RVFR: ROBUST VERTICAL FEDERATED LEARNING VIA FEATURE SUBSPACE RECOVERY

Anonymous authors
Paper under double-blind review

ABSTRACT

Vertical Federated Learning (VFL) is a distributed learning paradigm that allows multiple agents to jointly train a global model when each agent holds a different subset of features for the same sample(s). VFL is known to be vulnerable to backdoor attacks, where data from malicious agents are manipulated during training, and vulnerable to inference-phase attacks, where malicious agents manipulate the test data. However, unlike the standard horizontal federated learning, improving the robustness of VFL remains challenging, as there is no clear redundancy among the agents in VFL. To this end, we propose RVFR, a novel robust VFL training and inference framework. Under certain conditions, RVFR can recover the underlying uncorrupted features with provable guarantees and thus sanitizes the model against a vast range of backdoor attacks. Further, RVFR also defends against inference-phase adversarial and missing feature attacks. We conduct extensive experiments on NUS-WIDE and CIFAR-10 datasets and show that RVFR outperforms different baselines in terms of robustness against diverse types of attacks.

1 INTRODUCTION

Federated Learning (FL) involves distributed training where a central server coordinates with multiple agents to collaboratively train a machine learning model, and the agents keep their own training data, e.g., due to privacy concerns. The theory and practice of FL has progressed rapidly in recent years (Konečnỳ et al., 2016; McMahan et al., 2017; Bonawitz et al., 2017; Dayan et al., 2021). Two broad categories (Yang et al., 2019) of FL are Horizontal FL (HFL) and Vertical FL (VFL). In HFL, each agent has a different training set, while in VFL (Liu et al., 2019; Chen et al., 2020; Liu et al., 2020), different agents hold different parts of features for the same set of training data. For example, in VFL for credit score prediction, Agent 1 may have the banking data of a user and Agent 2 may have the shopping history of the same user, while the server holds corresponding labels.

Given the distributed training, FL raises new security concerns as the server has less control of the training data and agent devices. A variety of attacks have been demonstrated on HFL and on VFL. Several studies have focused on making HFL more robust, wherein each agent sends the model updates based on local data to the server and the server aggregates these updates. Robust aggregation protocols have been proposed as defenses in HFL against malicious attacks (Yin et al., 2018; Guerraoui et al., 2018; Blanchard et al., 2017; Fu et al., 2019; Pillutla et al., 2019; Fung et al., 2020; Chen et al., 2017; Xie et al., 2019b). However, it is challenging to defend against malicious attacks in VFL, as there is no clear redundancy among the agents. In fact, there are few studies on robustness exploration for VFL.

In this paper, we propose a robust VFL training and inference framework via feature subspace recovery (RVFR), which is able to defend against many types of attacks during both training and inference (see the attack taxonomy in Section 4.1). In particular, during **training**, we propose to train each agent's feature extractor separately based on feedback from the server (Quarantine training stage); the server then performs robust feature subspace recovery given the embedded features provided by different agents (Robust feature subspace recovery stage); and the server further purifies the features based on the assumption that the fraction of malicious agents is relatively small (Feature purifying stage). Finally, the server trains its global model based on the purified features (Server training stage). An overview of the training process is illustrated in Figure 1. During the **inference** phase, the server first purifies the embedded features provided by the agents and then feeds it to the trained global model for prediction. Building upon our framework, we aim to answer the following

questions: *Is it possible to train a clean model with theoretical guarantees in the presence of poisoned features? Can we make predictions based on incomplete (corrupted) features? If yes, how many malicious agents/instances and missing features can we tolerate?*

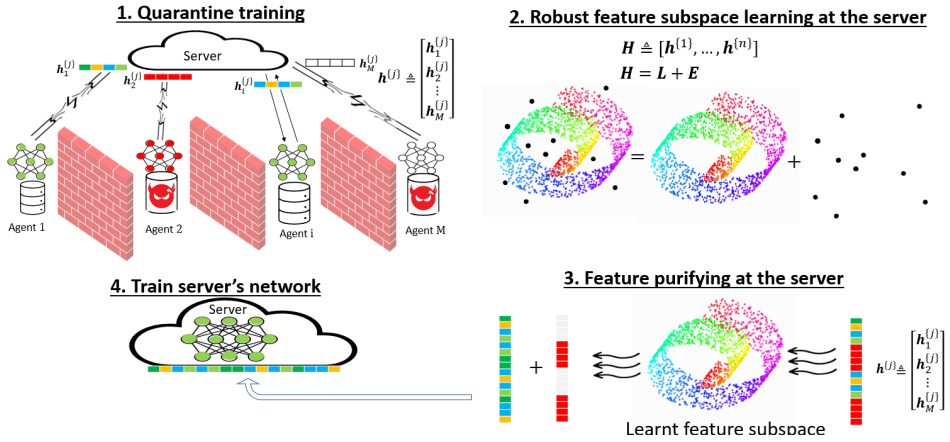

Figure 1: Overview of RVFR framework.

Under the RVFR framework, in the **robust feature subspace recovery stage**, we propose a variant of the Robust AutoEncoder (Zhou & Paffenroth, 2017) to recover the feature subspace. While the existing Robust AutoEncoder has no theoretical justification, we provide theoretical support for it by proving that when the underlying feature subspace is linear, it can exactly recover the linear feature subspace in the presence of corruption/poisoning. In the **feature purifying stage** (during both training and inference), we propose a novel AutoEncoder-based robust decomposition method that decomposes the potentially corrupted feature vectors into two parts: one lies on the learnt feature subspace (can be non-linear); and the other contains a block-sparse structure. We theoretically show that under certain conditions, the proposed method can recover the underlying feature vector *exactly* despite corruption from malicious agents.

Technical Contributions: We take a first step towards providing robust VFL against diverse adversarial attacks during both training and inference. Our main contributions include the following:

- We propose a novel robust training procedure to defend against backdoor attacks in VFL, and prove that under certain conditions it can exactly recover the underlying uncorrupted features. To the best of our knowledge, this is the first defense for VFL against training attacks with theoretical guarantees.
- We provide the first theoretical justification for the Robust AutoEncoder, which may be of independent interest.
- We propose a robust VFL inference procedure to defend against a variety of targeted or untargeted attacks (e.g., adversarial and missing feature attacks). To our knowledge, this is the first defense for VFL against inference-phase attacks with theoretical guarantees.
- We conduct extensive experiments on NUS-WIDE and CIFAR-10 datasets, and show that RVFR is significantly more robust than baselines against diverse types of attacks.

2 Related Work

In this section, we briefly review prior work on backdoor attacks and defenses in different types of federated learning frameworks.

Backdoor attack and defense in Horizontal Federated Learning. Many recent works, e.g., Bhagoji et al. (2019); Bagdasaryan et al. (2020); Wang et al. (2020); Xie et al. (2019a), have demonstrated the vulnerability of HFL to backdoor attacks. For instance, Bhagoji et al.; Bagdasaryan et al. show that training poisoned local models and submit malicious model updates to the server can mislead the global model effectively. While Xie et al. exploits the distributed nature of HFL and propose a distributed backdoor attack.

To defend against backdoor attacks in HFL, several robust federated protocols are proposed to mitigate the attacks during training empirically. For example, Sun et al. shows that clipping the norm of model

updates and adding Gaussian noise can mitigate backdoor attacks. Andreina et al. incorporates an additional validation phase to each round of FL to detect backdoor. To provide certain robustness guarantees, some robust aggregation methods are proposed (Yin et al., 2018; Guerraoui et al., 2018; Blanchard et al., 2017; Fu et al., 2019; Pillutla et al., 2019; Fung et al., 2020; Chen et al., 2017; Xie et al., 2019b); while such robustness guarantees are not for backdoor attacks directly. Recently, Cao et al. proposes Ensemble FL to defend against backdoor attacks with certifiable robustness. However, the proposed majority voting strategy requires training hundreds of FL models. A recent work exploits model clipping and smoothing in HFL, aiming to provide certified robustness to backdoor attacks (Xie et al., 2021). However, none of these works can provide robustness guarantees on the learned feature subspace of the server model, which is the focus of our work.

Backdoor attack and defense in Vertical Federated Learning. Backdoor attack against VFL is challenging since in the typical setting (Chen et al., 2020) the agent does not have the label information. Recent studies assume that the malicious agent knows at least one training instance belonging to the target class (Liu et al., 2020). Under this assumption, they propose the gradient-replacement method to perform backdoor attack. Some defense strategies are proposed against such attacks in VFL, e.g., sparsify the intermediate gradient before sending to the server (Liu et al., 2020). However, there is no work providing any robustness certification against backdoor attacks in VFL to our best knowledge.

