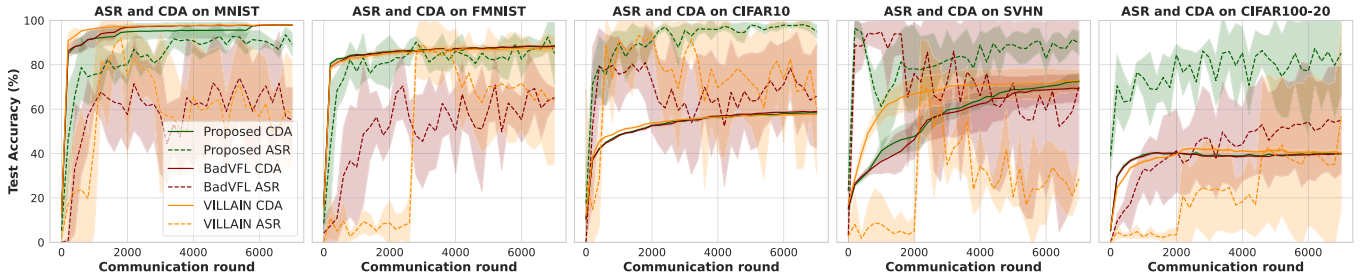
Fig. 4. Attack Success Rate (ASR) and Clean Data Accuracy (CDA) for MNIST, Fashion-MNIST, CIFAR-10, SVHN, and CIFAR100-20. The proposed method converges to a higher ASR value than the baselines (BadVFL [34] and VILLAIN [35]) due to (1) having a higher label inference accuracy as seen in Fig. 5a and (2) having an intensity based trigger that makes it easier for the server to draw the association between the target label and trigger.
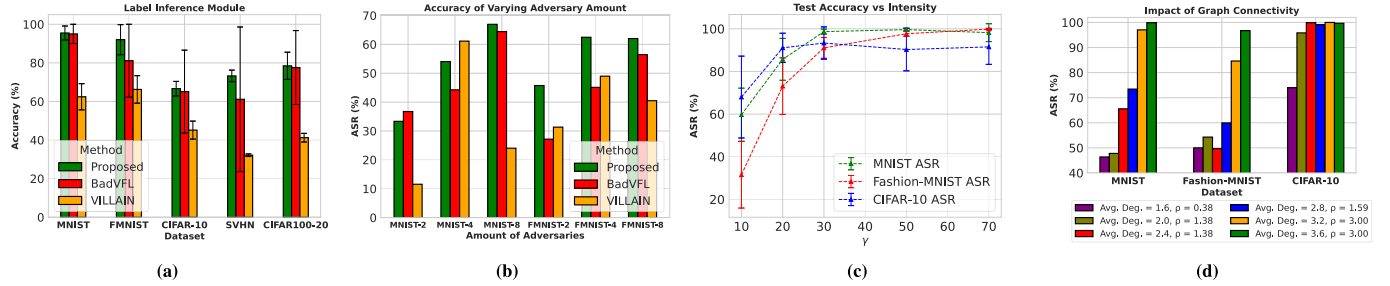

Fig. 5. (a) The accuracy of label inference across all three scenarios. The proposed method reaches higher accuracy than the baselines (BadVFL [34] and VILLAIN [35]) even without access to server information. (b) Impact of varying number of adversaries with Fashion-MNIST and MNIST. As the number of adversaries increases, a general trend in the increase of the ASR is noticed. In addition, the proposed attack is consistently has a higher or comparable ASR value to the baselines. (c) Varying trigger intensity parameter γ. We notice that an increase in γ allows for a greater success of a backdoor attack. (d) Performance of the ASR with different levels of graph connectivity. In general, the more connected the graph is, the better the performance of the backdoor attack.

TABLE III

IMPACT OF ADDING GAUSSIAN NOISE TO THE GRADIENTS AS A SERVER-SIDE DEFENSE: OUR PROPOSED METHOD IS MUCH MORE RESISTANT TO THE DEFENSE DUE TO NOT RELYING ON THE SERVER FOR LABEL INFORMATION. THE BASELINES, PARTICULARLY VILLAIN, EXPERIENCE A SIGNIFICANT DROP IN ASR WHEN NOISE IS INTRODUCED TO THE GRADIENTS

Task	MNIST			FMNIST			CIFAR-10		
	Proposed	BadVFL [34]	VILLAIN [35]	Proposed	BadVFL [34]	VILLAIN [35]	Proposed	BadVFL [34]	VILLAIN [35]
CDA	97.96 ± 0.18	97.91 ± 0.15	97.87 ± 0.07	88.54 ± 0.08	88.56 ± 0.18	88.86 ± 0.11	58.24 ± 0.5	58.62 ± 0.46	58.06 ± 0.32
CDA w/ Noise	97.97 ± 0.19	97.83 ± 0.2	97.51 ± 0.15	88.25 ± 0.41	88.07 ± 0.43	85.98 ± 3.0	47.84 ± 3.63	37.12 ± 14.18	33.25 ± 13.48
ASR	94.89 ± 3.76	36.53 ± 35.9	53.29 ± 12.78	88.23 ± 4.12	63.40 ± 15.73	41.57 ± 21.69	93.95 ± 6.86	36.45 ± 35.8	67.10 ± 16.78
ASR w/ Noise	$\mathbf{86.88} \pm 4.03$	25.88 ± 16.97	0.66 ± 0.38	$\mathbf{81.47} \pm 7.26$	38.64 ± 16.07	1.33 ± 1.33	$\mathbf{89.50} \pm 7.16$	9.38 ± 15.23	2.26 ± 2.36

door task is slowly forgotten over continuous iterations [63], whereas, our proposed attack consistently performs well.

Label Inference: We compare the accuracy of our label inference to the baselines [34], [35] in Fig. 5a. We see that the proposed method of utilizing a combination of metric learning and VAEs for label inference has the best performance in terms of label inference accuracy across the datasets, while not relying on the server information. Note that the proposed attack can still be quite successful (Fig. 4) even when the label inference accuracy is not very high (e.g., with the CIFAR-10 and SVHN datasets). This indicates that some margin of error in label inference can be acceptable, provided that the target data points significantly outnumber the combined total of other labels. As in Fig. 4, the inclusion of these five datasets validates the label inference accuracy module of our methodology across a wide range of learning scenarios. Specifically, while our proposed method achieves high label inference accuracy on benchmark datasets such as CIFAR-10 and MNIST, the results on the CIFAR100-20 dataset further demonstrate its robustness and scalability on more complex and diverse settings.

