GENERATIVE ADVERSARIAL FEDERATED MODEL

Anonymous authors
Paper under double-blind review

ABSTRACT

As an emerging technique, vertical federated learning collaborates with different data sources to jointly train a machine learning model without data exchange. However, federated learning is computationally expensive and inefficient in modeling due to complex encryption algorithms or secure computation protocols. Split learning offers a solution to the high computational cost and low modeling efficiency of traditional federated learning using encryption algorithms or secure computation protocols. However, vanilla split learning still suffers privacy leakage, especially the label leakage from the active party. Here, we propose the Generative Adversarial Federated Model (GAFM) built upon the vanilla split learning framework and the Generative Adversarial Network (GAN) for improved label privacy protection against commonly used attacks. In our empirical studies on two publicly available datasets, GAFM showed significant performance improvement for prediction and label privacy protection compared to existing models, including Marvell and SplitNN, which is an application of split learning to neural networks. We provide intuition on why GAFM can improve over SplitNN and Marvell and demonstrate that GAFM offers label protection through gradient perturbation compared to SplitNN.

1 INTRODUCTION

Federated learning collaboratively trains an algorithm across multiple decentralized remote devices or siloed data centers without exchanging their local data. Depending on different data partitioning methods, federated learning can be divided into three categories. They are horizontal federated learning (HFL), vertical federated learning(VFL), and federated transfer learning (Yang et al., 2019). In VFL, the data is vertically partitioned and participants whose datasets share the same sample space but differ in feature space (Hardy et al., 2017). With progressively stricter regulation of data privacy by government institutions such as the CCPA1 (Pardau, 2018) and GDPR3 (Voigt & Von dem Bussche, 2017), VFL is considered as one of the effective solutions for encouraging enterprise-level data collaborations, as it enables both collaboratively training and privacy protection. However, there are some concerns about VFL. One of these is the huge overhead in terms of memory costs and processing time. To provide a strong privacy guarantee for cross-silo training, VFL performs complex cryptographic operations, such as additively homomorphic encryption (Hardy et al., 2017) and secure multi-party computation (Mohassel & Zhang, 2017), which are extremely expensive to compute. It was measured that 80% of the time of the deep learning-based federated learning training process is spent on encryption and decryption(Zhang et al., 2020).

To solve this problem, split learning (Abuadbba et al., 2020; Gupta & Raskar, 2018; Vepakomma et al., 2018; Ceballos et al., 2020) was proposed. In vertical split learning, each participant (passive party) trains a partial model locally and then aggregates all outputs before feeding them to the server (active party). The active party then collaboratively train all participants' local models through back-propagation. Split learning allows multiple participants to jointly train the federated neural network and does not require encryption of intermediate gradients (intermediate computations of the cut layer), thus reducing the computation expense. SplitNN applies the idea of split learning to neural networks and has been used to analyze healthcare data (Ha et al., 2021; Poirot et al., 2019).

However, vanilla split learning approaches like SplitNN still face the privacy information leakage issue. In VFL, label leakage from gradients (LLG) (Wainakh et al., 2022) and passive party's feature leakage (Luo et al., 2021; Aono et al., 2017) are focused on simultaneously. It means VFL not only focuses on how much information of the passive party's features can be inferred by the active party,

but also focuses on how much label information can be inferred by the passive party during the training process. The security of split learning is based on the assumption that neural networks are black boxes and the attacker cannot recover the data and label information from the intermediate computations. Such an assumption does not actually hold, with intermediate gradient information flowing from the active party to passive party exposing not only the feature information but also the label information (Erdogan et al., 2021; Zhu et al., 2019). There are some approaches(Ceballos et al., 2020; Erdogan et al., 2021; Pereteanu et al., 2022; Titcombe et al., 2021) try to provide security for SplitNN, but they only focus on how to protect the passive party's feature privacy from being stolen by the active party, while neglecting the risk of exposure to highly sensitive information contained in labels.

Here, we propose Generative Adversarial Federated Model(GAFM) which protects label leakage from active party by combining SplitNN with generative adversarial networks (GANs) (Goodfellow et al., 2014). We reinterpret SplitNN from the generative adversarial perspective and view the task for each passive party as learning a generator using their local data whose aggregated distribution should be close to that of the response from the active party. The proposed GAFM framework offers better protection data and label privacy and against label stealing attacks without sacrificing the prediction performance. In this paper, we focus on applying GAFM to two-class classification problems, but we also demonstrate that it is applicable to other response types, like regression.

The paper is organized as following. In section 2, we provide more detailed descriptions on related work. We present GAFM model and its algorithm in section 3 and conduct numerical experiments comparing GAFM with competing methods on two publicly available data sets in section 4. We also provide numerical examples demonstrating that GAFM further protects the label information compared to SplitNN by outputting better mixed intermediate gradients.

2 Related Work

SplitNN-driven Vertical Partitioning Model. The SplitNN can be used in VFL which allows multiple participants with the same batch of samples but different features to learn the same distributed model without sharing data Gupta & Raskar (2018); Vepakomma et al. (2018); Ceballos et al. (2020). In SplitNN, each passive party trains partial neural network locally. The layer at which the active party and the passive parties share information (gradients from the active party and prediction from the passive parties) is called the cut layer. At the cut layer of SplitNN, each participant trains a fixed portion of the neural network locally and passes the intermediate results to the active party. The active party then aggregates intermediate results and implements backward propagation to update the local parameter of each each participant. There are various aggregation methods, such as element-wise average, element-wise maximum, element-wise sum, element-wise multiplication, concatenation or non-linear transformation (Ceballos et al., 2020).