3 PRELIMINARIES

Vertical Federated Learning. We first describe the basic setup of a typical VFL framework (Chen et al., 2020; Liu et al., 2020). There are M agents and a server collaboratively training a machine learning model based on a set of n training data $\mathcal{D} \triangleq \{\boldsymbol{x}_1^{\{j\}}, ..., \boldsymbol{x}_M^{\{j\}}, y^{\{j\}}\}_{j=1}^n$. The server (e.g., a third party) holds the label $y^{\{j\}}$, while agent i holds partial feature $\boldsymbol{x}_i^{\{j\}}$ of instance $\boldsymbol{x}^{\{j\}} \triangleq [\boldsymbol{x}_1^{\{j\}}; ...; \boldsymbol{x}_M^{\{j\}}]$. Due to privacy concerns, the raw agent data $\boldsymbol{x}_i^{\{j\}}$ are not shared with other agents and the server. Instead, each agent i learns a local embedding g_i parameterized by θ_i that maps the original feature vector $\boldsymbol{x}_i^{\{j\}}$ to feature vector $\boldsymbol{h}_i^{\{j\}} \triangleq g_i(\boldsymbol{x}_i^{\{j\}}; \theta_i)$. The dimension of $\boldsymbol{h}_i^{\{j\}}$ is usually smaller than $\boldsymbol{x}_i^{\{j\}}$. The server only has access to the embedded features from the agents. The learning objective of VFL is to minimize the following loss:

$$L(\mathcal{D}; \theta_0, \theta_1, ..., \theta_M) = \frac{1}{n} \sum_{j=1}^{n} \ell(\hat{y}^{\{j\}}, y^{\{j\}}) + \sum_{i=1}^{M} \gamma(\theta_i), \tag{1}$$

where $\hat{y}^{\{j\}} = f_{\theta_0}(\boldsymbol{h}_1^{\{j\}}, ..., \boldsymbol{h}_M^{\{j\}}) \triangleq f_{\theta_0}\left(g_1(\boldsymbol{x}_1^{\{j\}}; \theta_1), ..., g_M(\boldsymbol{x}_M^{\{j\}}; \theta_M)\right)$, and f_{θ_0} is the global model of server. ℓ is the loss function and γ is the regularizer that confines the model complexity.

Robust Subspace Recovery. Robust subspace recovery aims to recover the underlying subspace of data despite that some data points are arbitrarily corrupted (a.k.a. outliers). Mathematically, one observes a data matrix $\boldsymbol{H}$, where each column corresponds to a data point. We can decompose $\boldsymbol{H}$ as $\boldsymbol{L} + \boldsymbol{E}$, where $\boldsymbol{L}$ is the underlying uncorrupted data matrix, and the matrix $\boldsymbol{E}$ models the outlier corruptions. Since the fraction of the outliers is usually small, most columns of $\boldsymbol{E}$ are zero, *i.e.*, $\boldsymbol{E}$ is column-sparse.

Assume the underlying uncorrupted data points lie on a low-dimensional linear subspace, one of the notable Robust PCA works (Xu et al., 2012) showed that it is possible to recover the column-space of $\boldsymbol{L}$ exactly by solving the following convex optimization:

$$\min_{\boldsymbol{L}, \boldsymbol{E}} \|\boldsymbol{L}\|_* + \lambda \|\boldsymbol{E}^T\|_{2,1} \quad s.t.\ \boldsymbol{H} = \boldsymbol{L} + \boldsymbol{E}. \tag{2}$$

Later we will see that in our proposed RVFR framework, $\boldsymbol{H} = \left[\boldsymbol{h}^{\{1\}}, ..., \boldsymbol{h}^{\{n\}}\right]$, where the j-th column of $\boldsymbol{H}$ corresponds to the embedded features for the j-th instance. In the backdoor attack, a few columns of $\boldsymbol{H}$ are poisoned. In general the low-dimensional manifold that the embedded features $\boldsymbol{L}$ lies on is not linear, and we use a Robust AutoEncoder instead of Robust PCA to capture this non-linearity. We will show that Robust PCA can be viewed as a linearized version of the proposed Robust AutoEncoder, and we will provide theoretical justification for it.

Formally, the Robust PCA result from Xu et al. (2012) is as follows:

Theorem 1. (Exact subspace recovery (Xu et al., 2012)) Suppose we observe $\boldsymbol{H} = \boldsymbol{L}^* + \boldsymbol{E}^*$, where $\boldsymbol{L}^*$ has rank r and incoherence parameter μ (see Definition 1 in Appendix A.2). Suppose

further that $\boldsymbol{E}^*$ is supported on at most βn columns. Any output to Eq. 2 recovers the column space of $\boldsymbol{L}^*$ exactly, as long as the fraction of corrupted columns, β, satisfies $\frac{\beta}{1-\beta} \leq \frac{c_1}{\mu r}$, where $c_1 = 9/121$. This can be achieved by setting the parameter λ in Eq. 2 to be $\frac{3}{7\sqrt{\beta n}}$ (in fact it holds for any λ in a certain range).

Inspired by the Robust PCA properties, we propose the Robust AutoEncoder in this work, which can capture the more general **non-linear** manifold. So far there is no theoretical guarantees for Robust AutoEncoders. The proposed Robust AutoEncoder is slightly different from an existing one (Zhou & Paffenroth, 2017), and we theoretically justify the proposed one by showing that when the underlying feature subspace is linear, after transformation based on the proposed Robust AutoEncoder, it exactly recovers the linear feature subspace as Robust PCA. Note that our established connection between the proposed Robust AutoEncoder and Robust PCA is non-trivial, especially due to the proposed $\ell_{2,1}$-norm regularizer on the latent layer.

Note that the exact recovery of the feature subspace is not enough for our goal of robust VFL, and we need to recover the underlying $\boldsymbol{L}^*$. We further propose a robust decomposition technique to recover $\boldsymbol{L}^*$ exactly by utilizing our learnt feature subspace based on the assumption that the fraction of malicious agents is small. More details can be found in Theorem 3.

4 Robust VFL via Feature Subspace Recovery (RVFR)

We first describe the threat models during VFL training and inference, and then present the training and inference procedures of the proposed RVFR framework.

4.1 Threat Model

There are M agents which hold different parts of the feature of the same set of instances, and the label y is held by the server. The attacker knows the feature information held by every agent and the label information on the server, but it can only choose and control αM agents to perform malicious attacks. During training, the goal of the malicious agents is to perform backdoor attacks, *i.e.*, to achieve high accuracy on both the original main task and the targeted backdoor tasks.

Training phase threat model (backdoor attack): There are α fraction of agents that are malicious, and total β fraction of instances with backdoor trigger. With the adversarial target y^A, the backdoor attacks are conducted by optimizing the following objective for the instances with backdoor trigger:

$$\min_{\boldsymbol{h}_B} \ell(f_{\theta^B}(\{\boldsymbol{h}_B, \boldsymbol{h}_{benign}\}), y^A), \text{where } \theta^B = \arg\min_{\theta \in \Theta} \sum_j \ell(f_\theta(\boldsymbol{h}^{\{j\}}), y^{\{j\}}), \tag{3}$$

here $\boldsymbol{h}_B$ represents overall embedded features provided by the malicious agents for the instance with backdoor trigger, its magnitude can be arbitrarily large unless regularized by the server; while $\boldsymbol{h}_{benign}$ denotes the overall embedded features provided by benign agents. If j-th instance has backdoor trigger, $\boldsymbol{h}^{\{j\}}$ contains $\boldsymbol{h}_B$.