Robustness against Defense: We evaluate the effectiveness of our attack against traditional server-side defense mechanisms, specifically noising defenses [34], [64]. We insert noise with variance 10^{-10} in the gradients sent back by the server, and assess the corresponding ASR values averaged over several runs in Table III. We note that the proposed method maintains relatively small degradation in ASR performance compared to those obtained without any defenses, indicating the method's robustness to such strategies. By contrast, the ASR values drop significantly in presence of noise-injected gradients for the competing baselines. The reason behind this is that the baselines both rely on server gradients to construct the attack: noise addition affects the similarity comparison of the baselines, leading to poor label inference. Moreover, the server can employ noising without significantly affecting the CDA, suggesting that it can defend against baselines [34], [35], but remains vulnerable to our method. Overall, these results validate the robustness of our label inference based on hybrid VAEs rather than server gradients, which is crucial to maintaining a good ASR in the presence of such defenses.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

LEE et al.: COOPERATIVE DECENTRALIZED BACKDOOR ATTACKS ON VERTICAL FEDERATED LEARNING 11

TABLE IV

STUDY ON THE EFFECTIVENESS OF THE PROPOSED VAE-BASED SWAPPING METHOD FOR TRIGGER EMBEDDING: OUR PROPOSED METHOD IS MUCH MORE RESISTANT TO FORGETTING THE BACKDOOR TASK DUE TO THE SWAPPING OF CLEAN FEATURES WITH SIMILAR BUT HARDER-TO-DISTINGUISH VAE-GENERATED FEATURES. MOREOVER, THE CDA STILL REMAINS HIGH, EVEN WITH THE SWAPPING, INDICATING THAT IT IS STILL LEARNING THE OVERALL STRUCTURE OF THE CLEAN FEATURES PROPERLY. LASTLY, COMPARING THE ACCURACY OF THE NO ATTACK SCENARIO, WE NOTE THAT THE ACCURACY DOES NOT SIGNIFICANTLY DIFFER FROM THE CDA, INDICATING THE VAE-BASED SWAPPING METHOD SUCCESSFULLY KEEPS BOTH THE ASR AND CDA HIGH. (FOR THE NO ATK. COLUMN, NO RESULTS ARE GIVEN FOR THE VAE-BASED SWAP ROW AS NO ATTACK TAKES PLACE)

Task	MNIST		FMNIST		CIFAR-10		SVHN	
	VAE-based Swap	No Swap	VAE-based Swap	No Swap	VAE-based Swap	No Swap	VAE-based Swap	No Swap
ASR-1	$\mathbf{93.35 \pm 6.91}$	32.97 ± 18.48	$\mathbf{82.44 \pm 4.01}$	46.46 ± 11.51	$\mathbf{97.65 \pm 2.49}$	12.13 ± 8.01	$\mathbf{91.97 \pm 3.42}$	40.58 ± 9.00
CDA-1	97.94 ± 0.12	97.82 ± 0.12	88.39 ± 0.58	88.23 ± 0.24	58.59 ± 0.17	58.31 ± 0.21	70.23 ± 2.79	70.20 ± 1.78
ASR-2	$\mathbf{73.67 \pm 33.50}$	0.33 ± 0.67	$\mathbf{78.61 \pm 22.33}$	26.34 ± 49.15	$\mathbf{94.95 \pm 8.12}$	0.33 ± 0.43	$\mathbf{81.53 \pm 17.22}$	5.30 ± 9.67
CDA-2	98.01 ± 0.19	97.86 ± 0.10	88.51 ± 0.13	88.48 ± 0.23	58.45 ± 0.57	58.52 ± 0.37	66.65 ± 3.68	69.83 ± 9.13
No Atk.	—	97.84 ± 0.30	—	86.96 ± 2.14	—	58.99 ± 0.47	—	71.24 ± 9.07

TABLE V

STUDY ON THE EFFECTIVENESS OF THE PROPOSED ATTACK IN TERMS OF ASR WITH AND WITHOUT THE USE OF THE CONSENSUS-VOTING SYSTEM UTILIZING TRIGGER EMBEDDING METHOD 2 FROM FIG. 3. NOTE THAT MANY OF THE VOTING COLUMN RESULTS ARE THE SAME AS ASR-2 AND CDA-2 WITH VAE-BASED SWAPPING IN TABLE IV DUE TO BOTH BEING RUN ON THE SAME SETTINGS AND HYPERPARAMETERS

Task	MNIST		FMNIST		CIFAR-10		SVHN	
	Voting	No Vote	Voting	No Vote	Voting	No Vote	Voting	No Vote
ASR	$\mathbf{73.67 \pm 33.50}$	0.22 ± 0.45	$\mathbf{78.61 \pm 22.33}$	25.0 ± 50.0	$\mathbf{98.79 \pm 1.48}$	34.45 ± 38.86	$\mathbf{74.00 \pm 48.72}$	24.00 ± 45.40
CDA	98.01 ± 0.19	97.84 ± 0.30	88.51 ± 0.13	86.96 ± 2.14	58.70 ± 0.94	59.19 ± 0.84	66.28 ± 3.57	69.50 ± 7.74

C. Varying Adversaries and Attack Network

Impact of Varying Number of Adversaries: Now, we analyze the impact of the number of adversaries on the ASR on the server's top model. For this experiment, we utilize the trigger embedding method of splitting the trigger into smaller separate local subtriggers (i.e., Method 2 from Fig. 3). We assume that each adversary holds an 8×2 trigger when there are two adversaries, 4×2 triggers when there are four, and 2×2 triggers when there are eight, thus all have the same total area of 32.

The results are given in Fig. 5b. We see a general trend where an increase in the number of adversaries leads to a higher ASR, suggesting that more adversaries result in a stronger attack, even when the area threshold parameter ϵ remains unchanged. We also observed (not shown) that varying the number of adversaries did not significantly affect the CDA, indicating that the presence of multiple adversaries does not impact the main task across different methods.

Impact of Trigger Intensity: We next analyze the impact of the intensity value γ on the ASR. The results across three datasets are shown in 5c. The analysis reveals that a stronger trigger produces a higher ASR, with the top-performing server model showing a stronger association to the target as the trigger's intensity increases. Notably, while MNIST and CIFAR-10 achieve relatively good performance even at lower γ values, Fashion-MNIST requires a higher trigger intensity to attain desirable ASR accuracy. This is likely due to a significant portion of Fashion-MNIST datapoints (being covered in white), matching the background of the trigger, thus necessitating a stronger trigger to differentiate from the clean features.

Graph Connectivity: Next, we investigate how the algebraic connectivity ρ of the adversary graph affects the attack performance. We simulate this by progressively increasing the average degree of the graph, beginning from a line topology,

TABLE VI

STUDY OF THE EFFECTIVENESS OF THE ATTACK IN TERMS OF ASR WITH AND WITHOUT THE USE OF TRIPLET LOSS ON THE ADVERSARIAL VAEs

Architecture	Dataset		
	MNIST	FMNIST	CIFAR-10
VAE Only	84.00 ± 9.78	89.10 ± 10.81	76.18 ± 11.93
Hybrid VAE & Triplet	$\mathbf{89.23 \pm 6.09}$	$\mathbf{91.74 \pm 4.61}$	$\mathbf{92.72 \pm 9.38}$

and consider Method 2 from Fig. 3 for trigger embedding. The results are given in Fig. 5d. We see that the attack efficacy tends to get enhanced (i.e., ASR increases) with the increase of ρ, indicating that $\delta(\rho)$ from Theorem 1 is increasing in ρ. This is because the adversaries receive a higher share of the features for higher ρ. Note that due to the increased complexity of features in CIFAR10, the server becomes more reliant on the trigger, meaning a lower ρ is sufficient to achieve a high ASR.