Label protection via random gradient perturbation. To alleviate label information leak through intermediate gradients at the cut layer, one solution is adding randomness to the intermediate gradients which have been used the HFL (Geyer et al., 2017; Abadi et al., 2016; Hu et al., 2020). Marvell (Li et al., 2021) is a random perturbation technique which can be applied to the scenario of split learning for binary classification problems. Marvell protects the label privacy by perturbing the intermediate gradients. It aims to find the optimal zero-centered Gaussian perturbations to minimize the sum of KL divergences between the two perturbed distributions, under a budget constraint on the amount of perturbation added:

$$\min_{W^{(0)},W^{(1)}} \mathrm{KL}(\tilde{\mathrm{P}}^1 \| \tilde{\mathrm{P}}^0) + \mathrm{KL}(\tilde{\mathrm{P}}^0 \| \tilde{\mathrm{P}}^1), \quad \text{s.t.} \quad \alpha \mathrm{tr}(\Sigma_0) + (1-\alpha)\mathrm{tr}(\Sigma_1) \leq \mathrm{C},$$

where $W^{(k)} = \mathcal{N}(0, \Sigma_k)$, $k = 0, 1$, are the Gaussian perturbations added to class 0 and class 1, $\tilde{P}^k$ is the conversion of the original distribution for intermediate gradients from k and the Gaussian noise $W^{(k)}$ for $k = 0, 1$, and C is the budget for how much random perturbation Marvell is allowed to add. As the intermediate gradients with different labels become less distinguishable, intuitively, the difficulty for attackers to infer labels from gradients also increases.

Generative Adversarial Networks. The Generative Adversarial Network (GAN) train both a discriminator D and a generator G. The goal of the generator G is to learn the distribution P_g of the real data X. Sampling from a uniform or Gaussian distribution as the input variable Z, generator

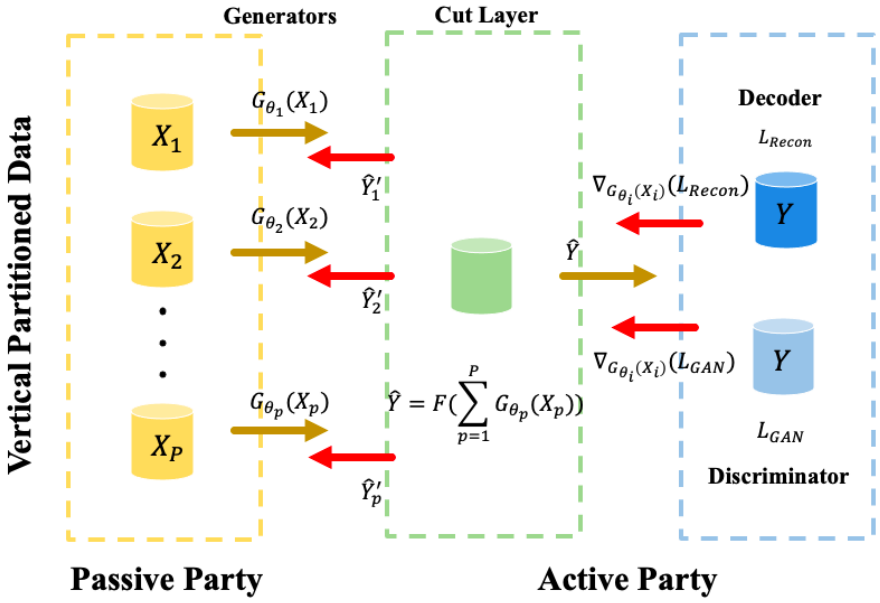


Figure 1: Systemic Structure of Generative Adversarial Federated Model

G then maps the input variable Z to the data space. The discriminator D aims to identify whether the distribution is from the real data or from the generated data. D and G continuously play a zero-sum game until convergence. For example, with cross-entropy loss, GAN considers the following optimization problem (Goodfellow et al., 2014):

$$\min_G \max_D L_{GAN}(G, D) = E_X[\log(D(X))] + E_Z[\log(1 - D(G(Z)))]. \tag{1}$$

GAN In Federated Learning. Previously, researchers have combined GAN and HFL for different purposes. In HFL, there are multiple clients who can access the same set of features and responses for different samples. GAN has been studied in this setting in three directions: (1) generating malicious attacks(Hitaj et al., 2017; Wang et al., 2019), (2) training high-quality GAN across distributed data under, e.g., privacy, constraints Hardy et al. (2019); Rasouli et al. (2020); Mugunthan et al. (2021), and also recently, (3) privacy protection of client data Wu et al. (2021). In FedCG, the authors let each client train a classifier predicting the response Y using a Z extracted from the original feature X, and a conditional GAN that learns the conditional distribution of Z given Y. The classifiers and generators (instead of the extracted Z) from different clients are then passed to the server for updating the common model. It has been shown that FedCG can effectively protect clients' data privacy. These previous works have demonstrated a promising role of GAN in HFL, but we are not aware of any prior work investigating the GAN model for label protection in VFL

3 Generative Adversarial Federated Model

3.1 Generative Adversarial Federated Model(GAFM)

In this section, we describe the Generative Adversarial Federated Model (GAFM), which enhances label privacy protection against identified attacks without sacrificing accuracy compared to the vanilla splitNN pipeline. Figure 1 demonstrates the model structure of GAFM.

GAFM passes the local prediction $\hat{Y}_p(X_p) = G_{\theta_p^g}(X_p)$ from each of the passive party to the active party, which are aggregated through $\hat{Y} = F(\sum_p \hat{Y}_p)$ for a final prediction minimizing the GAFM loss and $F(.)$ is a known transformation, e.g., the identify function or softmax function depending on the tasks. The GAFM loss consists of two parts. The first part is a GAN loss L_{GAN} defined in (2), which quantifies the distance between the distribution of $\hat{Y}$ and the distribution of Y through the discriminator. Here we define the GAN loss based on the Wasserstein-1 distance or Earth-Mover

distance. Compared to the classical GAN, the Wasserstein distance-based GAN can improve the stability of learning and get rid of problems such as pattern collapse (Arjovsky et al., 2017).

$$L_{GAN}(\theta^d, \theta^g; \varepsilon) = E_Y \left[D_{\theta^d}(Y + \varepsilon) \right] - E_X \left[D_{\theta^d}(\hat{Y}) \right] \tag{2}$$

where $\varepsilon \sim N(0, \sigma^2)$ is some additional additive Gaussian noise so that $Y + \varepsilon$ has a continuous support and not too concentrated in the space. By default, $\sigma^2 = 1$. The parameters θ^d represent model parameters for the discriminator and $\theta^g = (\theta^g_1, \ldots, \theta^g_{P_N})$ are the model parameters for the passive-party-specific generators. That is, the p passive parties locally train p models as generators based on their own data. The second part is a reconstruction loss L_{recon} shown in (3) which evaluates the prediction accuracy for the response label.