During inference, the malicious agents can perform adversarial attacks as well. Assume the total number of agents is M, and the fraction of malicious agents is α. We use $\boldsymbol{h}_{benign}$ to denote the embedded features provided by the $(1-\alpha)M$ benign agents (these agents are indexed by Ω_{benign} and $|\Omega_{benign}| = (1-\alpha)M$), and we use $\boldsymbol{h}_{adv}$ to denote the embedded features provided by malicious agents (indexed by Ω_{adv}). We categorize the inference phase attacks into the following three types:

Inference phase threat model A (adversarial attack): The malicious agents aim to send adversarial features $\boldsymbol{h}_{adv}$ to mislead the prediction, whose magnitude can be arbitrarily large unless regularized by the server:

$$\min_{\boldsymbol{h}_{adv}} \ell(f_{\theta_0}(\{\boldsymbol{h}_{adv}, \boldsymbol{h}_{benign}\}), y^A), \tag{4}$$

Inference phase threat model B (missing-feature attack): On the other hand, the malicious agents also have the option to NOT provide any features to the server. Such missing feature attack can make the malicious agents less detectable and still influence server's prediction. The presence of missing feature attack can be viewed as that the server only observes partial blocks indexed by Ω of the overall embedded features $\boldsymbol{h}$, *i.e.*, $\boldsymbol{h}_\Omega$. We use Ω^c to denote the index set of agents that perform missing feature attacks. The attacker can compromise arbitrary αM agents.

$$\min_{\Omega^c} \ell(f_{\theta_0}(\boldsymbol{h}_\Omega), y^A), s.t. \ |\Omega^c| \leq \alpha M \tag{5}$$

Inference phase threat model C (combined attack): The attackers can also perform a combined attack of the adversarial and missing feature attacks:

$$\min_{\Omega^c,\, \boldsymbol{h}_{adv}} \ell(f_{\theta_0}(\{\boldsymbol{h}_{adv}, \boldsymbol{h}_{benign}\}_\Omega), y^A), s.t.\ |\Omega^c| + |\Omega_{adv}| \leq \alpha M \tag{6}$$

Besides above mentioned targeted attacks, the attacker can also perform untargeted attacks by maximizing the distance between the prediction and the ground truth, which should be easier and here we focus on defending these strong targeted attacks.

4.2 Training Procedures of RVFR

The proposed training procedure has four stages: quarantine training, robust feature subspace recovery, feature purifying, and server training.

Stage 1 (**Quarantined training**): In VFL, the server trains the combined ML model, and each agent trains its local feature extractor. A strong adversary could interfere with the server training and further propagate errors to benign agents. So we propose that the server first connects to each agent separately to train the feature extractors g_i parameterized by $\theta_i, i = 1, ..., M$. The training can be viewed as a special case of (Chen et al., 2020, Algorithm 1) with only one agent.

In particular, agent i sends $\boldsymbol{h}_i^{\{j\}} \triangleq g_i(\boldsymbol{x}_i^{\{j\}}; \theta_i)$ to the server. The server uses it to update/train a temporary global model (whose input dimension equals that of $\boldsymbol{h}_i^{\{j\}}$) with learning rate η_0 and batch size B_0. Then the server calculates $\frac{\partial \ell}{\partial \boldsymbol{h}_i^{\{j\}}}$ and sends it back to agent i. The benign agent i updates its local feature extractor parameterized by θ_i using the combined gradient $\frac{\partial \ell}{\partial \boldsymbol{h}_i^{\{j\}}} \frac{\partial \boldsymbol{h}_i^{\{j\}}}{\partial \theta_i}$ with learning rate η_i and batch size B_i. After a number of iterations between the server and agent i, the server records the final embedded feature $\boldsymbol{h}_i^{\{j\}}$ for every instance $\boldsymbol{x}_i^{\{j\}}, j = 1, ..., n$. Finally, for each instance $\boldsymbol{x}^{\{j\}}$, the server holds $\{\boldsymbol{h}_1^{\{j\}}, \boldsymbol{h}_2^{\{j\}}, ..., \boldsymbol{h}_M^{\{j\}}\}$ from all the M agents, and concatenates them into a long column vector $\boldsymbol{h}^{\{j\}}$. Let $\boldsymbol{H} = \left[\boldsymbol{h}^{\{1\}}, ..., \boldsymbol{h}^{\{n\}}\right]$, the j-th column of $\boldsymbol{H}$ corresponds to the concatenated embeded features for the j-th instance.

Stage 2 (**Robust feature subspace recovery**): Since the fraction β of the backdoored training instances is small, this provides us the redundancy across instances for robustifying the server's model. In this stage, the server disconnects from all the agents, and uses $\boldsymbol{H}$ to train a Robust AutoEncoder by minimizing the following objective:

$$\min_{\boldsymbol{L},\boldsymbol{E},\phi,\psi} \|\mathcal{E}_\psi(\boldsymbol{L})\|_{2,1} + \lambda \|\boldsymbol{E}^T\|_{2,1} \quad s.t.\ \boldsymbol{H} = \boldsymbol{L} + \boldsymbol{E}, \quad \boldsymbol{L} = \mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{L})) \tag{7}$$

where $\boldsymbol{L}$ models the underlying feature subspace, and $\boldsymbol{E}$ models the outlier corruptions due to backdoor attacks. In Eq. 7, we only enforce $\boldsymbol{E}$ to be column-sparse. $\mathcal{D}_\phi$ is the decoder parameterized by ϕ, and $\mathcal{E}_\psi$ is the encoder parameterized by ψ.

Note that Eq. 7 is equivalent to the following:

$$\min_{\boldsymbol{L},\phi,\psi} \|\mathcal{E}_\psi(\boldsymbol{L})\|_{2,1} + \lambda \|(\boldsymbol{H} - \boldsymbol{L})^T\|_{2,1} \quad s.t.\ \boldsymbol{L} = \mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{L})) \tag{8}$$

In practice, we further relax the constraint $\boldsymbol{L} = \mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{L}))$ and solve the following instead:

$$\min_{\boldsymbol{L},\phi,\psi} \|\mathcal{E}_\psi(\boldsymbol{L})\|_{2,1} + \lambda \|(\boldsymbol{H} - \boldsymbol{L})^T\|_{2,1} + \eta \|\boldsymbol{L} - \mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{L}))\|_F^2 \tag{9}$$

To optimize this, one can perform alternating optimization for $\boldsymbol{L}$ and parameters of ϕ and ψ.

Stage 3 (**Feature purifying based on the recovered feature subspace**): Recovering the feature subspace is not enough to achieve robust VFL, as we need to recover the original features to train the server. Fortunately, there is another important prior information that we can leverage: the fraction α of the malicious agents is small. For each training instance $\boldsymbol{h}^{\{j\}}$, we use the learned AutoEncoder to decompose it as $\boldsymbol{h}^{\{j\}} = \mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}^{\{j\}})) + \boldsymbol{e}^{\{j\}}$, where $\boldsymbol{e}^{\{j\}}$ models the block-sparse corruptions (each block corresponds to an agent). Ideally we want to solve the following:

$$\min_{\boldsymbol{l},\boldsymbol{e}} \sum_{i=1}^{M} \mathbb{1}\{\boldsymbol{e}_i \neq \boldsymbol{0}\} \quad s.t. \quad \boldsymbol{h} = \mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l})) + \boldsymbol{e} \tag{10}$$

which is equivalent to:

$$\min_{\boldsymbol{l}} \sum_{i=1}^{M} \mathbb{1}\{[\boldsymbol{h} - \mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}))]_i \neq \boldsymbol{0}\} \tag{11}$$

However, the above block-sparsity minimization objective is NP-Complete, so we relax it to the $\ell_{2,1}$-norm:

$$\min_{\boldsymbol{l}} \sum_{i=1}^{M} \|[\boldsymbol{h} - \mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}))]_i\|_2 \tag{12}$$

Our theoretical analysis in next section shows that under certain conditions, solving the above relaxed objective is able to recover the underlying true features.

Stage 4 (**Server training**): The server trains its own global model parameterized by θ_0 based on the above purified features $\mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}))$, with learning rate η_0 and batch size B_0.

4.3 Inference Procedures of RVFR

We propose an all-in-one solution to defend against both training phase and inference phase malicious attacks defined in Section 4.1, and thus during inference we will also perform feature purifying using the recovered feature subspace and the sparse nature of the malicious agents.

Let Ω be the index set of the observed agent blocks. Assume we have successfully trained the AutoEncoder, such that we have $\boldsymbol{l} = \mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}))$, where $\boldsymbol{l}$ is any uncorrupted embedded feature vector. We want to use the AutoEncoder to decompose the observed $\boldsymbol{h}_\Omega$ as $\boldsymbol{h}_\Omega = \boldsymbol{l}_\Omega + \boldsymbol{e}_\Omega$ with $\boldsymbol{l} = \mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}))$, where $\boldsymbol{e}_\Omega$ models the block-sparse corruptions (each block corresponds to an agent) on the observed set Ω. Note that if all the blocks (agents) are observed, Ω becomes the whole set. Since $\boldsymbol{e}_\Omega$ is block-sparse, similar to the feature purifying during training, we naturally aim to solve the following:

$$\min_{\boldsymbol{l}} \sum_{i\in\Omega} \mathbb{1}\{[\boldsymbol{h} - \mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}))]_i \neq \boldsymbol{0}\} \tag{13}$$

However, the above block-sparsity minimization objective is NP-Complete, we instead relax it to the $\ell_{2,1}$-norm:

$$\min_{\boldsymbol{l}} \sum_{i\in\Omega} \|[\boldsymbol{h} - \mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}))]_i\|_2 \tag{14}$$

Our theoretical analysis in Theorems 3 & 4 show that under certain conditions, solving the above relaxed objective is able to recover the underlying true features, despite the partial observations. After obtained $\hat{\boldsymbol{l}}$, we feed $\mathcal{D}_\phi(\mathcal{E}_\psi(\hat{\boldsymbol{l}}))$ to server's trained machine learning model for the prediction.