Ablation Studies: We conduct several ablation studies. First, we explore the impact of incorporating the triplet loss into the VAE for label inference in (4), i.e., whether it results in a higher attack potency. The corresponding results are shown in Table VI, which demonstrate significant improvement in ASR values for all three datasets compared to the VAE-only loss. Additionally, we note that the CIFAR-10 dataset experiences a significantly larger enhancement compared to MNIST and Fashion-MNIST datasets in terms of ASR: the datapoints in CIFAR10 involve a more complex structure, where triplet loss may play a crucial role in refining embedding quality, facilitating a more effective attack. Overall, our findings validate our choice of hybrid VAE and metric learning for improving the attack performance.

In addition, we investigate the addition of the VAE-generated datapoint swapping mechanism in (5) to investigate whether it results in better overall attack performance for both

Methods 1 (ASR-1 and CDA-1) and 2 (ASR-2 and CDA-2) outlined in Fig. 3. For Method 2, each adversary adopts a subtrigger size of 4×2 for a total area of 40. Given the results in Table IV, we show that the VAE-based swapping of original samples with VAE-generated samples $\widetilde{x}_m^{(i)}$ (VAE-based Swap in Table IV) plays a crucial role in the success of the attack for both Methods 1 and 2. When no swapping mechanism is utilized (No Swap in Table IV), the ASR is significantly lower ($\approx 85\%$ and above with CIFAR-10) compared to when VAE-based swapping takes place, resulting in a failed attack. We also note that Method 1 of trigger embedding is more stable in its attack performance, with less variation in the averages compared to the attack utilized by Method 2. This is because the location where each adversary places the subtrigger for Method 2 varies instead being centered like Method 1, making it harder for the top model to learn the pattern. This means that having a denser adversary graph allows for better attack facilitation, as having a complete graph allows for the usage of Method 1. However, both methods still overall achieve high attack success rates across all four datasets. Moreover, when comparing the CDA of the proposed method to when no attack takes place, the accuracies are similar, indicating that the VAE generated samples still allow the top model to learn the general clean features of the target label well in addition to the trigger.

Finally, we also investigate the importance of the addition of majority voting in the label inference module by comparing the proposed method to no voting taking place. When no vote takes place, each adversary embeds its local trigger partition based off its local label inference results. Looking at the results presented in Table V, we notice a significant decrease in ASR performance when only local voting is considered on each adversary device, highlighting the importance of the majority voting module in the overall effectiveness of the attack. This is because no consensus has been reached on which final datapoints to poison, and so often the trigger implanted for each sample is incomplete, making it harder for the top model to fully learn the trigger pattern. Moreover, this also means that it is often the case that only a portion of the poisoned adversary-owned features is swapped out with VAE-generated sample $\widetilde{x}_m^{(i)}$, making it less likely for the server to rely on the trigger for classification due to the presence of more unchanged features, thereby resulting in a failed attack.

Effectiveness of Attack under Differing Latent Sizes: We analyze whether the latent embedding size that the server requires each local bottom model to send affects the overall effectiveness of the proposed attack, as outlined in Table VII. We note that no matter the size of the embedding required by the server, the attack in terms of the ASR remains high. This means that the proposed attack can accommodate and learn the trigger with both small and large latent space dimensions. Moreover, when an increase in the latent space dimension improves the overall main learning task (CDA), as is the case with SVHN, the ASR remains largely unchanged. Overall, this indicates the robustness of the proposed attack to varying embedding sizes required by the server.

Effectiveness of Attack under Differing Margins: Next, we analyze whether the choice of the margin value $\widehat{\kappa}$ when training the triplet-based VAE affects the overall performance

TABLE VII

CDA AND ASR EVALUATION METRICS FOR DIFFERING LATENT SPACE DIMENSIONS OF THE EMBEDDINGS PRODUCED BY THE BOTTOM MODELS ON EACH CLIENT. FOR BOTH SMALL AND LARGE LATENT EMBEDDING VALUES, THE TRIGGER PATTERN IS SUCCESSFULLY INJECTED INTO THE LEARNING PROCESS OF THE SERVER'S TOP MODEL, RESULTING IN A SUCCESSFUL BACKDOOR ATTACK

Dimension	Datasets	ASR	CDA
32	MNIST	89.23 ± 6.09	97.92 ± 0.15
	FMNIST	96.34 ± 2.32	88.33 ± 0.09
	CIFAR-10	95.74 ± 4.21	56.37 ± 0.31
	SVHN	94.07 ± 5.98	45.02 ± 9.60
64	MNIST	87.07 ± 9.65	98.09 ± 0.12
	FMNIST	98.53 ± 1.13	88.33 ± 0.28
	CIFAR-10	97.55 ± 2.43	57.55 ± 0.10
	SVHN	90.50 ± 10.36	52.29 ± 13.63
128	MNIST	86.49 ± 7.28	98.15 ± 0.06
	FMNIST	99.65 ± 0.22	87.33 ± 2.66
	CIFAR-10	96.03 ± 2.16	57.44 ± 0.72
	SVHN	86.96 ± 7.87	60.28 ± 7.88
256	MNIST	86.93 ± 12.40	98.02 ± 0.16
	FMNIST	97.70 ± 3.15	88.45 ± 0.14
	CIFAR-10	95.65 ± 5.14	58.24 ± 0.44
	SVHN	87.33 ± 10.63	69.85 ± 3.06
512	MNIST	95.02 ± 2.69	98.18 ± 0.08
	FMNIST	98.10 ± 0.53	87.72 ± 2.36
	CIFAR-10	97.38 ± 1.44	59.50 ± 0.71
	SVHN	93.29 ± 4.37	71.57 ± 1.92

of the backdoor attack, as outlined in Table VIII. We note that in all data sets, the choice of $\widehat{\kappa}$ does not significantly impact both the CDA and ASR. This could be attributed to the use of an online batch-hard triplet mining strategy during training (Sec. III-A), meaning difficult negatives are always used in the training of the adversarial VAEs, no matter the margin value chosen. However, some margin values, such as 0.3 vs 0.15 for SVHN or 0.25 vs 0.40 for FMNIST, perform better than others. In addition, with the case of SVHN, we note that arbitrarily selecting too high a margin value results in a $\approx 9\%$ drop in CDA performance. This might happen if the latent space is not well-separated, leading to many samples of differing classes to be selected. This could possibly create a lack of consistency in the type of data points that are trigger-embedded, in addition to the types of samples the VAE generates, leading to a degradation of the CDA. Overall, these findings indicate that while the margin is not a highly sensitive parameter that requires meticulous tuning, some consideration should be taken when selecting an ideal value for maximal attack potency and main task performance.