$$L_{recon}(\theta^g, \theta^m) = -E_{XY}[(Y \log(M_{\theta^m}(\hat{Y})) + (1 - Y) \log(1 - M_{\theta^m}(\hat{Y}))] \tag{3}$$

where M_{θ^m} is the decoder model which remaps the distribution generated by generators back to the label space to obtain the final prediction. The final GAFM loss combines L_{GAN} and L_{recon}, and is defined below in (4).

$$\min_{\theta^g, \theta^m} \max_{\theta^d} L_\varepsilon(\theta^d, \theta^g, \theta^m) = L_{GAN}(\theta^d, \theta^g; \varepsilon) + \gamma L_{recon}(\theta^g, \theta^m), \tag{4}$$

where $\gamma > 0$ is a tunable hyper-parameter. We discuss how it reconciles the trade-off between model performance and security in section 5.

The model parameters $\theta^g, \theta^d, \theta^m$ are updated sequentially:

(1) Update θ^d to maximize the GAN loss $\theta^d \leftarrow \arg\max_{\theta^d} L_{GAN}(\theta^d, \theta^g)$ given θ^g, θ^m.

(2) Update θ^m to minimize the reconstruction loss $\theta^m \leftarrow \arg\min_{\theta^m} L_{recon}(\theta^g, \theta^m)$ given θ^d, θ^g.

(3) Update θ^g to minimize the GAFM loss $\theta^g \leftarrow \arg\min_{\theta^g} L_{GAFM}(\theta^d, \theta^g, \theta^m)$ given θ^d, θ^m.

The final algorithm implements the above procedures with via stochastic gradient descent with learning rates α^d, α^g and α^m respectively. Algorithm 1 provides the details.

3.2 CAN GAFM PROTECTS LABEL STEALING ATTACKS?

We demonstrate that GAFM can provide more privacy protection for labels compared to vanilla SplitNN by offering a better mixed gradients. The vanilla splitNN does not have any privacy protection. In Li et al. (2021), the authors showed that a small KL divergences between the intermediate gradients from two classes can upper bound the maximal amount of information the attacker can steal for label inference.

Proposition 1 (Theorem 1 from Li et al. (2021))**.** *For $\epsilon \in [0, 4)$ and any perturbed distribution $\tilde{P}^1, \tilde{P}^0$ that are continuous with respect to each other,*

$$\mathrm{KL}(\tilde{\mathrm{P}}^1 \| \tilde{\mathrm{P}}^0) + \mathrm{KL}(\tilde{\mathrm{P}}^0 \| \tilde{\mathrm{P}}^1) \leq \epsilon \quad \textit{implies} \quad \max_{\mathrm{r}} \mathrm{AUC}_{\mathrm{r}} \leq \frac{1}{2} + \frac{\sqrt{\epsilon}}{2} - \frac{\epsilon}{8},$$

where r is a scoring function for how likely a sample is from class 1 utilizing only the intermediate gradients, and $\mathrm{AUC}_{\mathrm{r}}$ represents the achieved AUC using r.

Proposition 1 indicates if the GAFM model outputs more mixed intermediate gradients from the two classes, it is more likely to offer better protection against different types of label stealing attacks comapred to vanilla SplitNN.

In Figure 2, we show the scatter plots of the intermdiate gradients using vanilla SplitNN and GAFM from the two data sets Spambase and Credit, and we indeed observe that the intermediate gradients are much more mixed from GAFM. In particular, gradients from class 0 and class 1 using GAFM have centers much closer to each other compared to that using vanilla SplitNN.

3.3 TWO LABEL STEALING ATTACKS

This section discusses two attack methods by which the attacker can steal label information from intermediate gradients in the binary classification task.

Algorithm 1 Generative Adversarial Federated Model

Input : Training data $X_1, X_2, ..., X_P \in \mathbb{R}^{n \times d}$, training labels $Y \in \mathbb{R}^n$, epoch T, learning rates $\alpha^d, \alpha^g, \alpha^m$, clip value c

Initialize $\hat{Y}_1^0, \hat{Y}_2^0, ..., \hat{Y}_P^0$ from a Uniform or Gaussian distribution and θ_d^0,θ_g^0,θ_m^0

while *Not converge* **do**

  Active party

  `/* Step 1: Update the discriminator */`

  The active party computes $\hat{Y}(\sum_{i=1}^{P} \hat{Y}_p)$ and updates the discriminator by ascending its stochastic gradient $\theta^d \leftarrow \theta^d - \alpha^d \nabla_{\theta^d} L_{GAN}(\theta^d, \theta^g)$ and clip $\theta^d \leftarrow clip(\theta^d, -c, c)$.

  `/* Step 2: Update the decoder */`

  Then the active party updates the decoder by descending its stochastic gradient $\theta^m \leftarrow \theta^m - \alpha^m \nabla_{\theta^m} L_{recon}(\theta^g, \theta^m)$.

  `/* Step 3: Back-propagation */`

  The active party updates the intermediates $\hat{Y}_p$ by descending its stochastic gradient $\theta^g \leftarrow \theta^g - \alpha^g \nabla_{\hat{Y}_p} h(\hat{Y}_p)$, where $h(\hat{Y}_p) = L_{GAN}(\theta^d, \theta^g) + \gamma L_{recon}(\theta^g, \theta^m)$ is a function of the intermediate passive party outputs $(\hat{Y}_p)_{p=1}^P$.