5 Theoretical Analysis of RVFR

We first analyze the proposed Robust AutoEncoder, which is the backbone of the proposed defense framework. Then, we prove that the proposed feature purifying method can recover the underlying features exactly under certain conditions, which means that the server's global model can be both *trained and tested on correct features*, and thus is robust against diverse types of attacks.

The Robust AutoEncoder can be viewed as the generalization of Robust PCA, where the low-dimensional linear subspace is extended to the more general low-dimensional manifold. It is challenging to prove the exact recovery of the low-dimensional manifold in the presence of outlier corruptions. However, we next show that when the underlying feature subspace is linear, the Robust AutoEncoder can exactly recover the underlying subspace.

Theorem 2. (Exact subspace recovery) Assume $\boldsymbol{l}^{*\{i\}} \in \mathbb{R}^d, i = 1, ..., n$ lie on a low-dimension subspace, *i.e.*, $\text{rank}(\boldsymbol{L}^*) = r \ll d$. For the AutoEncoder, assume linear activation functions and no bias terms, while setting the dimension of the latent layer of AutoEncoder as $r = \text{rank}(\boldsymbol{L}^*)$ and restricting the weight matrices to be orthonormal. Then under the conditions in Theorem 1, the global optimal solution of Eq. 7 is equivalent to the solution of Eq. 2, and the corresponding weight matrix of the encoder is guaranteed to recover the subspace of $\boldsymbol{L}^*$ exactly.

The proof is omitted to Appendix A.3. The key is to show that the objective value of Eq. 7 is greater than or equal to the objective value of Eq. 2, with equality achieved when the weight matrix of the encoder equals the top-r left singular vectors of $\boldsymbol{L}$. This is non-trivial especially due to the proposed $\ell_{2,1}$-norm regularizer on the latent layer of the Robust AutoEncoder.

Remark 1. The conditions required in Theorem 2 provide some useful insights. First, the total fraction β of poisoned instances needs to be small. Second, the dimension of the underlying feature subspace needs to be small. If the dimension of the feature subspace is too large, it is difficult to distinguish corrupted features from the uncorrupted ones. On the other hand, low dimensionality implies that there is enough redundancy among the features among agents, which is preferred since redundancy provides more robustness. Third, as discussed in Appendix A.2 regarding the incoherence condition, preferring small incoherence parameters implies that we do not want any dimension of the feature subspace to be defined by very few data points. Otherwise, it is risky if those instances are poisoned, as it becomes impossible to recover that dimension.

Note that exact recovery of the subspace does not mean exact recovery of $\boldsymbol{L}$. For the corrupted/poisoned instances, usually it is impossible to restore them as the corruptions can be arbitrary. Fortunately, once we have recovered the underlying feature subspace, it is possible to recover the original features exactly by utilizing the sparseness of the malicious agents. This holds for both training and inference phases. The following theorem shows that we can recover the underlying embedded features exactly against the inference phase threat models A & C.

Theorem 3. (Exact feature recovery under threat model A & C) Let $\boldsymbol{W}_{end}$ be the weight matrix of the last layer of the AutoEncoder. Assume there is no bias term nor non-linear activation function in the *last layer*. Assume the trained AutoEncoder captures the underlying feature subspace (*i.e.*, $\mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l})) = \boldsymbol{l}$ for uncorrupted feature vector $\boldsymbol{l}$). Let Ω be the index set of the observed blocks (agents). If $\forall \boldsymbol{v} \in Range(\boldsymbol{W}_{end})\backslash \boldsymbol{0}$, for any partition $\{S_\Omega, \bar{S}_\Omega\}$ of Ω with $|S_\Omega| = g > |\Omega|/2$, it holds that $\sum_{i\in S_\Omega} \|\boldsymbol{v}_i\|_2 > \sum_{i\in \bar{S}_\Omega} \|\boldsymbol{v}_i\|_2$, then for any $\boldsymbol{h}_\Omega = \boldsymbol{l}^*_\Omega + \boldsymbol{e}^*_\Omega$ with $\mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}^*)) = \boldsymbol{l}^*$ and $\sum_{i\in\Omega} \mathbb{1}\{\boldsymbol{e}^*_i = \boldsymbol{0}\} \geq g$, $\mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}^*))$ is the unique solution of Eq. 14 and Eq. 13.

The proof can be found in Appendix A.4, where we use contradiction to show that there does not exist any global optimal solution of Eq. 14 or Eq. 13 that is different from $\mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}^*))$.

Remark 2. (Exact feature recovery via feature purifying during training) Theorem 3 directly justifies the proposed feature purifying during training. Under certain conditions, we can recover each column of $\boldsymbol{L}$ exactly during training.

Next, we analyze the exact feature recovery property under inference phase threat model B (missing feature attack), where there are no adversarial features.

Theorem 4. (Exact feature recovery under threat model B) Let $\boldsymbol{W}_{end}$ be the weight matrix of the last layer of the AutoEncoder. Assume there is no bias term nor non-linear activation function in the *last layer*. Assume the trained AutoEncoder captures the underlying feature subspace (*i.e.*, $\mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l})) = \boldsymbol{l}$ for uncorrupted feature vector $\boldsymbol{l}$). Let Ω be the index set of the observed blocks (agents). If $\forall \boldsymbol{v} \in Range(\boldsymbol{W}_{end})\backslash \boldsymbol{0}$, it holds that $\boldsymbol{v}_\Omega \neq 0$, then given $\boldsymbol{h}_\Omega = \boldsymbol{l}^*_\Omega$, $\mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}^*))$ is the unique solution of Eq. 14 and Eq. 13.

The proof is deferred to Appendix A.5, where we use contradiction to show that there does not exist any global optimal solution of Eq. 14 or Eq. 13 that is different from $\mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}^*))$.

The condition $\boldsymbol{v}_\Omega \neq \boldsymbol{0}$ for any $\boldsymbol{v} \in Range(\boldsymbol{W}_{end})\backslash \boldsymbol{0}$ required by Theorem 4 is much weaker than that required by Theorem 3, which is reasonable as there are no corruptions on the observed blocks in Theorem 4. This required condition also provides very useful insights on how many agents we need to observe in order to make a prediction. It can be understood through the following toy example. Suppose the prediction task is credit approval. Agent 1 holds the banking data, Agent 2 holds the shopping history, while Agent 3 holds age and gender features. Intuitively, if we only observe the features from Agent 3, it is very hard to make the prediction, since there are definitely Person A and B of the same age and gender, but with quite different other features. Let $\boldsymbol{l}^A$ be Person A's overall features, $\boldsymbol{l}^B$ be Person B's overall features. In this toy example, Ω corresponds to Agent 3 only. We have $\boldsymbol{l}^A \neq \boldsymbol{l}^B$ but $\boldsymbol{l}^A_\Omega = \boldsymbol{l}^B_\Omega$, or say $(\boldsymbol{l}^A - \boldsymbol{l}^B)_\Omega = \boldsymbol{0}$. Note that both $\boldsymbol{l}^A$ and $\boldsymbol{l}^B$ are in the range space of $\boldsymbol{W}_{end}$, so $(\boldsymbol{l}^A - \boldsymbol{l}^B) \in Range(\boldsymbol{W}_{end})\backslash \boldsymbol{0}$. The smaller the set Ω implies the higher chance that $(\boldsymbol{l}^A - \boldsymbol{l}^B)_\Omega = \boldsymbol{0}$. In general, there is no exact value for the minimal fraction of agents we need to observe in order to make a prediction. It highly depends on whether the observed agents cover enough information to distinguish different classes in the classification task. For example, if there is enough redundancy across the agents, say Agent 3 holds age, gender, banking data, and shopping history, then observing Agent 3 could be sufficient.

In summary, Theorem 2 shows that under certain natural conditions (e.g., the fraction of poisoning instances β is small), we can recover the underlying feature subspace exactly. Further, Theorems 3 & 4 show that under certain natural conditions (e.g., the fraction of malicious agents α is small and the AutoEncoder we have learnt captures the underlying feature subspace), we can recover the underlying embedded features exactly during both training and inference.