Function Analysis of $\delta(\rho)$: Finally, we examine the relationship between $\delta(\rho)$ and ρ from our analysis in Sec. V. We compute δ as the L2 norm difference between the gradients used to update the top model under attack and under benign conditions. To enable broader analysis of connectivity over a larger range of ρ, we resize the samples [65], [66], [67], [68] to 64×64. This adjustment allows for accommodating 20 total clients, including 10 adversaries, thereby supporting a denser adversary graph. Starting with a line topology, we incrementally add edges until the network becomes fully connected. Each adversary adopts a subtrigger size of 2×2

TABLE VIII
CDA AND ASR EVALUATION METRICS ON DIFFERENT $\hat{\kappa}$ MARGIN VALUES DURING TRIPLET MARGIN LOSS (3) FOR LABEL INFERENCE WITH THE ADVERSARIAL VAEs. OVERALL, THE ATTACK IN TERMS OF THE ASR REMAINS HIGH ACROSS ALL DATASETS, BUT SOME MARGIN VALUES PERFORM BETTER THAN OTHERS AND ARE MORE STABLE IN THEIR PERFORMANCE. IN ADDITION, WITH THE EXCEPTION OF SVHN, THE CDA STAYS MOSTLY THE SAME AND IS UNAFFECTED BY THE MARGIN VALUE ADOPTED BY THE ADVERSARIES

| Margin ($\hat{\kappa}$)	MNIST		FMNIST		CIFAR-10		SVHN	
	ASR	CDA	ASR	CDA	ASR	CDA	ASR	CDA
0.10	91.13 ± 5.78	97.94 ± 0.10	94.67 ± 1.96	88.34 ± 0.53	95.82 ± 2.73	58.13 ± 0.32	91.97 ± 2.66	65.83 ± 14.49
0.15	91.36 ± 2.58	97.93 ± 0.10	88.12 ± 3.11	88.41 ± 0.42	92.37 ± 4.76	58.70 ± 0.18	81.44 ± 10.97	69.50 ± 3.64
0.2	87.30 ± 9.62	97.91 ± 0.17	91.59 ± 2.27	88.23 ± 0.50	97.76 ± 1.09	57.98 ± 0.61	89.28 ± 7.76	68.48 ± 1.16
0.25	86.17 ± 11.38	98.05 ± 0.05	85.10 ± 16.31	88.63 ± 0.17	87.87 ± 9.08	58.49 ± 0.43	89.14 ± 6.15	71.33 ± 5.90
0.30	92.54 ± 1.17	97.89 ± 0.12	87.99 ± 7.12	88.85 ± 0.17	91.16 ± 3.83	58.67 ± 0.17	92.15 ± 3.56	73.81 ± 3.06
0.35	90.38 ± 6.52	98.02 ± 0.22	95.46 ± 4.89	86.23 ± 3.35	91.97 ± 7.88	58.89 ± 0.96	83.15 ± 7.27	72.97 ± 5.87
0.40	89.23 ± 6.09	97.92 ± 0.15	96.14 ± 4.90	87.89 ± 0.35	94.90 ± 3.56	58.54 ± 0.20	95.16 ± 2.90	64.69 ± 5.31
0.45	86.39 ± 9.05	97.98 ± 0.12	94.99 ± 2.99	88.24 ± 0.28	97.00 ± 2.18	58.46 ± 0.53	90.52 ± 6.18	69.20 ± 6.50
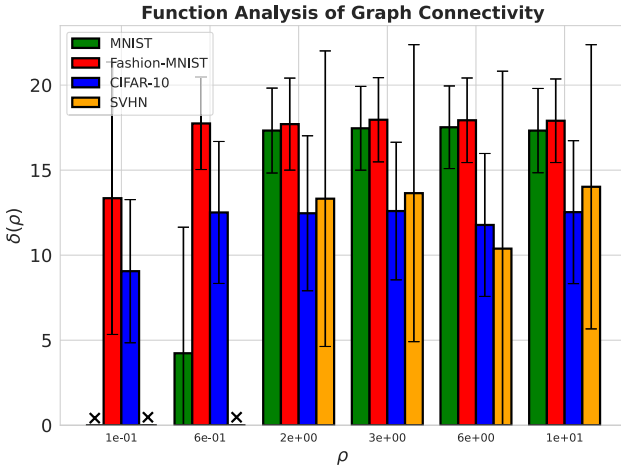
Fig. 6. The gradient perturbation in the top model from the adversaries for differing levels of connectivity ρ. This validates the hypothesis in Sec. V that $\delta(\rho)$ is a non-decreasing function of ρ. As ρ increases, the perturbation saturates quickly. When no perturbation takes place, it is marked with an "x", as seen for some datasets when $\rho \in \{0.1, 0.6\}$.

with a total trigger area of 40. For CIFAR-10 and SVHN, the VAE architecture is adjusted to a 2-layer encoder and decoder CVAE.

The results are shown for each dataset in Fig. 6. We see that as the connectivity ρ increases, the gradient perturbation introduced by the attack increases until it saturates once a certain level of connectivity is reached. When connectivity is lower, the number of poisoned datapoints is lower, resulting in a smaller disturbance during the training process. In addition, the poisoning budget ζ set by the adversaries prevents excessive poisoning, limiting the amount of perturbation presented to the top model. Moreover, we note that the perturbation increases more dramatically when the connectivity is lower, as individual adversaries benefit more from receiving features from other adversary devices when their shared knowledge is more limited. The function saturates rather quickly, highlighting the potency of the attack even for relatively sparsely connected graphs (i.e., $\rho \approx 1$). Therefore, if the adversaries decide to utilize Method 2 (Fig. 3) for their attack, a fully-connected graph is not necessary. Finally, we remark that when dealing with low connectivity values for MNIST and SVHN (i.e.,

$\rho = 0.1$), no datapoints are inferred via the consensus voting, leading to no datapoints being poisoned, i.e., no perturbation.

VII. CONCLUSION

In this paper, we introduced a novel methodology for conducting backdoor attacks in cross-device VFL environments. Our method considers a hybrid VAE and metric learning approach for label inference, and exploits the available graph topology among adversaries for cooperative trigger implantation. We theoretically analyzed VFL convergence behavior under backdooring, and showed that the server model would have a stationarity gap proportional to the level of adversarial gradient perturbation. Our numerical experiments showed that the proposed method surpasses existing baselines in label inference accuracy and attack performance across various datasets, while also exhibiting increased resilience to server-side defenses.