  Passive parties

  for $p = 1, \ldots, P$ **do**

    `/* Step 4: Update generators */`

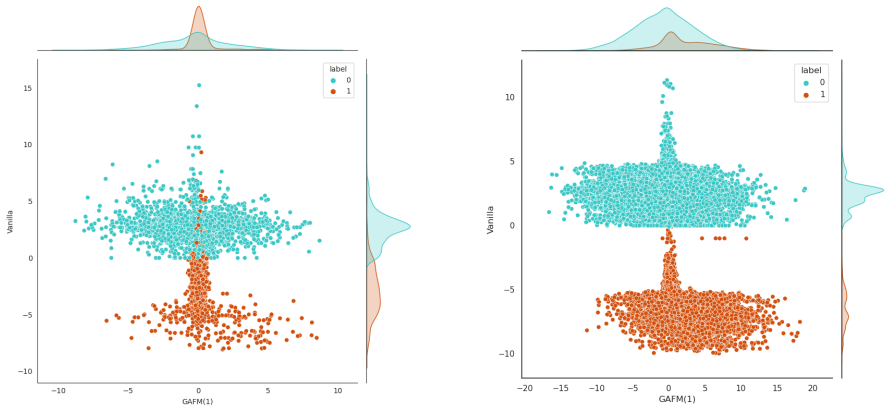
    Based on the locally determined model architecture and the target $\hat{Y}_p$, passive party p trains the local model G_{θ_p} (generator).

    `/* Step 5: Forward-propagation */`

    Then, update $\hat{Y}_p \leftarrow G_{\theta_p}(X_p)$.

  in parallel

end



(a) Spambase (b) Credit

Figure 2: GAFM ($\sigma = 1$) vs. Vanilla SplitNN: the intermediate gradients distribution g on Spambase and Credit (see experiments). Compared to Vanilla, GAFM has a much more mixed intermediate gradients distribution. It means the intermediate gradient distribution of label 1 and label 0 of GAFM have a larger overlap area, making it more difficult for the attacker to distinguish labels by gradients.

Norm Attack. Norm attack(Li et al., 2021) means norm-based attack which is a simple heuristic for black-box attack. It can be used for label inference in binary classification tasks. Norm attack is based on the observation that, for binary classification, the gradient norm $||g||_2$ of the positive instances are generally larger than that of the negative ones in both balanced and unbalanced data sets, thus the gradient norm $||g||_2$ is a strong predictor for labels.

Mean Attack. Mean-based attack is also a simple heuristic for black-box attack. In Li et al. (2021), the authors point out that the gradients from vanilla SplitNN tend to have a clustering structure with one class contain negative gradients and the other contain positive gradients. In GAFM model, this sign relationship does not necessarily hold anymore. However, we can still try to make attacks based on which cluster a sample is closer to. We assume that the attackers know the gradient centers of class 0 and class 1, which are denoted as μ_0 and μ_1. Let g_i be the intermediate gradient for sample i. The mean attacks assign the sample i to the cluster that g_i is closer to:

$$\hat{y_i} = \begin{cases} 1, if ||g_i - \mu_1||_2 \leq ||g_i - \mu_0||_2 \\ 0, else \end{cases} \tag{5}$$

4 EXPERIMENTS

In this section, experiments are conducted to demonstrate the performance and security of GAFM. We first introduce datasets in experiments. Only the binary classification task is considered in the experiment, which is a very common task in federated learning, especially in financial scenarios. In the Appendix A.3, we also provide more experimental results of GAFM on regression tasks. Secondly, we describe the experiment setup, mainly including the model architecture, baselines and evaluation metrics. Finally, we report the experiment results in terms of both performance and security, and also measure the performance-security trade-off under different γ .

4.1 DATASETS

We evaluate GAFM on two publicly available datasets. The specific information on datasets is as follows.

Spambase Dataset [1] collects 4061 emails with 57 attributes and they are classified as whether they are spam or not. And the proportion of positive instances is around 40%.

Credit Dataset [2] contains 24 attributes of credit card clients such as default payments, demographic factors and payment history. The labels for the Credit dataset is a binary variable which shows whether the client is at risk of default. And the proportion of positive instances is around 22%.

Spambase and Credit datase have different degrees of imbalance, which allows us to evaluate the model in different label distribution scenarios. In addition, the Spambase dataset is a linearly separable dataset on which the logistic regression model has a good performance(AUC). The Credit dataset, however, is not linearly separable. Datasets with different degrees of linear separability also allow for a more comprehensive assessment of GAFM.

4.2 EXPERIMENT SETUP

Model Architecture We consider the two-party setting where there is only one passive party, which poses the highest danger for label stealing. More specifically, the active party only has the labels without feature data and the one passive party has all feature data but not the labels. The decoder, discriminator and encoder are all one-layer neural networks with different activation functions:

- Generator: a single hidden layer neural network with a LeakyReLu activation function.
- Decoder: a single hidden layer neural network with a Sigmoid activation function.
- Discriminator:a single hidden layer neural network with a LeakyReLu activation function.

[1] https://archive.ics.uci.edu/ml/datasets/spambase

[2] https://www.kaggle.com/datasets/uciml/default-of-credit-card-clients-dataset

Table 1: Average AUC of different models.

Model	Spambase		Credit	
	Train	Test	Train	Test
GAFM(0.25)	0.970±0.005	0.961±0.007	0.685±0.152	0.683±0.131
GAFM(1)	**0.970±0.013**	**0.962±0.013**	**0.725±0.005**	**0.719±0.008**
GAFM(3)	0.964±0.007	0.957±0.012	0.719±0.005	0.712±0.006
Marvell	0.787±0.099	0.784±0.097	0.711±0.006	0.708±0.010
Vanilla	0.961±0.005	0.950±0.005	0.718±0.005	0.712±0.008
Logistic Regression	0.931	0.911	0.500	0.499

The transformation function $F(.)$ is again a LeakyReLu function. We carry out our numerical experiments with parameter $\sigma = 0.25, 1, 3$ to investigate the influence of σ on the performance of GAFM. For both Spambase and Credit datasets, we pick up the clip value $c = 0.01$,$\gamma = 300$, the learning rate $\alpha^d = \alpha^m = \alpha^g = 0.01$ and the training epoch $T = 300$.

Baselines We compare the performance and security of GAFM with Marvell model and vanilla SplitNN on the two public data sets. We also report on the performance of logistic regression which can be regarded as a simple one-layer neural network.