6 Empirical Evaluation

We first describe experimental setup in Section 6.1, and present experimental results in Section 6.2. Additional implementation details and results can be found in Appendix A.1.

6.1 Experimental Setup

Dataset and Models. We study the classification task on two datasets: NUS-WIDE and CIFAR-10. Following (Liu et al., 2020), which proposed the backdoor attack against VFL, we first use NUS-WIDE dataset to evaluate our defense. In NUS-WIDE, each sample has 634 image features, 1000 text features, and 5 different labels. We consider two VFL settings (1 server and M agents): $M = 2$ and $M = 4$. We denote G1 as Agent 1, G2 as Agent 2, etc. for simplicity. Following (Liu et al., 2020), when $M = 2$, G1 holds image features and G2 holds text features. When $M = 4$, G1 (G2) holds 225 (409) image features while G3 (G4) holds 500 (500) text features. In all settings, only the server holds the labels. In CIFAR-10 dataset, each sample have 32x32x3 image features. We consider the VFL settings for CIFAR-10 with $M = 2$ and $M = 3$. When $M = 2$, each agent has 16x32x3 features, i.e., half image. When $M = 3$, G1, G2, G3 have 9x32x3, 10x32x3, 13x32x3 features respectively. Following the standard setup, we use SGD to update the local models and server model for E epochs with learning rate 0.01 and batch size B_s. The detailed dataset and model descriptions as well as training parameter setups are summarized in Table 1 in Appendix.

Attack Setup. We consider the backdoor attack, its combination with the missing feature attack, as well as the combination of adversarial evasion attack (Goodfellow et al., 2014) and missing feature attack: **(1)** The backdoor attack is proposed by (Liu et al., 2020) and the detailed description of the attack method can be found in Appendix. A.1. In NUS-WIDE, the attacker's backdoor attack goal is to change the predicted label of the instance with a backdoor trigger to be a specific target class (*i.e.*, 'buildings'), where the backdoor trigger is 'the last text feature equals 1'. There are 152 backdoored training samples. In CIFAR-10, the backdoor targeted label is 'truck' and the trigger pattern is a white rectangle with size 4x6x3 in the lower right corner. The number of backdoored training samples is 100 when $M = 3$, and 600 when $M = 2$. **(2)** In the missing feature attack, the malicious agent does not send its local embedded feature to the server during inference phase. The server uses a zero vector as its local embedded feature. We combine this attack with other attacks as this attack alone is not powerful enough. **(3)** In the evasion attack, the attacker use the information from server to locally generate the partial adversarial input features (i.e., adversarial examples) based on the Fast Gradient Signed Method (FGSM) attack method (Goodfellow et al., 2014). The detailed setup of this attack method against VFL is presented in Appendix. A.1. The evasion attack (or its combination with missing feature attack) is only conducted during the VFL inference phase (VFL training procedure is attack-free).

Baseline methods. Besides the Vanilla VFL framework without defense (Liu et al., 2020, Algorithm 1), we compare our proposed defense method with two state-of-the-art VFL defense techniques, e.g., *Gradient Sparsification* method that sparsifies the intermediate gradients sent from the server to agents, and *Differential Privacy (DP)* that adds noise to such exchanged gradients (Liu et al., 2020). *Laplacian Noise* is used as a realization of the DP method. The noise variance is 0.05 for NUS-WIDE and 0.0001 for CIFAR-10 to preserve reasonable clean accuracy of the model. In Gradient Sparsification, the hyperparameter controlling the sparsity is 0.999 for NUS-WIDE and 0.95 for CIFAR-10. As for our proposed RVFR, in Stage 1 (i.e., quarantine training for local models) and Stage 4 (i.e., training for server model), we use the same training parameters as in Table 1. The robust autoencoder is a four-layer fully connected neural network. The details about the training of robust autoencoder, the alternative optimization, and the purified training of the server in Stage 2, 3, 4 and details of the inference procedures can be founded in Appendix. A.1. All of our experiments are run three times.

6.2 Empirical Results

Evaluation Metrics. For the backdoor attack (**BKD**) and its combination with missing feature attack (**Miss**), the evaluation metrics for defense methods are the Backdoor Accuracy (the lower the better)

on the backdoored test data and the main task Clean Accuracy (the higher the better) on the clean test dataset. There are 102 backdoored testing samples in NUS-WIDE and 10000 in CIFAR-10. For the adversarial evasion attack (**Adv**) and missing feature attack, the evaluation metric is the main task Accuracy on the adversarial examples. 10000 adversarial examples are generated for both datasets.

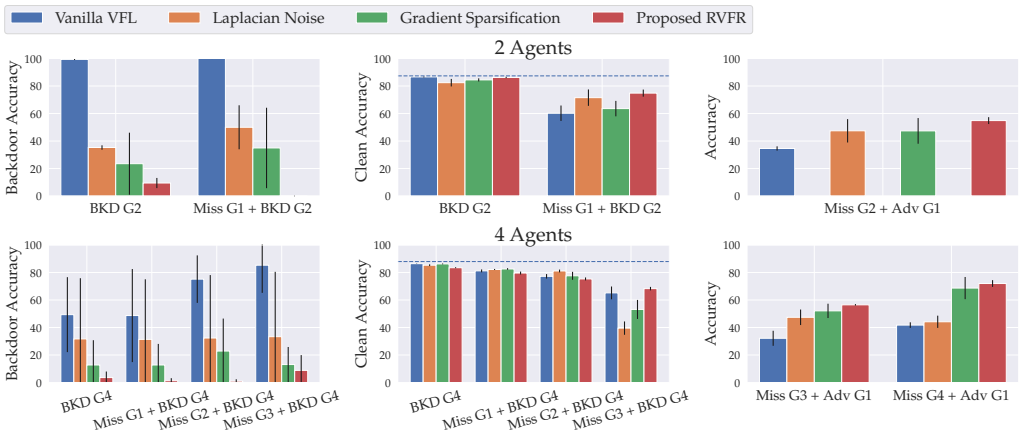

Figure 2: Performance comparison on NUS-WIDE under $M = 2$ (first row) and $M = 4$ (second row). The missing bars are due to zero values. Dashed blue line is Vanillia VFL without adversaries.

Figure 2 & 3 show the performance of the proposed RVFR and baseline methods on NUS-WIDE and CIFAR-10. When defending against backdoor attack and its combination with missing feature attack, the proposed RVFR method exhibits significantly lower Backdoor Accuracy and similar Clean Accuracy on both datasets, compared to other methods. Note that under the backdoor attack, in some cases, the Clean Accuracy of all defense methods is lower than the vanilla VFL, which exhibits a trade-off between robustness and accuracy. As shown in the right column of Figure 2 & 3, when defending against the combination of adversarial evasion attack and missing feature attack, the proposed RVFR method achieves consistently higher accuracy than the baseline methods on both datasets, which demonstrates its robustness against such attacks.

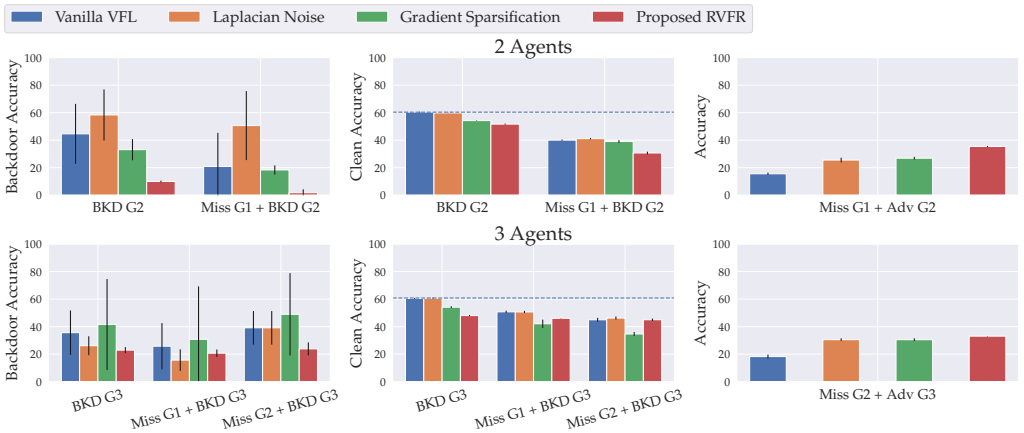

Figure 3: Performance comparison on CIFAR-10 under $M = 2$ (first row) and $M = 3$ (second row). Dashed blue line is Vanillia VFL without adversaries.

7 Conclusions

In this work, we proposed a novel robust feature subspace recovery based VFL framework to defend against backdoor attacks during training, and a variety of attacks during inference, both with theoretical guarantees. An important byproduct of our analysis is the first theoretical justification for the Robust AutoEncoder, which may be of independent interest. We further validate the robustness of our proposed framework through extensive experiments on NUS-WIDE and CIFAR-10 datasets.

ETHICS STATEMENT

We propose a novel defense algorithm for vertical federated learning, and provide analysis from theoretical and empirical perspectives. All the datasets and packages we use are open-sourced. We do not have ethical concerns in our paper.

REPRODUCIBILITY STATEMENT

The source code of the proposed RVFR method is available as the supplemental material, which also includes the detailed data processing steps. The complete proof of the theorems can be found in the Appendix.

REFERENCES

Sebastien Andreina, Giorgia Azzurra Marson, Helen Möllering, and Ghassan Karame. Baffle: Backdoor detection via feedback-based federated learning. *arXiv preprint arXiv:2011.02167*, 2020.

Eugene Bagdasaryan, Andreas Veit, Yiqing Hua, Deborah Estrin, and Vitaly Shmatikov. How to backdoor federated learning. In *International Conference on Artificial Intelligence and Statistics*, pp. 2938–2948. PMLR, 2020.

Arjun Nitin Bhagoji, Supriyo Chakraborty, Prateek Mittal, and Seraphin Calo. Analyzing federated learning through an adversarial lens. In *International Conference on Machine Learning*, pp. 634–643. PMLR, 2019.

Peva Blanchard, El Mahdi El Mhamdi, Rachid Guerraoui, and Julien Stainer. Machine learning with adversaries: Byzantine tolerant gradient descent. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pp. 118–128, 2017.

Keith Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. Practical secure aggregation for privacy-preserving machine learning. In *proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1175–1191, 2017.

Xiaoyu Cao, Jinyuan Jia, and Neil Zhenqiang Gong. Provably secure federated learning against malicious clients. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 6885–6893, 2021.

Tianyi Chen, Xiao Jin, Yuejiao Sun, and Wotao Yin. Vafl: a method of vertical asynchronous federated learning. *arXiv preprint arXiv:2007.06081*, 2020.

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

Ittai Dayan, Holger R Roth, Aoxiao Zhong, Ahmed Harouni, Amilcare Gentili, Anas Z Abidin, Andrew Liu, Anthony Beardsworth Costa, Bradford J Wood, Chien-Sung Tsai, et al. Federated learning for predicting clinical outcomes in patients with covid-19. *Nature medicine*, 27(10): 1735–1743, 2021.

Shuhao Fu, Chulin Xie, Bo Li, and Qifeng Chen. Attack-resistant federated learning with residual-based reweighting. *arXiv preprint arXiv:1912.11464*, 2019.

Clement Fung, Chris JM Yoon, and Ivan Beschastnikh. The limitations of federated learning in sybil settings. In *23rd International Symposium on Research in Attacks, Intrusions and Defenses ({RAID} 2020)*, pp. 301–316, 2020.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

Rachid Guerraoui, Sébastien Rouault, et al. The hidden vulnerability of distributed learning in byzantium. In *International Conference on Machine Learning*, pp. 3521–3530. PMLR, 2018.

Jakub Konečnỳ, H Brendan McMahan, Daniel Ramage, and Peter Richtárik. Federated optimization: Distributed machine learning for on-device intelligence. *arXiv preprint arXiv:1610.02527*, 2016.

Yang Liu, Yan Kang, Xinwei Zhang, Liping Li, Yong Cheng, Tianjian Chen, Mingyi Hong, and Qiang Yang. A communication efficient collaborative learning framework for distributed features. *arXiv preprint arXiv:1912.11187*, 2019.

Yang Liu, Zhiqian Yi, and Tianjian Chen. Backdoor attacks and defenses in feature-partitioned collaborative learning. *ArXiv*, abs/2007.03608, 2020.

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

Krishna Pillutla, Sham M Kakade, and Zaid Harchaoui. Robust aggregation for federated learning. *arXiv preprint arXiv:1912.13445*, 2019.

Ziteng Sun, Peter Kairouz, Ananda Theertha Suresh, and H Brendan McMahan. Can you really backdoor federated learning? *arXiv preprint arXiv:1911.07963*, 2019.

Hongyi Wang, Kartik Sreenivasan, Shashank Rajput, Harit Vishwakarma, Saurabh Agarwal, Jy-yong Sohn, Kangwook Lee, and Dimitris Papailiopoulos. Attack of the tails: Yes, you really can backdoor federated learning. *arXiv preprint arXiv:2007.05084*, 2020.

Chulin Xie, Keli Huang, Pin-Yu Chen, and Bo Li. Dba: Distributed backdoor attacks against federated learning. In *International Conference on Learning Representations*, 2019a.

Chulin Xie, Minghao Chen, Pin-Yu Chen, and Bo Li. Crfl: Certifiably robust federated learning against backdoor attacks. *arXiv preprint arXiv:2106.08283*, 2021.

Cong Xie, Sanmi Koyejo, and Indranil Gupta. Zeno: Distributed stochastic gradient descent with suspicion-based fault-tolerance. In *International Conference on Machine Learning*, pp. 6893–6901. PMLR, 2019b.

Huan Xu, Constantine Caramanis, and Sujay Sanghavi. Robust pca via outlier pursuit. *IEEE Transactions on Information Theory*, 58(5):3047–3064, 2012. doi: 10.1109/TIT.2011.2173156.

Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 10(2):1–19, 2019.

Dong Yin, Yudong Chen, Ramchandran Kannan, and Peter Bartlett. Byzantine-robust distributed learning: Towards optimal statistical rates. In *International Conference on Machine Learning*, pp. 5650–5659. PMLR, 2018.

Chong Zhou and Randy C Paffenroth. Anomaly detection with robust deep autoencoders. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 665–674, 2017.

A Appendix

A.1 Additional Experimental Details

Simulation Environment. We simulate the VFL setup (1 server and multiple agents) on a Linux machine with Intel® Xeon® Gold 6132 CPUs and 8 NVidia® 1080Ti GPUs. Our code is based on Tensorflow 2.0.

Table 1: Dataset description and parameters

Dataset	# training samples	# test samples	Batch size B_s	Epoch E	Local model	Embedded feature dim.	Server model
NUS-WIDE	60000	40000	64	200	1 fc	32	2 fc
CIFAR-10	50000	10000	128	100	1 conv and 2 fc	10	2 fc

VFL Backdoor Attacks. For the j-th instance with backdoor trigger, the malicious agent i does not update its local feature extractor (parameterized by θ_i) like benign agents via using the combined gradient $\frac{\partial \ell}{\partial \boldsymbol{h}_i^{\{j\}}} \frac{\partial \boldsymbol{h}_i^{\{j\}}}{\partial \theta_i}$. Instead, it replaces the intermediate gradient received from the server $\frac{\partial \ell}{\partial \boldsymbol{h}_i^{\{j\}}}$ by $\gamma \frac{\partial \ell}{\partial \boldsymbol{h}_i^{\{target\}}}$ and uses the manipulated gradient $\gamma \frac{\partial \ell}{\partial \boldsymbol{h}_i^{\{target\}}} \frac{\partial \boldsymbol{h}_i^{\{j\}}}{\partial \theta_i}$ to update its local feature extractor, where $\frac{\partial \ell}{\partial \boldsymbol{h}_i^{\{target\}}}$ is the intermediate gradient received from the server for any instance belonging to the target class, and γ is the gradient amplify ratio. In our experiments, we use $\gamma = 10$ for NUS-WIDE and $\gamma = 1$ for CIFAR-10. Moreover, since the label owned by the server is clean, the attacker would submit zero vector to the server to prevent the server mapping the poisoned embedding to the true label.

Evasion Attacks. Given the j-th test sample, the attacker adds small perturbations $\boldsymbol{x}_{\boldsymbol{test}i}^{\boldsymbol{per}\,\{j\}}$ generated from FGSM to the original data $\boldsymbol{x}_{\boldsymbol{test}i}^{\{j\}}$ to obtain the adversarial example $\boldsymbol{x}_{\boldsymbol{test}i}^{\boldsymbol{adv}\,\{j\}} = \boldsymbol{x}_{\boldsymbol{test}i}^{\{j\}} + \epsilon \boldsymbol{x}_{\boldsymbol{test}i}^{\boldsymbol{per}\,\{j\}}$ where ϵ controlling the degree of attack. For NUS-WIDE, we set $\epsilon = 0.01$ when $M = 2$ and $\epsilon = 0.1$ when $M = 4$; for CIFAR-10, we set $\epsilon = 0.01$.

Training and Inference Details of RVFR. In Stage 2, we first train the AutoEncoder for E_{rae} epochs, and alternatively update AutoEncoder and the estimated uncorrupted embedded features $\boldsymbol{L}$ for E_{alt} epochs using SGD with an initial learning rate 0.01 which exponentially decays by a factor of 0.99 every 10000 steps. During the alternating optimization, λ and η in the objective function are set to be 0.1. In NUS-WIDE, $E_{rae} = 100$ when $M = 2$ and $E_{rae} = 200$ when $M = 4$; in CIFAR-10, $E_{rae} = 100$. We use $E_{alt} = 100$ for all settings. In Stage 3 & 4, for each batch of training data, we first purifies the embedded features according to Eq. 12 (using the estimated $\boldsymbol{L}$ from Stage 2 as initialization) for $E_{pur\_train} = 2$ epochs, then use the purified embedded feature to train server's global ML model. We perform such procedure for $E_{server} = 100$ epochs using the same SGD optimizer as in Stage 2. In the inference phase, the server purifies the embedded features according to Eq. 14 (use the embedded features as initialization) for $E_{pur\_test} = 2$ epochs, and feeds the purified feature to the trained server model for prediction.

Clean Accuracy under Non-adversarial Setting In this section, we report the clean accuracy of different VFL methods under non-adversarial setting in Table 2.

Table 2: Clean accuracy of different VFL methods under non-adversarial setting

Setting	Vanilla VFL	Laplacian	Gradient Sparsification	Proposed RVFR
NUS 2 clients	87.3683 ± 0.0664	86.0283 ± 0.8465	86.5033 ± 0.2166	82.1408 ± 1.6045
NUS 4 clients	87.9542 ±0.0510	86.5583 ± 0.1008	86.3067 ± 0.2938	81.1667 ± 1.7558
CIFAR 2 clients	60.3633 ± 0.5413	60.1733 ± 0.4606	54.0400 ± 0.7758	59.4400 ± 0.0990

A.2 Definition of Incoherence parameter

Definition 1. (Incoherence parameter (Xu et al., 2012)) A matrix $\boldsymbol{L} \in \mathbb{R}^{d \times n}$ with thin SVD $\boldsymbol{L} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$ and $(1-\beta)n$ whose columns are non-zero, is said to be column-incoherent with parameter μ if

$$\max_i \|V^T \boldsymbol{e}_i\|_2^2 \leq \frac{\mu r}{(1-\beta)n}, \tag{15}$$

where $\{\boldsymbol{e}_i\}$ are the coordinate unit vectors, r is the rank of matrix $\boldsymbol{L}$.

A small incoherence parameter implies the right singular vectors of $\boldsymbol{L}$ are not 'spiky'. As mentioned in Xu et al. (2012), if the left hand side of Eq. 15 is as big as 1, it essentially means that one of the directions of the column space which we wish to recover, is defined by only a single observation. Given the regime of a constant fraction of arbitrarily chosen and arbitrarily corrupted points, such a setting is highly undesirable.

A.3 PROOF OF THEOREM 2

Proof. First note that $\boldsymbol{W}_{encoder} \in \mathbb{R}^{r\times d}$ since we limit the dimension of the latent layer of the AutoEncoder to be $r = \text{rank}(\boldsymbol{L}^*)$. The problem we consider is:

$$\min_{\boldsymbol{L},\boldsymbol{E},\boldsymbol{W}_{encoder}\in\mathbb{R}^{r\times d}} \|\boldsymbol{W}_{encoder}\boldsymbol{L}\|_{2,1} + \lambda\|\boldsymbol{E}^T\|_{2,1}, s.t.\ \boldsymbol{H} = \boldsymbol{L} + \boldsymbol{E}, \boldsymbol{L} = \mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{L})) \tag{16}$$

Since we have the constraint $\boldsymbol{L} = \mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{L}))$, the rank of any feasible $\boldsymbol{L}$ should be no more than r, and $\boldsymbol{L}$ must lie within the row-space of $\boldsymbol{W}_{encoder}$. So we can write $\boldsymbol{W}_{encoder} = \boldsymbol{R}\boldsymbol{U}^T$ where $\boldsymbol{R} \in \mathbb{R}^{r\times r}$, and $\boldsymbol{U}$ are the top-r left singular vectors of $\boldsymbol{L}$. As the rows of $\boldsymbol{W}_{encoder}$ are orthornormal, *i.e.*, $\boldsymbol{W}_{encoder}\boldsymbol{W}_{encoder}^T = \boldsymbol{I}$, so $\boldsymbol{R}\boldsymbol{U}^T\boldsymbol{U}\boldsymbol{R}^T = \boldsymbol{I}$ and therefore $\boldsymbol{R}\boldsymbol{R}^T = \boldsymbol{I}$ (so this square matrix $\boldsymbol{R}$ is unitary). Further, since setting $\boldsymbol{W}_{decoder} = \boldsymbol{W}_{encoder}^T$ will always meet the constraint $\boldsymbol{L} = \mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{L}))$, the main problem we need to consider becomes:

$$\min_{\substack{\{\boldsymbol{L}|\text{rank}(\boldsymbol{L})\le r\},\boldsymbol{E},\\ \boldsymbol{W}_{encoder}=\boldsymbol{R}\boldsymbol{U}^T,\boldsymbol{R}\text{ is unitary}}} \|\boldsymbol{W}_{encoder}\boldsymbol{L}\|_{2,1} + \lambda\|\boldsymbol{E}^T\|_{2,1}, s.t.\ \boldsymbol{H} = \boldsymbol{L} + \boldsymbol{E}, \tag{17}$$

Since $\|\boldsymbol{L}\|_* = \|\boldsymbol{R}\boldsymbol{U}^T\boldsymbol{L}\|_* = \|\boldsymbol{W}_{encoder}\boldsymbol{L}\|_* = \|\boldsymbol{I}(\boldsymbol{W}_{encoder}\boldsymbol{L})\|_* = \|\sum_{i=1}^r \boldsymbol{e}_i(\boldsymbol{W}_{encoder}\boldsymbol{L})_{i,:}\|_* \le \sum_{i=1}^r \|\boldsymbol{e}_i(\boldsymbol{W}_{encoder}\boldsymbol{L})_{i,:}\| = \sum_{i=1}^r \|(\boldsymbol{W}_{encoder}\boldsymbol{L})_{i,:}\|_2 \triangleq \|\boldsymbol{W}_{encoder}\boldsymbol{L}\|_{2,1}$. The equality is achieved when $\boldsymbol{R} = \boldsymbol{I}$, *i.e.*, $\boldsymbol{W}_{encoder} = \boldsymbol{U}^T$, because $\|\boldsymbol{W}_{encoder}\boldsymbol{L}\|_{2,1} = \|\boldsymbol{U}^T\boldsymbol{L}\|_{2,1} = \|\boldsymbol{U}^T\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T\|_{2,1} = \|\boldsymbol{\Sigma}\boldsymbol{V}^T\|_{2,1} = \sum_{i=1}^r \sigma_i = \|\boldsymbol{L}\|_*$.

So

$$\min_{\substack{\{\boldsymbol{L}|\text{rank}(\boldsymbol{L})\le r\},\boldsymbol{E},\\ \boldsymbol{W}_{encoder}=\boldsymbol{R}\boldsymbol{U}^T,\boldsymbol{R}\text{ is unitary}}} \|\boldsymbol{W}_{encoder}\boldsymbol{L}\|_{2,1} + \lambda\|\boldsymbol{E}^T\|_{2,1} \ge \min_{\{\boldsymbol{L}|\text{rank}(\boldsymbol{L})\le r\},\boldsymbol{E}} \|\boldsymbol{L}\|_* + \lambda\|\boldsymbol{E}^T\|_{2,1}, \tag{18}$$

with equality achieved when $\boldsymbol{W}_{encoder}$ equals top-r left singular vectors of $\boldsymbol{L}$.