APPENDIX

A. Proof of Theorem 1

First, we denote the mini-batch gradient of the loss function as

$$\nabla_{\theta^{(t)}} \mathcal{L} = [(\nabla_{\theta_1^{(t)}} \mathcal{L})^\mathsf{T}, (\nabla_{\theta_2^{(t)}} \mathcal{L})^\mathsf{T}, \ldots, (\nabla_{\theta_K^{(t)}} \mathcal{L})^\mathsf{T}]^\mathsf{T}.$$

Due to the adversarial attacks, the gradients used for the updates will be perturbed. We denote the perturbed gradient as

$$\nabla_{\theta^{(t)}}^a \mathcal{L} = [(\nabla_{\theta_1^{(t)}}^a \mathcal{L})^\mathsf{T}, (\nabla_{\theta_2^{(t)}}^a \mathcal{L})^\mathsf{T}, \ldots, (\nabla_{\theta_K^{(t)}}^a \mathcal{L})^\mathsf{T}]^\mathsf{T},$$

which is further used for model update in VFL, i.e.,

$$\theta_k^{(t+1)} = \theta_k^{(t)} - \eta^{(t)} \nabla_{\theta_k^{(t)}}^a \mathcal{L}.$$

Hence, the update of the whole model can be expressed as

$$\theta^{(t+1)} - \theta^{(t)} = -\eta^{(t)} [(\nabla_{\theta_1^{(t)}} \mathcal{L})^\mathsf{T}, (\nabla_{\theta_2^{(t)}} \mathcal{L})^\mathsf{T}, \ldots, (\nabla_{\theta_K^{(t)}} \mathcal{L})^\mathsf{T}]^\mathsf{T}$$
$$- \underbrace{[\ldots, (\nabla_{\theta_k^{(t)}}^a \mathcal{L})^\mathsf{T} - (\nabla_{\theta_k^{(t)}} \mathcal{L})^\mathsf{T}, \ldots]^\mathsf{T}}_{\Delta^{(t)}}.$$

From Assumption 1 and the above iterative equation, we have

$$F(\theta^{(t+1)})$$
$$\leq F(\theta^{(t)}) + \langle \nabla F(\theta^{(t)}), \theta^{(t+1)} - \theta^{(t)} \rangle + \frac{L}{2} \|\theta^{(t+1)} - \theta^{(t)}\|^2$$

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

14

IEEE TRANSACTIONS ON NETWORKING

$$\leq F(\theta^{(t)}) - \langle \nabla F(\theta^{(t)}), \eta^{(t)} \nabla_{\theta^{(t)}} \mathcal{L} + \eta^{(t)} \Delta^{(t)} \rangle$$
$$+ \frac{L}{2} \| \eta^{(t)} \nabla_{\theta^{(t)}} \mathcal{L} + \eta^{(t)} \Delta^{(t)} \|^2. \tag{7}$$

Next, from Assumption 2, we have $\mathbb{E}[\nabla_{\theta_k} \mathcal{L}] = \mathbb{E}[\nabla_{\theta_k} F(\theta)]$. Therefore, taking expectation over (7) leads us to

$$\mathbb{E}[F(\theta^{(t+1)})]$$
$$\leq F(\theta^{(t)}) - \eta^{(t)} \| \nabla F(\theta^{(t)}) \|^2 + (\eta^{(t)})^2 L \mathbb{E} \| \Delta^{(t)} \|^2$$
$$- \eta^{(t)} \mathbb{E} \langle \nabla F(\theta^{(t)}), \Delta^{(t)} \rangle + (\eta^{(t)})^2 L \mathbb{E} \| \nabla_{\theta^{(t)}} \mathcal{L} \|^2$$
$$\leq F(\theta^{(t)}) - \eta^{(t)} \| \nabla F(\theta^{(t)}) \|^2 + \frac{1}{2} \eta^{(t)} \| \nabla F(\theta^{(t)}) \|^2$$
$$+ \frac{1}{2} \eta^{(t)} \mathbb{E} \| \Delta_t \|^2 + (\eta^{(t)})^2 L \mathbb{E} \| \nabla_{\theta^{(t)}} \mathcal{L} \|^2$$
$$+ (\eta^{(t)})^2 L \mathbb{E} \| \Delta^{(t)} \|^2 \tag{8}$$
$$\leq F(\theta^{(t)}) - \left(\eta^{(t)} - \frac{1}{2} \eta^{(t)} - (\eta^{(t)})^2 L \right) \| \nabla F(\theta^{(t)}) \|^2$$
$$+ (\eta^{(t)})^2 L \mathbb{E} \| \nabla_{\theta^{(t)}} \mathcal{L} - \nabla F(\theta^{(t)}) \|^2$$
$$+ \frac{1}{2} (\eta^{(t)}) \mathbb{E} \| \Delta_t \|^2 + (\eta^{(t)})^2 L \mathbb{E} \| \Delta^{(t)} \|^2$$
$$\leq F(\theta^{(t)}) - \left((\eta^{(t)}) - \frac{1}{2} \eta^{(t)} - (\eta^{(t)})^2 L \right) \| \nabla F(\theta^{(t)}) \|^2$$
$$+ (\eta^{(t)})^2 K L \Gamma + \frac{1}{2} \eta^{(t)} K \delta(\rho) + (\eta^{(t)})^2 K L \delta(\rho), \tag{9}$$

where we utilize $\langle c_1, c_2 \rangle \leq \frac{\|c_1\|^2}{2} + \frac{\|c_2\|^2}{2}$ [52] to obtain (8), and Assumptions 2 and 3 to obtain (9). Now, letting $\eta^{(t)} \leq \frac{1}{4L}$, we have $\eta^{(t)} - \frac{1}{2} \eta^{(t)} - (\eta^{(t)})^2 L \geq \frac{\eta^{(t)}}{4}$. Hence, (9) can be rewritten as

$$\frac{\eta^{(t)}}{4} \| \nabla F(\theta^{(t)}) \|^2$$
$$\leq F(\theta^{(t)}) - \mathbb{E}[F(\theta^{(t+1)})] + (\eta^{(t)})^2 K L \Gamma$$
$$+ \frac{1}{2} \eta^{(t)} K \delta(\rho) + (\eta^{(t)})^2 K L \delta(\rho). \tag{10}$$

Taking expectation of the above inequality over $\theta^{(t)}$, we have

$$\eta^{(t)} \mathbb{E} \| \nabla F(\theta^{(t)}) \|^2$$
$$\leq 4 \mathbb{E}[F(\theta^{(t)})] - 4 \mathbb{E}[F(\theta^{(t+1)})] + 4(\eta^{(t)})^2 K L \Gamma$$
$$+ 4(\eta^{(t)})^2 L K \delta(\rho) + 2 \eta^{(t)} K \delta(\rho).$$

Summing the above inequality from $t = 0$ to $T - 1$ and utilizing the fact that the loss function is non-negative, i.e., $\mathbb{E}[F(\theta^T)] \geq 0$, we have

$$\sum_{t=0}^{T-1} \eta^{(t)} \mathbb{E} \| \nabla F(\theta^{(t)}) \|^2 \leq 4 F(\theta^0) + 4 \left(\sum_{t=0}^{T-1} (\eta^{(t)})^2 \right) K L \Gamma$$
$$+ 4 \left(\sum_{t=0}^{T-1} (\eta^{(t)})^2 \right) L K \delta(\rho) + 2 \left(\sum_{t=0}^{T-1} \eta^{(t)} \right) K \delta(\rho). \tag{11}$$