Evaluation Metrics We use AUC to evaluate the model performance and use the leak AUC to measure the level of information leak. The leak AUC is defined as the AUC achieved using the defined attacks, e.g., noram attack or mean attack. When leak AUC is close to 1, it implies that the attacker can very accurately recover the private label, whereas a small leak AUC value indicates that the attacker is less informative in predicting the labels. In our experiments, to account for the possibility that a simple label flipping may lead to higher leak AUC, we modify the leak AUC for a predefined attack as the following,

$$\text{leakAUC} \leftarrow \max(\text{leakAUC}, 1 - \text{leakAUC}).$$

That is, we flip our label assignment if this leads to higher AUC, and the resulting leak AUC is guaranteed to be no smaller than 0.5. Leak AUC have been widely used as a metric for measuring label leakage in previous work (Li et al., 2021; Yang et al., 2022; Sun et al., 2022).

4.3 Results

In this section, we report the performance of these three models and then demonstrate the label protection as well as the performance-security trade-off under different γ.

4.3.1 Evaluation of Prediction accuracy and security

We compare GAFM with different baselines using AUC and Leak AUC. The first metric measures the prediction accuracy, and the latter measures the degree of label leakage.

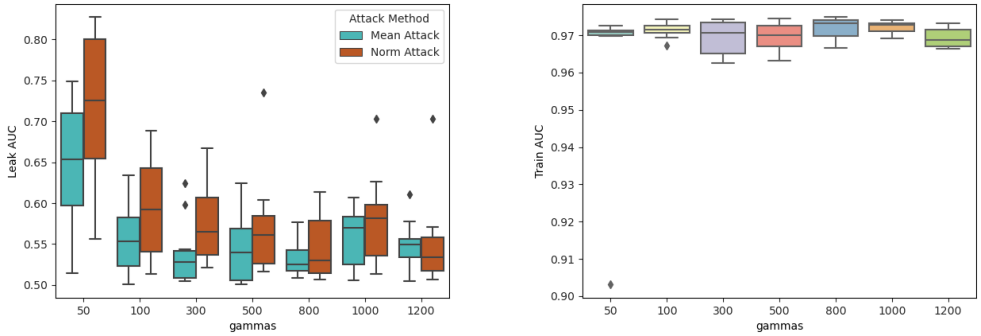
For each dataset, we split the data into training and test, perform model training on the training data and evaluate them on the test. We repeat it five times and take the average AUC across different repetitions (one repetition for Logistic regression). Table 1 shows the achieved average AUC for different methods on the Spambase and Credit datasets. GAFM achieves the best performance (AUC) on both datasets, even better than the vanilla SplitNN. Marvell experiences a performance loss due to perturbing the intermediate gradients with Gaussian noise.

Next, we compare GAFM with Vanilla and Marvell for label protection against the norm and mean attacks. Table 2 shows the average Leak AUC across 5 repetitions. We show the Leak AUC achieved from different runs in Appendix A.1. We see that the vanilla model almost experiences high leak AUC using norm and mean attacks. In contrast, GAFM and Marvell can protect against mean and norm attacks with relevant Leak AUC much closer to 0.5. GAFM is particularly good at protecting against mean attacks.

Comparing the GAFM models with different additional noise levels σ, we observe that a moderately large σ does not lead to severe deterioration in prediction accuracy but can provide more robust protection against label stealing attacks.

Table 2: Average Leak AUC using different models.

Model	Spambase		Credit	
	Norm Attack	Mean Attack	Norm Attack	Mean Attack
GAFM(0.25)	0.577±0.058	**0.529±0.026**	0.785±0.251	0.506±0.007
GAFM(1)	0.633±0.125	0.610±0.101	0.675±0.167	**0.503±0.005**
GAFM(3)	**0.567±0.157**	0.549±0.126	**0.606±0.128**	0.504±0.005
Marvell	0.614±0.078	0.584±0.076	0.666±0.138	0.559±0.007
Vanilla	0.886±0.052	0.806±0.030	0.675±0.163	0.601±0.071



(a) Spambase: Leak AUC

(b) Spambase: Train AUC

Figure 3: Performance (train AUC) vs. Security (leak AUC) trade-off of GAFM ($\sigma = 1$) under two attacks on Spambase.

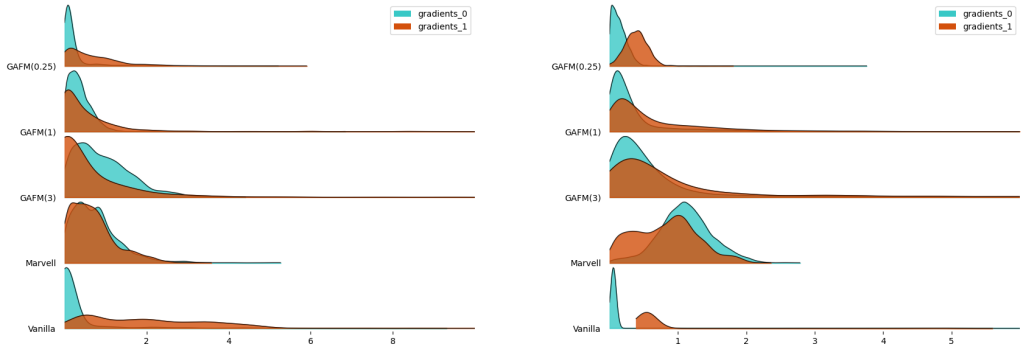
4.3.2 Performance-Security Trade-off

We further investigate the relationship between performance and security of GAFM given different γ. Figure 3 shows the trade-off between security (leak AUC) and performance (train AUC) on Spambase (see Appendix A.2 for Credit datasets). And we conduct ten training runs for each γ. Taking the median as an example, we find that as γ increases, train AUC, which represents the performance of GAFM, remains almost stable or increases slightly. But the leak AUC, which represents the degree of privacy leakage, shows a U-shape as γ increases. At the start stage, the leak AUC gradually decreases as γ increases; then, the leak AUC gradually increases as γ increases. Empirically, we observe good performance for a wide range γ, e.g., $\gamma \in [300, 800]$ for both the Spambase and Credit datasets. In practice, we can determine the order of magnitude of γ by comparing the absolute value of GAN loss with that of reconstruction loss so that their contribution to the final loss is comparable.

5 Discussions

This paper presents the Generative Adversarial Federated Model (GAFM), which does not rely on complex encryption computing and can be applied to various vertical federated learning tasks. Our empirical experiments demonstrate GAFM as a promising approach for effective learning with privacy protection against label stealing attacks.

One popular way of guarding against label stealing attacks is to pass a noisy version of the intermediate gradients from the active party to the passive parties, as in Marvell, which leads to better mixing of gradients from two classes in a binary classification problem. The GAFM takes a different path. Interestingly, by introducing a GAN loss into the learning task for the active party, we also observe improved gradient mixing compared to the vanilla SplitNN. This is demonstrated in Figure 2: the centers of the two gradient distributions from class 0 and class 1 are much closer to each other than that from vanilla SplitNN. In Figure 4, we examine the relationships between the gradient norms and the class label and plot the distributions of the magnitudes of the gradient for two classes from Vanilla SplitNN, Marvell, and GAFM in our numerical examples where we have selected the



(a) Spambase (b) Credit

Figure 4: GAFM (σ=0.25,1,3) vs. Marvell vs. Vanilla: the distribution of the intermediate gradients norm values $||g||_2$.Here we select the best gradient norm separation across all repetitions for each model.

most separable run from each model when making the plots. We observe that the distributions of the magnitudes become more spread out and better mixed as σ becomes larger. Interestingly, although the gradients from GAFM have better mixing as in Marvell for moderately large σ, we do not observe model performance deterioration compared to Vanilla SplitNN.

The goal of this paper is to introduce the GAFM framework. Although we have considered a simple model architecture with decoders/encoders/discriminators being one-layer neural networks, we can apply it to other types of local models for the passive parties. For example, in the Credit data set, compared to the simple one-layer neural network as the local generator, the Xgboost model may be a much better base learner, which can achieve 0.999/0.971 (train AUC/test AUC) on Spambase and 0.776/0.649 (train AUC/test AUC) on Credit without parameters tuning. GAFM can also be applied to the setting where the local model is trained using more complicated models like Xgboost where we can still pass along the gradients for model updates at the passive parties without label leakage.

We also want to point out that GAFM and Marvell do not compete with each other. A nice property about Marvel is that it can guarantee the upper bound on label leak from the gradients, while GAFM provides empirically good performance but no theoretical guarantees. We can combine Marvell and GAFM by adding optimized random noise to the GAFM gradients to control the sum of KL divergences between the two perturbed distributions. This combination will automatically adapt to the amount of mixing offered by GAFM and add less or no random noise compared to the original Marvell when GAFM is doing well.

REFERENCES

Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pp. 308–318, 2016.

Sharif Abuadbba, Kyuyeon Kim, Minki Kim, Chandra Thapa, Seyit A Camtepe, Yansong Gao, Hyoungshick Kim, and Surya Nepal. Can we use split learning on 1d cnn models for privacy preserving training? In *Proceedings of the 15th ACM Asia Conference on Computer and Communications Security*, pp. 305–318, 2020.

Yoshinori Aono, Takuya Hayashi, Lihua Wang, Shiho Moriai, et al. Privacy-preserving deep learning via additively homomorphic encryption. *IEEE Transactions on Information Forensics and Security*, 13(5):1333–1345, 2017.

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pp. 214–223. PMLR, 2017.

Iker Ceballos, Vivek Sharma, Eduardo Mugica, Abhishek Singh, Alberto Roman, Praneeth Vepakomma, and Ramesh Raskar. Splitnn-driven vertical partitioning. *arXiv preprint arXiv:2008.04137*, 2020.

Ege Erdogan, Alptekin Kupcu, and A Ercument Cicek. Splitguard: Detecting and mitigating training-hijacking attacks in split learning. *arXiv preprint arXiv:2108.09052*, 2021.

Robin C Geyer, Tassilo Klein, and Moin Nabi. Differentially private federated learning: A client level perspective. *arXiv preprint arXiv:1712.07557*, 2017.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

Otkrist Gupta and Ramesh Raskar. Distributed learning of deep neural network over multiple agents. *Journal of Network and Computer Applications*, 116:1–8, 2018.

Yoo Jeong Ha, Minjae Yoo, Gusang Lee, Soyi Jung, Sae Won Choi, Joongheon Kim, and Seehwan Yoo. Spatio-temporal split learning for privacy-preserving medical platforms: Case studies with covid-19 ct, x-ray, and cholesterol data. *IEEE Access*, 9:121046–121059, 2021.

Corentin Hardy, Erwan Le Merrer, and Bruno Sericola. Md-gan: Multi-discriminator generative adversarial networks for distributed datasets. In *2019 IEEE international parallel and distributed processing symposium (IPDPS)*, pp. 866–877. IEEE, 2019.

Stephen Hardy, Wilko Henecka, Hamish Ivey-Law, Richard Nock, Giorgio Patrini, Guillaume Smith, and Brian Thorne. Private federated learning on vertically partitioned data via entity resolution and additively homomorphic encryption. *arXiv preprint arXiv:1711.10677*, 2017.

Briland Hitaj, Giuseppe Ateniese, and Fernando Perez-Cruz. Deep models under the gan: information leakage from collaborative deep learning. In *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security*, pp. 603–618, 2017.

Rui Hu, Yuanxiong Guo, Hongning Li, Qingqi Pei, and Yanmin Gong. Personalized federated learning with differential privacy. *IEEE Internet of Things Journal*, 7(10):9530–9539, 2020.

Oscar Li, Jiankai Sun, Xin Yang, Weihao Gao, Hongyi Zhang, Junyuan Xie, Virginia Smith, and Chong Wang. Label leakage and protection in two-party split learning. *arXiv preprint arXiv:2102.08504*, 2021.

Xinjian Luo, Yuncheng Wu, Xiaokui Xiao, and Beng Chin Ooi. Feature inference attack on model predictions in vertical federated learning. In *2021 IEEE 37th International Conference on Data Engineering (ICDE)*, pp. 181–192. IEEE, 2021.

Payman Mohassel and Yupeng Zhang. Secureml: A system for scalable privacy-preserving machine learning. In *2017 IEEE symposium on security and privacy (SP)*, pp. 19–38. IEEE, 2017.

Vaikkunth Mugunthan, Vignesh Gokul, Lalana Kagal, and Shlomo Dubnov. Bias-free fedgan: A federated approach to generate bias-free datasets. *arXiv preprint arXiv:2103.09876*, 2021.

Stuart L Pardau. The california consumer privacy act: Towards a european-style privacy regime in the united states. *J. Tech. L. & Pol'y*, 23:68, 2018.

George-Liviu Pereteanu, Amir Alansary, and Jonathan Passerat-Palmbach. Split he: Fast secure inference combining split learning and homomorphic encryption. *arXiv preprint arXiv:2202.13351*, 2022.

Maarten G Poirot, Praneeth Vepakomma, Ken Chang, Jayashree Kalpathy-Cramer, Rajiv Gupta, and Ramesh Raskar. Split learning for collaborative deep learning in healthcare. *arXiv preprint arXiv:1912.12115*, 2019.

Mohammad Rasouli, Tao Sun, and Ram Rajagopal. Fedgan: Federated generative adversarial networks for distributed data. *arXiv preprint arXiv:2006.07228*, 2020.

Jiankai Sun, Xin Yang, Yuanshun Yao, and Chong Wang. Label leakage and protection from forward embedding in vertical federated learning. *arXiv preprint arXiv:2203.01451*, 2022.

Tom Titcombe, Adam J Hall, Pavlos Papadopoulos, and Daniele Romanini. Practical defences against model inversion attacks for split neural networks. *arXiv preprint arXiv:2104.05743*, 2021.

Praneeth Vepakomma, Otkrist Gupta, Tristan Swedish, and Ramesh Raskar. Split learning for health: Distributed deep learning without sharing raw patient data. *arXiv preprint arXiv:1812.00564*, 2018.

Paul Voigt and Axel Von dem Bussche. The eu general data protection regulation (gdpr). *A Practical Guide, 1st Ed., Cham: Springer International Publishing*, 10(3152676):10–5555, 2017.

Aidmar Wainakh, Fabrizio Ventola, Till Müßig, Jens Keim, Carlos Garcia Cordero, Ephraim Zimmer, Tim Grube, Kristian Kersting, and Max Mühlhäuser. User-level label leakage from gradients in federated learning. *Proceedings on Privacy Enhancing Technologies*, 2022(2):227–244, 2022.

Zhibo Wang, Mengkai Song, Zhifei Zhang, Yang Song, Qian Wang, and Hairong Qi. Beyond inferring class representatives: User-level privacy leakage from federated learning. In *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*, pp. 2512–2520. IEEE, 2019.

Yuezhou Wu, Yan Kang, Jiahuan Luo, Yuanqin He, and Qiang Yang. Fedcg: Leverage conditional gan for protecting privacy and maintaining competitive performance in federated learning. *arXiv preprint arXiv:2111.08211*, 2021.

Qiang Yang, Yang Liu, Yong Cheng, Yan Kang, Tianjian Chen, and Han Yu. Federated learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 13(3):1–207, 2019.

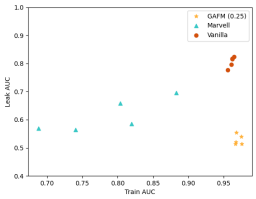
Xin Yang, Jiankai Sun, Yuanshun Yao, Junyuan Xie, and Chong Wang. Differentially private label protection in split learning. *arXiv preprint arXiv:2203.02073*, 2022.

Chengliang Zhang, Suyi Li, Junzhe Xia, Wei Wang, Feng Yan, and Yang Liu. {BatchCrypt}: Efficient homomorphic encryption for {Cross-Silo} federated learning. In *2020 USENIX Annual Technical Conference (USENIX ATC 20)*, pp. 493–506, 2020.

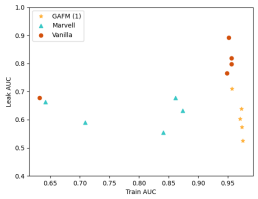
Ligeng Zhu, Zhijian Liu, and Song Han. Deep leakage from gradients. *Advances in Neural Information Processing Systems*, 32, 2019.

A Additional Experiment Results

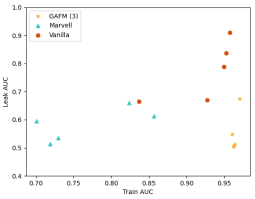
A.1 Distribution of Leak AUC across repetitions



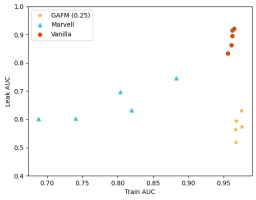
(a.1) Leak AUC ($\sigma = 0.25$)



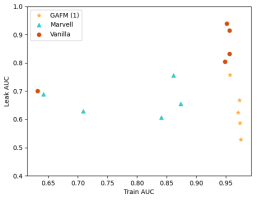
(b.1) Leak AUC ($\sigma = 1$)



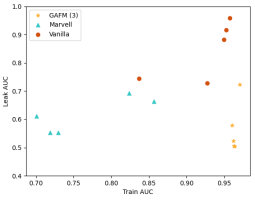
(c.1) Leak AUC ($\sigma = 3$)



(a.2) Leak AUC ($\sigma = 0.25$)

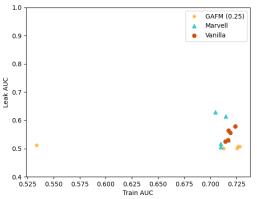


(b.2) Leak AUC ($\sigma = 1$)

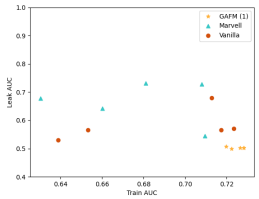


(c.2) Leak AUC ($\sigma = 3$)

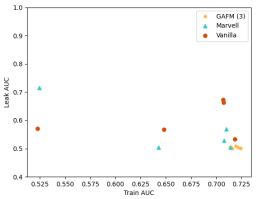
Figure 5: Spambase: (a.1) $\sim$ (c.1) are results under the mean attack and (a.2) $\sim$ (c.2) are results under the norm attack



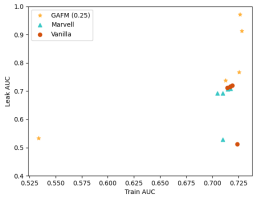
(a.1) Leak AUC ($\sigma = 0.25$)



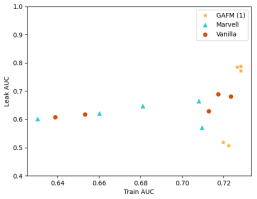
(b.1) Leak AUC ($\sigma = 1$)



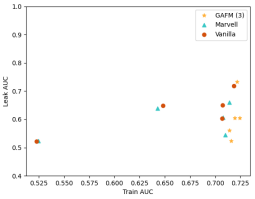
(c.1) Leak AUC ($\sigma = 3$)



(a.2) Leak AUC ($\sigma = 0.25$)



(b.2) Leak AUC ($\sigma = 1$)



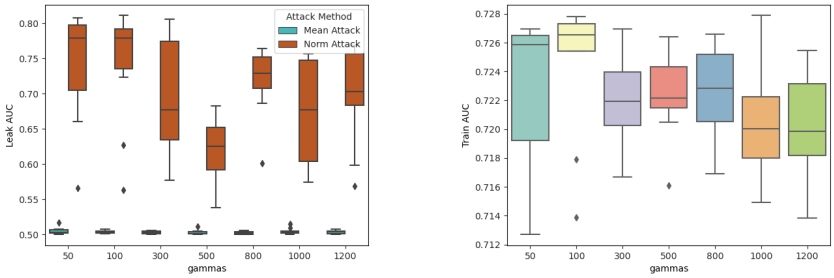
(c.2) Leak AUC ($\sigma = 3$)

Figure 6: Credit: (a.1) $\sim$ (c.1) are results under the mean attack and (a.2) $\sim$ (c.2) are results under the norm attack

Figure 5 and 6 show the security results of GAFM, Marvell and Vanilla under norm and mean attacks. The X-axis represents the performance of the model, measured by train AUC, and the Y-axis represents the privacy of the model, measured by leak AUC. We can observe that in most of the figures, GAFM is clustered in the lower right corner, which illustrates the advantages of GAFM over Marvell and Vanilla in terms of both performance and privacy.

A.2 TRADE-OFF RESULTS

In Figure 7, we show the trade-off results on Credit data in section 4.3.2.Similar to the Spambase, the leak AUC decreases and then increases as γ increases, and the train AUC remains almost stable. For mean attack, the leak AUC of GAFM are around 0.5.



(a) Leak AUC (b) Train AUC

Figure 7: Performance (train AUC) vs. Security (leak AUC) trade-off of GAFM under two attacks on Credit

A.3 GAFM ON REGRESSION

As stated in the main text, GAFM can also be applied to regression tasks. In this section, we train GAFM on two public regression datasets. They are (1) Boston Housing dataset [3] with 14 attributes and 506 instances and (2) Diabates(regression) dataset [4] with 11 attributes and 442 instances. And the loss function is:

$$\begin{aligned} \min_{\theta^g,\theta^m} \max_{\theta^d} L(\theta^d,\theta^g,\theta^m) = E_Y[D_{\theta^d}(y+\epsilon)] - E_X[D_{\theta^d}(F(G_{\theta^g}(x)))] \\ + E_{XY}[(y - M_{\theta^m}(F(G_{\theta^g}(x))))^2] \end{aligned} \tag{6}$$

The average Pearson correlation coefficients (conduct 10 training runs for each model) are reported in Table 3. The ρ_g here is the average Pearson correlation coefficient between the intermediate gradients g and the true label, which is $\rho_g = corr(g, Y)$. $\rho_{\hat{Y}}$ is the average Pearson correlation coefficient between the prediction $\hat{Y}$ and the true label, which is $\rho_{\hat{Y}} = corr(\hat{Y}, Y)$. We also report the results of linear regression (LR) as a baseline.

Table 3: Average Pearson correlation coefficient of different models ($\gamma = 800$)

		GAFM		Vanilla		LR
		ρ_g	$\rho_{\hat{Y}}$	ρ_g	$\rho_{\hat{Y}}$	$\rho_{\hat{Y}}$
Boston	Train	0.323±0.546	0.837±0.036	0.869±0.000	0.871±0.000	0.869
	Test	0.300±0.514	0.787±0.031	0.814±0.001	0.816±0.001	0.815
Diabetes	Train	0.283±0.425	0.708±0.000	0.708±0.000	0.708±0.000	0.708
	Test	0.304±0.456	0.759±0.000	0.760±0.000	0.760±0.000	0.759

By comparing the Pearson correlation coefficient of GAFM and baselines, we conclude that GAFM also can handle regression problems well. According to the definition of ρ_g and $\rho_{\hat{Y}}$, we know that if the passive party (the attacker) wants to infer the label through the intermediate gradient, the ρ_g should be positive. And larger the value, more label information the attacker can infer. Table 3

[3] https://www.cs.toronto.edu/~delve/data/boston/bostonDetail.html

[4] https://scikit-learn.org/stable/modules/generated/sklearn.datasets.load_diabetes.html

shows that the ρ_g of the GAFM model is very small and even less than 0 sometimes, which implies that it is difficult for the attacker to steal valid label information from the intermediate gradient.At the same time, the $\rho_{\hat{Y}}$ of the GAFM model is very close to that of linear regression model and the Vanilla model, which implies GAFM also can handle the regression prediction. In contrast, the ρ_g of Vanilla model is almost the same as its $\rho_{\hat{Y}}$. It means the attacker can easily infer the label from the intermediate gradients of Vanilla.