Then we only need to consider the right-hand-side of Eq. 18, *i.e.*,

$$\min_{\{\boldsymbol{L}|\text{rank}(\boldsymbol{L})\le r\},\boldsymbol{E}} \|\boldsymbol{L}\|_* + \lambda\|\boldsymbol{E}^T\|_{2,1} \quad s.t.\ \boldsymbol{H} = \boldsymbol{L} + \boldsymbol{E} \tag{19}$$

Recall that under the conditions of Theorem 1, the solution $\hat{\boldsymbol{L}}$ of Eq. 2 has exactly the same column space as $\boldsymbol{L}^*$ ($\hat{\boldsymbol{L}}$ may not and not necessary to be equal to $\boldsymbol{L}^*$), so $\text{rank}(\hat{\boldsymbol{L}}) = r$. Then the solution $\hat{\boldsymbol{L}}$ of Eq. 2 must also be the global optimal solution of Eq. 19. Finally, as the row-space of $\boldsymbol{W}_{encoder}$ equals the column-space of $\hat{\boldsymbol{L}}$, it recovers the underlying subspace of $\boldsymbol{L}^*$ exactly. □

A.4 PROOF OF THEOREM 3

Proof. A) Suppose $\mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}'))$ is the global optimal solution of Eq. 14 that is different from $\mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}^*))$. Let $\mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}')) = \mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}^*)) - \boldsymbol{v}$. So we have $\boldsymbol{v} \in Range(\boldsymbol{W}_{end})\backslash\boldsymbol{0}$.

$$\sum_{i\in\Omega} \|[\boldsymbol{h} - \mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}'))]_i\|_2 \tag{20}$$

$$= \sum_{i\in\Omega} \|[\boldsymbol{h} - \mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}^*)) + \boldsymbol{v}]_i\|_2 \tag{21}$$

$$= \sum_{i\in\Omega} \|[\boldsymbol{h} - \mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}^*))]_i + \boldsymbol{v}_i\|_2 \tag{22}$$

$$= \sum_{i\in S_\Omega} \|[\boldsymbol{h} - \mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}^*))]_i + \boldsymbol{v}_i\|_2 + \sum_{i\in \bar{S}_\Omega} \|[\boldsymbol{h} - \mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}^*))]_i + \boldsymbol{v}_i\|_2 \tag{23}$$

$$= \sum_{i\in S_\Omega} \|\boldsymbol{v}_i\|_2 + \sum_{i\in \bar{S}_\Omega} \|[\boldsymbol{h} - \mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}^*))]_i + \boldsymbol{v}_i\|_2 \tag{24}$$

$$\geq \sum_{i\in S_\Omega} \|\boldsymbol{v}_i\|_2 + \sum_{i\in \bar{S}_\Omega} \|[\boldsymbol{h} - \mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}^*))]_i\|_2 - \sum_{i\in \bar{S}_\Omega} \|\boldsymbol{v}_i\|_2 \tag{25}$$

$$> \sum_{i\in \bar{S}_\Omega} \|[\boldsymbol{h} - \mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}^*))]_i\|_2 \tag{26}$$

$$= \sum_{i\in\Omega} \|[\boldsymbol{h} - \mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}^*))]_i\|_2 \tag{27}$$

where S_Ω is the index set of size g such that $\boldsymbol{e}_i^* = \boldsymbol{0}, \forall i \in S_\Omega$. And the last inequality follows from the assumed range space property since $\boldsymbol{v} \in Range(\boldsymbol{W}_{end})\backslash\boldsymbol{0}$.

The above contradicts the assumption that $\mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}'))$ is the global optimal solution that is different from $\mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}^*))$.

B) First, note that $\boldsymbol{h}_\Omega = \mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}^*))_\Omega + \boldsymbol{e}_\Omega^*$ and $\sum_{i\in\Omega} \mathbb{1}\{\boldsymbol{e}_i^* \neq \boldsymbol{0}\} \leq |\Omega| - g$. Suppose $\mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}'))$ is the global optimal solution of Eq. 13 that is different from $\mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}^*))$. Let $\boldsymbol{h}_\Omega = \mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}'))_\Omega + \boldsymbol{e}'_\Omega$, then we have

$$\sum_{i\in\Omega} \mathbb{1}\{\boldsymbol{e}'_i \neq \boldsymbol{0}\} \leq \sum_{i\in\Omega} \mathbb{1}\{\boldsymbol{e}_i^* \neq \boldsymbol{0}\} \leq |\Omega| - g \tag{28}$$

and $\mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}'))_\Omega + \boldsymbol{e}'_\Omega = \mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}^*))_\Omega + \boldsymbol{e}_\Omega^*$.

From Eq. 28 we know that

$$\sum_{i\in\Omega} \mathbb{1}\{[\mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}^*)) - \mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}'))]_i = \boldsymbol{0}\} \geq |\Omega| - \sum_{i\in\Omega} \mathbb{1}\{\boldsymbol{e}_i^* \neq \boldsymbol{0}\} - \sum_{i\in\Omega} \mathbb{1}\{\boldsymbol{e}'_i \neq \boldsymbol{0}\} \tag{29}$$

$$\geq |\Omega| - (|\Omega| - g) - (|\Omega| - g) = 2g - |\Omega| \tag{30}$$

Let $\mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}')) = \mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}^*)) - \boldsymbol{v}$, so we have $\boldsymbol{v} \in Range(\boldsymbol{W}_{end})\backslash\boldsymbol{0}$ and $\sum_{i\in\Omega} \mathbb{1}\{\boldsymbol{v}_i = \boldsymbol{0}\} \geq 2g - |\Omega|$. Now split Ω into 3 disjoint sets $\{\Omega_0, \Omega_1, \Omega_2\}$, where Ω_0 is any subset of Ω with size $2g - |\Omega|$ such that $\boldsymbol{v}_{\Omega_0} = \boldsymbol{0}$, and $|\Omega_1| = |\Omega_2| = |\Omega| - g$. Since $|\Omega_0 \cup \Omega_1| = g$, by our assumption, we have $\sum_{i\in\Omega_0\cup\Omega_1} \|\boldsymbol{v}_i\|_2 > \sum_{i\in\Omega_2} \|\boldsymbol{v}_i\|_2$. Since $|\Omega_0 \cup \Omega_2| = g$, by our assumption, we have $\sum_{i\in\Omega_0\cup\Omega_2} \|\boldsymbol{v}_i\|_2 > \sum_{i\in\Omega_1} \|\boldsymbol{v}_i\|_2$. However, this leads to a contradiction since $\sum_{i\in\Omega_0} \|\boldsymbol{v}_i\|_2 = 0$. □

A.5 Proof of Theorem 4

Proof. Suppose $\mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}'))$ is the global optimal solution of Eq. 14 that is different from $\mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}^*))$. Let $\mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}')) = \mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}^*)) - \boldsymbol{v}$. So we have $\boldsymbol{v} \in Range(\boldsymbol{W}_{end})\backslash\boldsymbol{0}$.

$$\sum_{i\in\Omega} \|[\boldsymbol{h} - \mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}'))]_i\|_2 \tag{31}$$

$$= \sum_{i\in\Omega} \|[\boldsymbol{h} - \mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}^*)) + \boldsymbol{v}]_i\|_2 \tag{32}$$

$$= \sum_{i\in\Omega} \|\boldsymbol{v}_i\|_2 \tag{33}$$

$$> 0 \tag{34}$$

$$= \sum_{i\in\Omega} \|[\boldsymbol{h} - \mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}^*))]_i\|_2, \tag{35}$$

where the inequality is due to the condition that $\boldsymbol{v}_\Omega \neq \mathbf{0}$ for any $\boldsymbol{v} \in Range(\boldsymbol{W}_{end})\backslash\mathbf{0}$.

The above contradicts the assumption that $\mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}'))$ is the global optimal solution of Eq. 14 that is different from $\mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}^*))$.

Similarly, suppose $\mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}'))$ is the global optimal solution of Eq. 13 that is different from $\mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}^*))$. Let $\mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}')) = \mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}^*)) - \boldsymbol{v}$. So we have $\boldsymbol{v} \in Range(\boldsymbol{W}_{end})\backslash\mathbf{0}$.

$$\sum_{i\in\Omega} \mathbb{1}\{[\boldsymbol{h} - \mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}'))]_i \neq \mathbf{0}\} \tag{36}$$

$$= \sum_{i\in\Omega} \mathbb{1}\{[\boldsymbol{h} - \mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}^*)) + \boldsymbol{v}]_i \neq \mathbf{0}\} \tag{37}$$

$$= \sum_{i\in\Omega} \mathbb{1}\{\boldsymbol{v}_i \neq \mathbf{0}\} \tag{38}$$

$$> 0 \tag{39}$$

$$= \sum_{i\in\Omega} \mathbb{1}\{[\boldsymbol{h} - \mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}^*))]_i \neq \mathbf{0}\} \tag{40}$$

The above contradicts the assumption that $\mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}'))$ is the global optimal solution that is different from $\mathcal{D}_\phi(\mathcal{E}_\psi(\boldsymbol{l}^*))$.

□