Dividing the both sides by $\sum_{t=0}^{T-1} \eta^{(t)}$ leads us to

$$\sum_{t=0}^{T-1} \frac{\eta^{(t)}}{\sum_{t=0}^{T-1} \eta^{(t)}} \mathbb{E} \| \nabla F(\theta^{(t)}) \|^2$$
$$\leq 4 \frac{F(\theta^0)}{\sum_{t=0}^{T-1} \eta^{(t)}} + 4 \left(\sum_{t=0}^{T-1} (\eta^{(t)}) \right) K L \Gamma$$

$$+ 4 \left(\sum_{t=0}^{T-1} (\eta^{(t)}) \right) L K \delta(\rho) + 2 K \delta(\rho). \tag{12}$$

Furthermore, as $\min_{t=0,\dots,T-1} z_t \leq \frac{\omega_t}{\sum_{t=0}^{T-1} \omega_t} z_t$, we thus complete the proof of Theorem 1.

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. 20th Int. Conf. Artif. Intell. Statist.*, vol. 54, 2017, pp. 1273–1282.

[2] N. Bouacida and P. Mohapatra, "Vulnerabilities in federated learning," *IEEE Access*, vol. 9, pp. 63229–63249, 2021.

[3] A. N. Bhagoji, S. Chakraborty, P. Mittal, and S. Calo, "Analyzing federated learning through an adversarial lens," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 634–643.

[4] Y. Ma, X. Zhu, and J. Hsu, "Data poisoning against differentially-private learners: Attacks and defenses," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, 2019, pp. 4732–4738.

[5] H. Masuda, K. Kita, Y. Koizumi, J. Takemasa, and T. Hasegawa, "Model fragmentation, shuffle and aggregation to mitigate model inversion in federated learning," in *Proc. IEEE Int. Symp. Local Metrop. Area Netw. (LANMAN)*, Jul. 2021, pp. 1–6.

[6] W. Issa, N. Moustafa, B. Turnbull, and K.-K.-R. Choo, "RVE-PFL: Robust variational encoder-based personalized federated learning against model inversion attacks," *IEEE Trans. Inf. Forensics Security*, vol. 19, pp. 3772–3787, 2024.

[7] E. Bagdasaryan, A. Veit, Y. Hua, D. Estrin, and V. Shmatikov, "How to backdoor federated learning," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2020, pp. 2938–2948.

[8] K. Wei et al., "Vertical federated learning: Challenges, methodologies and experiments," 2022, *arXiv:2202.04309*.

[9] D. Romanini et al., "PyVertical: A vertical federated learning framework for multi-headed SplitNN," 2021, *arXiv:2104.00489*.

[10] T. M. Mengistu, T. Kim, and J.-W. Lin, "A survey on heterogeneity taxonomy, security and privacy preservation in the integration of IoT, wireless sensor networks and federated learning," *Sensors*, vol. 24, no. 3, p. 968, Feb. 2024.

[11] R. Yang, J. Ma, J. Zhang, S. Kumari, S. Kumar, and J. J. P. C. Rodrigues, "Practical feature inference attack in vertical federated learning during prediction in artificial Internet of Things," *IEEE Internet Things J.*, vol. 11, no. 1, pp. 5–16, Jan. 2024.

[12] X. Luo, Y. Wu, X. Xiao, and B. C. Ooi, "Feature inference attack on model predictions in vertical federated learning," in *Proc. IEEE 37th Int. Conf. Data Eng. (ICDE)*, Apr. 2021, pp. 181–192.

[13] C. Fu et al., "Label inference attacks against vertical federated learning," in *Proc. 31st USENIX Security Symp. (USENIX Security)*, 2022, pp. 1397–1414.

[14] C. Song and V. Shmatikov, "Overlearning reveals sensitive attributes," in *Proc. Int. Conf. Learn. Represent.*, 2020. [Online]. Available: https://openreview.net/forum?id=SJeNz04tDS

[15] X. Lyu et al., "Poisoning with cerberus: Stealthy and colluded backdoor attack against federated learning," in *Proc. AAAI Conf. Artif. Intell.*, Jun. 2023, vol. 37, no. 7, pp. 9020–9028.

[16] J. Luo, Q. Dong, D. Huang, and M. Kang, "Attribute based encryption for information sharing on tactical mobile networks," in *Proc. MILCOM - IEEE Mil. Commun. Conf. (MILCOM)*, Oct. 2018, pp. 1–9.

[17] D. Pratama et al., "Behind the wings: The case of reverse engineering and drone hijacking in DJI enhanced Wi-Fi protocol," in *Proc. Int. Conf. Platform Technol. Service (PlatCon)*, Aug. 2024, pp. 127–132.

[18] I. Gulatas, H. H. Kilinc, A. H. Zaim, and M. A. Aydin, "Malware threat on edge/fog computing environments from Internet of Things devices perspective," *IEEE Access*, vol. 11, pp. 33584–33606, 2023.

[19] S. Wagle, A. B. Das, D. J. Love, and C. G. Brinton, "Multi-agent reinforcement learning for graph discovery in D2D-enabled federated learning," *IEEE J. Sel. Topics Signal Process.*, early access, Apr. 21, 2025, doi: 10.1109/JSTSP.2025.3560914.

[20] S. Lee, A. B. Das, S. Wagle, and C. G. Brinton, "Smart information exchange for unsupervised federated learning via reinforcement learning," in *Proc. IEEE Int. Conf. Commun.*, Jun. 2024, pp. 3494–3499.

[21] H. Xing, O. Simeone, and S. Bi, "Decentralized federated learning via SGD over wireless D2D networks," in *Proc. IEEE 21st Int. Workshop Signal Process. Adv. Wireless Commun. (SPAWC)*, May 2020, pp. 1–5.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

LEE et al.: COOPERATIVE DECENTRALIZED BACKDOOR ATTACKS ON VERTICAL FEDERATED LEARNING 15

[22] F. Lin, Q. Ling, and Z. Xiong, "Byzantine-resilient distributed large-scale matrix completion," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2019, pp. 8167–8171.

[23] C. Fung, C. J. M. Yoon, and I. Beschastnikh, "Mitigating Sybils in federated learning poisoning," 2018, *arXiv:1808.04866*.

[24] H. Wang et al., "Attack of the tails: Yes, you really can backdoor federated learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 16070–16084.

[25] J. Cao and L. Zhu, "A highly efficient, confidential, and continuous federated learning backdoor attack strategy," in *Proc. 14th Int. Conf. Mach. Learn. Comput. (ICMLC)*, Feb. 2022, pp. 18–27.

[26] C. Xie, K. Huang, P. Chen, and B. Li, "DBA: Distributed backdoor attacks against federated learning," in *Proc. Int. Conf. Learn. Represent.*, 2020. [Online]. Available: https://openreview.net/forum?id=rkgyS0VFvr

[27] T. D. Nguyen et al., "FLAME: Taming backdoors in federated learning," in *Proc. USENIX Secur. Symp.*, 2022, pp. 1415–1432.

[28] C. Fung, C. J. M. Yoon, and I. Beschastnikh, "The limitations of federated learning in Sybil settings," in *Proc. Int. Symp. Res. Attacks*, 2020, pp. 301–316.

[29] K. Hsieh, A. Phanishayee, O. Mutlu, and P. B. Gibbons, "The non-IID data quagmire of decentralized machine learning," in *Proc. Int. Conf. Mach. Learn.*, vol. 1, 2019, pp. 4387–4398.

[30] X. Han et al., "Badsfl: Backdoor attack against scaffold federated learning," 2024, *arXiv:2411.16167*.

[31] S. P. Karimireddy, S. Kale, M. Mohri, S. J. Reddi, S. U. Stich, and A. T. Suresh, "SCAFFOLD: Stochastic controlled averaging for federated learning," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 5132–5143.

[32] I. Goodfellow et al., "Generative adversarial networks," *Commun. ACM*, vol. 63, no. 11, pp. 139–144, 2020.

[33] T. Gu, K. Liu, B. Dolan-Gavitt, and S. Garg, "BadNets: Evaluating backdooring attacks on deep neural networks," *IEEE Access*, vol. 7, pp. 47230–47244, 2019.

[34] Y. Xuan, X. Chen, Z. Zhao, B. Tang, and D. Ye, "Practical and general backdoor attacks against vertical federated learning," in *Proc. Joint Eur. Conf. Mach. Learn. Knowl. Discovery Databases*, 2023, pp. 402–417.

[35] Y. Bai, Y. Chen, H. Zhang, W. Xu, H. Weng, and D. Goodman, "VILLAIN: Backdoor attacks against vertical split learning," in *Proc. USENIX Secur. Symp.*, 2023, pp. 2743–2760.

[36] Y. Liu, Z. Yi, and T. Chen, "Backdoor attacks and defenses in feature-partitioned collaborative learning," 2020, *arXiv:2007.03608*.

[37] Y. He et al., "Backdoor attack against split neural network-based vertical federated learning," *IEEE Trans. Inf. Forensics Security*, vol. 19, pp. 748–763, 2024.

[38] P. Chen, J. Yang, J. Lin, Z. Lu, Q. Duan, and H. Chai, "A practical clean-label backdoor attack with limited information in vertical federated learning," in *Proc. IEEE Int. Conf. Data Mining (ICDM)*, Dec. 2023, pp. 41–50.

[39] J. Yang, P. Chen, Z. Lu, R. Deng, Q. Duan, and J. Zeng, "Backdoor attack on vertical federated graph neural network learning," 2024, *arXiv:2410.11290*.

[40] M. Naseri, Y. Han, and E. De Cristofaro, "BadVFL: Backdoor attacks in vertical federated learning," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2024, pp. 2013–2028.

[41] Y. Liu et al., "Vertical federated learning: Concepts, advances, and challenges," *IEEE Trans. Knowl. Data Eng.*, vol. 36, no. 7, pp. 3615–3634, Jul. 2024.

[42] P. Chen, X. Du, Z. Lu, and H. Chai, "Universal adversarial backdoor attacks to fool vertical federated learning in cloud-edge collaboration," 2023, *arXiv:2304.11432*.

[43] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," 2013, *arXiv:1312.6114*.

[44] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 815–823.

[45] J. Zhai, S. Zhang, J. Chen, and Q. He, "Autoencoder and its various variants," in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, Oct. 2018, pp. 415–419.

[46] S. Kullback and R. A. Leibler, "On information and sufficiency," *Ann. Math. Statist.*, vol. 22, no. 1, pp. 79–86, 1951.

[47] H. Ishfaq, A. Hoogi, and D. Rubin, "TVAE: Triplet-based variational autoencoder using metric learning," 2018, *arXiv:1802.04403*.

[48] W. Li, K. Qi, W. Chen, and Y. Zhou, "Unified batch all triplet loss for visible-infrared person re-identification," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2021, pp. 1–8.

[49] T. Chen, S. Kornblith, M. Norouzi, and G. E. Hinton, "A simple framework for contrastive learning of visual representations," in *Proc. 37th Int. Conf. Mach. Learn.*, vol. 119, 2020, pp. 1597–1607.

[50] R. Bellman, "On a routing problem," *Quart. Appl. Math.*, vol. 16, no. 1, pp. 87–90, 1958.

[51] L. Bottou, F. E. Curtis, and J. Nocedal, "Optimization methods for large-scale machine learning," *SIAM Rev.*, vol. 60, no. 2, pp. 223–311, Jan. 2018.

[52] W. Fang, Z. Yu, Y. Jiang, Y. Shi, C. N. Jones, and Y. Zhou, "Communication-efficient stochastic zeroth-order optimization for federated learning," *IEEE Trans. Signal Process.*, vol. 70, pp. 5058–5073, 2022.

[53] M. Fiedler, "Algebraic connectivity of graphs," *Czechoslovak Math. J.*, vol. 23, no. 2, pp. 298–305, 1973.

[54] T. Chen, X. Jin, Y. Sun, and W. Yin, "VAFL: A method of vertical asynchronous federated learning," 2020, *arXiv:2007.06081*.

[55] L. Deng, "The MNIST database of handwritten digit images for machine learning research [Best of the Web]," *IEEE Signal Process. Mag.*, vol. 29, no. 6, pp. 141–142, Nov. 2012.

[56] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-MNIST: A novel image dataset for benchmarking machine learning algorithms," 2017, *arXiv:1708.07747*.

[57] A. Krizhevsky, "Learning multiple layers of features from tiny images," Univ. Toronto, Toronto, ON, Canada, Tech. Rep. 0, 2009.

[58] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, "Reading digits in natural images with unsupervised feature learning," in *Proc. NIPS Workshop Deep Learn. Unsupervised Feature Learn.*, 2024, pp. 1–4.

[59] M. Patacchiola and A. Storkey, "Self-supervised relational reasoning for representation learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 4003–4014.

[60] Z. Novack, S. Garg, J. McAuley, and Z. C. Lipton, "CHiLS: Zero-shot image classification with hierarchical label sets," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2023, pp. 26342–26362.

[61] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.

[62] A. Paszke et al., "PyTorch: An imperative style, high-performance deep learning library," in *Proc. Adv. neural Inf. Process. Syst.*, vol. 32, 2019, pp. 8026–8037.

[63] T. Liu et al., "Beyond traditional threats: A persistent backdoor attack on federated learning," in *Proc. AAAI Conf. Artif. Intell.*, 2024, vol. 38, no. 19, pp. 21359–21367.

[64] Y. Mu, "Deep leakage from gradients," in *Proc. Adv. Neural Inf. Process. Syst.*, 2022, pp. 14774–14784.

[65] A. Dosovitskiy et al., "An image is worth 16×16 words: Transformers for image recognition at scale," in *Proc. Int. Conf. Learn. Represent.*, 2021. [Online]. Available: https://openreview.net/forum?id=YicbFdNTTy&utm_campaign=f86497ed3a-EMAIL_CAMPAIGN_2019_04_24_03_18_COPY_01&utm_medium=email&utm_source=Deep%20Learning%20Weekly&utm_term=0_3845 67b42d-f86497ed3a-72965345

[66] Z. Tu, P. Milanfar, and H. Talebi, "MULLER: Multilayer Laplacian resizer for vision," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2023, pp. 6854–6864.

[67] Q. H. Nguyen and W. J. Beksi, "Single image super-resolution via a dual interactive implicit neural network," in *Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis. (WACV)*, Jan. 2023, pp. 4936–4945.

[68] Z. Pan et al., "Towards bidirectional arbitrary image rescaling: Joint optimization and cycle idempotence," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 17389–17398.

Seohyun Lee (Student Member, IEEE) received the B.S. degree in computer science from Purdue University with a concentration in machine intelligence in 2025 and is a graduated from Purdue University's John Martinson Honors College. He is currently pursuing the M.S. degree in electrical and computer engineering with Purdue University. His research interests include federated learning and fairness, security, and personalization in machine learning.

Wenzhi Fang (Graduate Student Member, IEEE) received the B.S. degree from Shanghai University in 2020 and the master's degree from ShanghaiTech University in 2023. He is currently pursuing the Ph.D. degree in electrical and computer engineering with Purdue University, West Lafayette, USA. His research interests include optimization theory and its applications in machine learning, signal processing, and wireless networks.

Anindya Bijoy Das (Member, IEEE) received the B.Sc. degree from the Department of Electrical and Electronic Engineering, Bangladesh University of Engineering and Technology, Dhaka, Bangladesh, in 2014, and the M.Eng. and Ph.D. degrees from the Department of Electrical and Computer Engineering, Iowa State University, Ames, IA, USA, in 2018 and 2022, respectively. He is currently an Assistant Professor with the Department of Electrical and Computer Engineering, The University of Akron, Akron, OH, USA. His research interests include the different aspects of federated learning, large language models, coding theory, information theory, and ML/AI applications. He was a recipient of the Karas Award from Iowa State University in 2022 for his outstanding dissertation in the area of mathematical and physical sciences and engineering. He also received the Research Excellence and Teaching Excellence Awards from the Department of Electrical and Computer Engineering, Iowa State University.

Seyyedali Hosseinalipour (Senior Member, IEEE) received the B.S. degree (Hons.) in electrical engineering from the Amirkabir University of Technology, Tehran, Iran, in 2015, and the M.S. and Ph.D. degrees in electrical engineering from North Carolina State University, NC, USA, in 2017 and 2020, respectively. He was a Post-Doctoral Researcher at Purdue University, IN, USA, from 2020 to 2022. His research interests include the analysis of modern wireless networks, synergies between machine learning methods and fog/edge computing systems, distributed/federated machine learning, and network optimization. He was a recipient of the 2020 ECE Doctoral Scholar of the Year Award and the 2021 ECE Distinguished Dissertation Award at NC State University, and the 2023 Students' Choice Teaching Excellence Award at the University at Buffalo–SUNY. Furthermore, he was the first author of a paper published in IEEE/ACM TRANSACTIONS ON NETWORKING that received the 2024 IEEE Communications Society William Bennett Prize.

David J. Love (Fellow, IEEE) received the B.S. (Hons.), M.S.E., and Ph.D. degrees in electrical engineering from The University of Texas at Austin in 2000, 2002, and 2004, respectively. Since 2004, he has been with the Elmore Family School of Electrical and Computer Engineering, Purdue University, where he is currently the Nick Trbovich Professor of Electrical and Computer Engineering. He holds 32 issued U.S. patents. His research interests include the design and analysis of broadband wireless communication systems, beyond-5G wireless systems, multiple-input multiple-output (MIMO) communications, millimeter wave wireless, software-defined radios and wireless networks, coding theory, and MIMO array processing. He was a member of the Executive Committee for the National Spectrum Consortium. He is a Fellow of American Association for the Advancement of Science (AAAS) and the National Academy of Inventors (NAI). He was named a Thomson Reuters Highly Cited Researcher (2014 and 2015). Along with his co-authors, he won best paper awards from the IEEE Communications Society (the 2016 Stephen O. Rice Prize, the 2020 Fred W. Ellersick Prize, and the 2024 William R. Bennett Prize), the IEEE Signal Processing Society (the 2015 IEEE Signal Processing Society Best Paper Award), and the IEEE Vehicular Technology Society (the 2010 Jack Neubauer Memorial Award). He serves as a Senior Editor for IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS. Previously, he was a Senior Editor of *IEEE Signal Processing Magazine*, an Editor of IEEE TRANSACTIONS ON COMMUNICATIONS, an Associate Editor of IEEE TRANSACTIONS ON SIGNAL PROCESSING, and a Guest Editor for special issues of IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS and *EURASIP Journal on Wireless Communications and Networking*.

Christopher G. Brinton (Senior Member, IEEE) received the M.S. and Ph.D. (Hons.) degrees in electrical engineering from Princeton University in 2016 and 2013, respectively. He is currently the Elmore Associate Professor of electrical and computer engineering (ECE) and the Faculty Director of the HIVE Engineering Entrepreneurship Center, Purdue University. Prior to joining Purdue University, he was the Associate Director of the EDGE Laboratory and a Lecturer of electrical engineering at Princeton University. He co-founded Zoomi Inc., a start-up company that has provided learning optimization to more than one million users worldwide. He holds the U.S. patents in machine learning for education, based on his Ph.D. research. His book *The Power of Networks: 6 Principles That Connect Our Lives* and associated Massive Open Online Courses (MOOCs) have reached over 4 00 000 students globally. His research interests include the intersection of networking, machine learning, and communications, specifically in edge network intelligence, heterogeneous distributed learning, and AI/ML-inspired wireless network optimization. He was a recipient of five of the top early-career awards given in the USA from the National Science Foundation (CAREER), the Office of Naval Research (YIP), the Defense Advanced Research Projects Agency (YFA and Director's Fellowship), and the Air Force Office of Scientific Research (YIP). He was a recipient of the IEEE Communication Society William Bennett Prize Best Paper Award, the Intel Rising Star Faculty Award, the Qualcomm Faculty Award, and the Purdue College of Engineering Faculty Excellence Award in Early Career Research, Early Career Teaching, and Online Learning. He serves as an Associate Editor for IEEE TRANSACTIONS ON NETWORKING. Previously, he was an Associate Editor of IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS.