Participants sample generation method based on correlated attributes generation and vertical federated imputation

Author information scrubbed for double-blind reviewing

No Institute Given

Abstract. When multiple parties align their samples for vertical federation learning, some participants may lack certain samples that other participants possess, resulting in insufficient joint samples and ultimately impairing performance. To address this issue, we propose a novel Participants Sample Generation method based on Correlated Attributes Generation and vertical federated imputation, abbreviated as FedPSG-CAG. Overall, to generate the missing samples, FedPSG-CAG first generates some highly correlated attributes for those samples and then impute the remaining attributes by vertical federated imputation models. Specifically, for each participant with missing samples, we leverage the aligned sample set of that participant to identify its highly-correlated attributes. These attributes are then utilized to train a local generation model, generating those highly-correlated attributes data for the missing samples of that participant. Then, a vertical federated imputation framework based on GANs is constructed to generate the remaining attributes of these missing samples. In this federated imputation framework, we redesign the model structure, loss function and training process based on GANs. Experiments on multiple public datasets have thoroughly validated that our method outperforms the state-of-the-art baseline models currently available for participants sample generation in vertical federation learning.

Keywords: Participants sample generation · Vertical federated learning · Correlated attributes generation · Data imputation.

1 Introduction

In many practical applications, multiple organizations collaboratively utilize their data to build and train more powerful machine learning models. For example, banks and e-commerce companies may improve credit risk prediction for their common customer group by integrating distinct feature sets from each organization. Vertical Federated Learning (VFL) [1]is a privacy-preserving distributed machine learning approach designed for scenarios where multiple participants share the same sample ID but possess different features. However, in many application of VFL setting, some participants may lack samples that other participants possess, as shown in Fig.1. These missing samples cannot undergo

multi-party cryptographic alignment, resulting in a reduced number of joint samples available for federated machine learning model training. Therefore, in the VFL setting, generating the missing samples for participants to address the issue of insufficient joint samples after multi-party alignment, is a valuable research direction.

To address the issue of insufficient joint samples after multi-party sample alignment caused by the missing samples in some participants, certain methods for generating joint samples can be employed. Currently, several sample generation methods based on Vertical Federated Learning (VFL) enable the generation of joint samples from multiple participants. For instance, vertical federated data generation methods based on GANs: FedDA [2], VertiGAN [3], VFLGAN [4], GTV [5], and vertical federated tabular data generation methods based on Markov Random Fields (MRFs) : VertiMRF [6]. However, these methods yield completely new joint samples, and the data across all parties is synthesized and non-real. Moreover, when the missing sample ratio of some participants is high, a few joint samples are available, making it impossible to train an excellent vertical federation generation model. Therefore, methods for generating joint samples do not guarantee the acquisition of high - quality joint samples.

Besides the methods for generating joint samples, samples generation can be implemented locally for the participants with missing samples. The generated samples can then be aligned with the unaligned samples from other participants to obtain more joint samples. This is the key problem and method addressed in this paper. Currently, various high-performing methods are available for generating these missing samples, such as Generative Adversarial Networks (GANs) (including GAN [7], CGANs [8], CTGAN [9], TableGAN [10], and CTAB-GAN [11]), Autoencoders (AEs) (such as AE [12] and VAE [13]), and Denoising Diffusion Probabilistic Models (DDPMs) (e.g., TabDDPM [14]). These methods learn locally from the data of the participant with missing sapmles, and do not consider the influence of other parties. However, in the VFL setting, there are always certain associations among the participants' data. When these models are applied locally within each party, they overlook the role and impact of multi-party data associations on the generation results.

To address these challenges,this paper proposes a Participants Sample Generation method based on Correlated Attributes Generation and vertical federated imputation, referred to FedPSG-CAG. It aims to generate samples for participants with missing samples, thereby providing more high-quality joint samples for federated machine learning. The contributions of this paper are as follows:

(1) To generate the missing samples, FedPSG-CAG first generates some highly correlated attributes for those samples and then impute the remaining attributes by vertical federated imputation models. For each participant with missing samples, we leverage the aligned sample set of that participant to identify its highly-correlated attributes. These attributes are then utilized to train a local generation model, generating those highly-correlated attributes data for the missing samples of that participant.

(2) This paper proposes a framework of vertical federated imputation based on GANs. In this federated imputation framework, we redesign the model structure, loss function and training process based on GANs. By incorporating data from other parties, this framework is used to generate the remaining attributes of these missing samples.

2 Problem Description

Suppose there are N data owners (N participants) and a central server in a given vertical federated learning scenario. To clearly illustrate the methodology of this paper, we take the example of two participants: Party A and Party B, with their trusted collaborator C serving as the central server. Party A and Party B possess sensitive data and are required to safeguard data privacy during their collaborative model-training process. We apply various preprocessing techniques to the data held by Parties A and B, including data cleaning, normalization, and feature encoding. After preprocessing, the two parties securely align their samples based on ID spaces by the Blind RSA-based Private Set Intersection (PSI) protocol [15].

Suppose the sample set in Party A as D_A, $D_A = \{X_1^A, \ldots, X_i^A, \ldots, X_{d_A}^A\}$, where $X_i^A = (x_{i1}^A, \ldots, x_{im}^A, \ldots, x_{ia}^A)$ is the i-th sample of Party A, x_{im}^A is the m-th attribute of the i-th sample, $i = 1, \ldots, d_A$, $m = 1, \ldots, a$. d_A denotes the number of samples in Party A, and a denotes the number of attributes in Party A's samples.

In Party B, the sample set is represented as D_B, $D_B = \{X_1^B, \ldots, X_i^B, \ldots, X_{d_B}^B\}$, where $X_i^B = (x_{i1}^B, \ldots, x_{in}^B, \ldots, x_{ib}^B)$ is the i-th sample of Party B, x_{in}^B is the n-th attribute of the i-th sample, $i = 1, \ldots, d_B$, $n = 1, \ldots, b$. d_B denotes the number of samples in Party B, and b denotes the number of attributes in Party B's samples.

Parties A and B align their datasets, D_A and D_B, based on their sample IDs through encrypted matching, resulting in $d_{ID_A \cap ID_B}$ aligned samples. The $d_A - d_{ID_A \cap ID_B}$ samples in Party A cannot find corresponding aligned samples in Party B. The $d_B - d_{ID_A \cap ID_B}$ samples in Party B cannot find corresponding aligned samples in Party A. In Party A, the aligned sample set is represented as $D_{\text{Align}}^A = \{X_1^A, \ldots, X_{d_{ID_A \cap ID_B}}^A\}$, and the unaligned sample set is $D_{\text{UnAlign}}^A = \{X_{d_{ID_A \cap ID_B}+1}^A, \ldots, X_{d_A}^A\}$. In Party B, the aligned sample set is represented as $D_{\text{Align}}^B = \{X_1^B, \ldots, X_{d_{ID_A \cap ID_B}}^B\}$, and the unaligned sample set is $D_{\text{UnAlign}}^B = \{X_{d_{ID_A \cap ID_B}+1}^B, \ldots, X_{d_B}^B\}$. As illustrated in Fig. 1, the number of samples in D_{Align}^A equals that in D_{Align}^B, and they have identical sample IDs.

Consequently, the problem addressed in this paper is as follows: in the VFL settin, the data correlation within the participants and the data associations among multi-parties are utilized to generate the missing samples in Party A, and the missing samples in Party B, as shown in Fig. 1. These generated samples can then be aligned with the unaligned samples from Party A or Party B, resulting

in a larger set of joint samples that contain more real data. It aims to improve the training performance of vertical federated learning models.

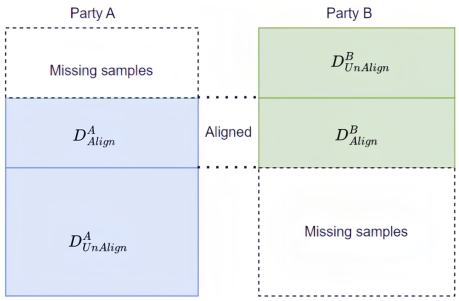

Fig. 1. Diagram of participants with missing samples.

3 Methods

The overall process of the proposed method FedPSG-CAG is illustrated in Fig. 2. This method consists of two stages: correlated attribute generation, vertical federated imputation based on GANs.

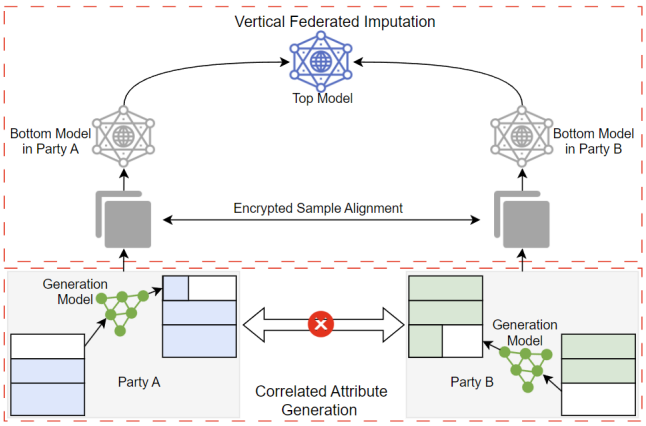

Fig. 2. The overall process of the proposed method FedPSG-CAG.

3.1 Correlated attribute generation

Since generation models can better focus on and capture the data distribution of highly-correlated attributes during their learning process, we leverage the aligned sample set of that participant with missing samples to identify its highly-correlated attributes.These attributes are then utilized to train a local generation model, generating those highly-correlated attributes data for the missing samples of that participant.Taking the example of correlated attribute generation for Party B, the detailed process is as follows.

Spearman correlation coefficient [16] was employed to calculate the highly correlated attributes in Party B. For the m-th attribute columns $X_m^B = (x_{1m}^B, ..., x_{im}^B, ..., x_{d_{ID_A \cap ID_B} m}^B)$,and the n-th attribute columns $X_n^B = (x_{1n}^B, ..., x_{in}^B, ..., x_{d_{ID_A \cap ID_B} n}^B)$ of the samples from Party B, iterate over each attribute value x_{in}^B, x_{in}^B in X_m^B, X_n^B, and assign ranking to them, respectively, where $i = 1, ..., d_{ID_A \cap ID_B}$. For each attribute value x_{im}^B and x_{in}^B,their corresponding rankings are $Rank_{im}^B$ and $Rank_{in}^B$ are obtained. Calculate the difference d_i between $Rank_{im}^B$ and $Rank_{in}^B$:

$$d_i = Rank_{im}^B - Rank_{in}^B \tag{1}$$

The Spearman correlation coefficient $\rho_{(X_m^A, X_n^B)}$ is then calculated according to the differences d_i :

$$\rho_{(X_m^A, X_n^B)} = 1 - \frac{6 \sum_{i=1}^{num} {d_i}^2}{num(num^2 - 1)} \tag{2}$$

where, num is the number of samples in the aligned sample set of Party B

The correlation coefficient matrix M^B for all the attributes of Party B is calculated :

$$M^B = \begin{bmatrix} \rho_{(X_1^B, X_1^B)} & \cdots & \rho_{(X_1^B, X_b^B)} \\ \vdots & \ddots & \vdots \\ \rho_{(X_b^B, X_1^B)} & \cdots & \rho_{(X_b^B, X_b^B)} \end{bmatrix} \tag{3}$$

Initialize the set $\mathrm{X}_{\mathrm{Ar}}^B$ of correlated attribute columns to be generated for Party B as empty. Then, iterate through each coefficient in matrix M^B to find the largest correlation coefficient $\mathrm{Max}(\rho_{(X_m^B, X_n^B)})$ and accordingly determine the attribute pair (X_m^B, X_n^B) with strong correlation. If neither X_m^B nor X_n^B from the attribute pair (X_m^B, X_n^B) exists in $\mathrm{X}_{\mathrm{Ar}}^B$, add either X_m^B or X_n^B to $\mathrm{X}_{\mathrm{Ar}}^B$. If either X_m^B nor X_n^B is already in $\mathrm{X}_{\mathrm{Ar}}^B$, add the other attribute in the pair to $\mathrm{X}_{\mathrm{Ar}}^B$ and reset the value of $\rho_{(X_m^A, X_n^B)}$ to 0. Repeat this process until the total number of generated columns of correlated attributes in $\mathrm{X}_{\mathrm{Ar}}^B$ for Party B meets the specified requirement.

$\mathrm{X}_{\mathrm{Ar}}^B$ is used to train a generation model locally in Party B. The trained generation model is then used to generate the correlated attributes for the missing samples of Party B. Generation models that can be used include CTGAN, TableGAN, CTABGAN, VAE, and TabDDPM, among others.

3.2 Vertical federated imputation based on GANs

This paper propose a Vertical Federated imputation Framework based on Generative Adversarial Networks (GANs) to impute the remaining attributes of the missing samples. This framework will use GANs to perform collaborative imputation with multi-party data, while ensuring data security and privacy protection. This vertical federated imputation framework includes the bottom models of each party and the top model, as shown in Fig. 3. In the process of whole vertical federated imputation, we use fully homomorphic encryption, Cheon-Kim-Kim-Song (CKKS) [17], for data encryption operations.

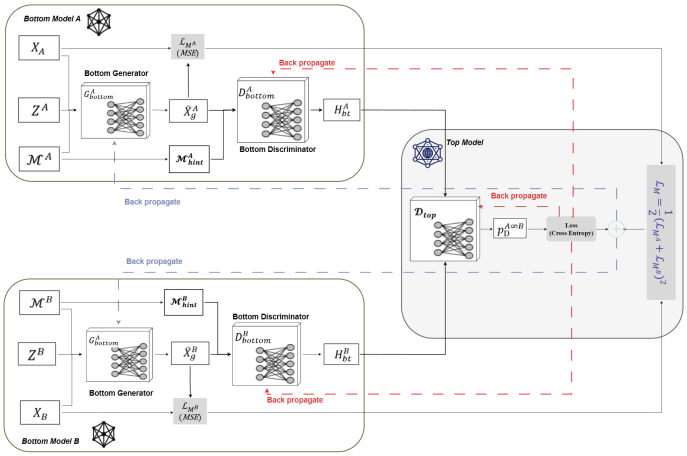

Fig. 3. The framework of vertical federated imputation Based on GANs.

(1) Bottom models In the Bottom model, we design separate generators and discriminators for each party (e.g., Party A and Party B, as shown in Fig. 3). It should be noted that we mention a variety of vertical federation imputation models based on GANs including VF-GAIN, VF-CGAIN and VF-VGAIN, in the experiment in Section 4. These vertical federated imputation models are similar in overall architecture, and the differences are the network structure of their respective bottom models. In other words, the bottom models of each party in VF-GAIN, VF-CGAIN and VF-VGAIN have the same structure as that of GAIN [18], CGAIN [19], VGAIN [20].

Set the encoding vectors of the feature attributes as X_A for Party A's sample and as X_B for Party B's sample. Numerical variables are normalized by using Min-Max normalization and encode categorical variables with One-Hot encoding. Set the mask vector of Party A as $M^A = \left[M_1^A, ..., M_d^A, ..., M_{d'A}^A\right]$, where $d = 1, \ldots, d'_A$, d'_A represents the dimensionality of the sample vector of Party A. And the mask vector of Party B as $M^B = \left[M_1^B, ..., M_d^B, ..., M_{d'B}^B\right]$, where $d = 1, \ldots, d'_B$, d'_B represents the dimensionality of the sample vector of Party B. The mask vector indicates the locations of the remaining attributes in the sample that need to be imputed. The remaining sample data to be imputed for Party A and Party B is referred to as the missing data from Party A and Party B, and is represented by the vectors X_{miss}^A, X_{miss}^B, respectively:

$$X_{miss}^A = \begin{cases} X_d^A, & \text{if } M_d^A = 1 \\ \text{NaN}, & \text{if } M_d^A = 0 \end{cases}, X_{miss}^B = \begin{cases} X_d^B, & \text{if } M_d^B = 1 \\ \text{NaN}, & \text{if } M_d^B = 0 \end{cases} \tag{4}$$

that is,

$$X_{miss}^A \odot M^A = X_A \odot M^A, X_{miss}^B \odot M^B = X_B \odot M^B \tag{5}$$

Where, $\odot$ denotes element-wise multiplication between vectors.

In the federated learning process, the inputs of Party A's bottom generator include: X^A_{miss}, M^A and random noise vector Z^A. And, the inputs of Party B's bottom generator include: X^B_{miss}, M^B and random noise vector Z^B. The outputs of the bottom generators of Parties A and B are:

$$\tilde{X}^A = G_{bottom}\left(X^A_{miss}, M^A, Z^A\right), \tilde{X}^B = G_{bottom}\left(X^B_{miss}, M^B, Z^B\right) \tag{6}$$

The joint sample $\tilde{X}_{A\infty B}$, comprising the sample data $\tilde{X}^A$ generated by Party A and the sample data $\tilde{X}^B$ generated by Party B, is represented as follows:

$$\tilde{X}_{A\infty B} = \tilde{X}^A \oplus_\infty \tilde{X}^B \tag{7}$$

where, because the data of both parties are subject to data security and privacy requirements, $\oplus_\infty$ denotes the concatenation under vertical federated learning, which is not a direct concatenation of the data from Parties A and B. It is only for the convenience of representing the joint sample.

The imputed samples $\tilde{X}^A_g$, $\tilde{X}^B_g$ are obtained from the generated vectors $\tilde{X}A$ of Party A, $\tilde{X}^B$ of Party B, and the observed data from the original missing sample vectors X^A_{miss}, X^B_{miss}, respectively:

$$\tilde{X}^A_g = (1-M^A)\odot\tilde{X}^A + M^A\odot X^A_{miss}, \tilde{X}^B_g = (1-M^B)\odot\tilde{X}^B + M^B\odot X^B_{miss} \tag{8}$$

Under such circumstances, the imputation operation can be seen as being performed based on joint samples consisting of Parties A and B. The imputed joint sample is denoted as $\tilde{X}^{A\infty B}_g$, which consists of data $\tilde{X}^A_g$ from party A and data $\tilde{X}^B_g$ from party B, as shown in Equation (9):

$$\tilde{X}^{A\infty B}_g = \tilde{X}^A_g \oplus_\infty \tilde{X}^B_g \tag{9}$$

$\tilde{X}^A \odot M^A$ in generated $\tilde{X}^A$ from Party A and $\tilde{X}^B \odot M^B$ in generated $\tilde{X}^B$ from Party B will apply in calculating the generator loss function in the federated learning process.

The bottom network of discriminator uses a fully connected neural network structure, which primarily consists of two hidden layers. Each hidden layer employs the LeakyReLU activation function, and dropout is applied to randomly ignore some neurons with a certain probability to reduce model overfitting. In this paper, the negative slope of the LeakyReLU is set to 0.2, and the dropout rate is set to 0.5. In this paper, we introduce the hint mask M_{hint} from GAIN to improve imputation quality by guiding the discriminator, setting the hyperparameter p_{hint}=0.9 to control the masking probability, as referenced from the GAIN model [21].The output of the bottom discriminator D^A_{bottom} in Party A is a vector H^A_{bt} obtained by $\tilde{X}^A_g$ and M^A_{hint} through its network, and the output of the bottom discriminator D^B_{bottom} in Party B is a vector H^B_{bt} obtained by $\tilde{X}^B_g$ and M^B_{hint} through its network:

$$\begin{aligned} H^A_{bt} &= D^A_{bottom}\left(\tilde{X}^A_g | M^A_{hint}\right) = D^A_{bottom}\left(\tilde{X}^A_g \oplus M^A_{hint}\right) \\ H^B_{bt} &= D^B_{bottom}\left(\tilde{X}^B_g | M^B_{hint}\right) = D^B_{bottom}\left(\tilde{X}^B_g \oplus M^A_{hint}\right) \end{aligned} \tag{10}$$

(2) Top model Based on the bottom discriminators of Parties A and B, the quality of the generated and imputed data is further evaluated and optimized by a top discriminator. The top discriminator D_{top} uses a fully connected network to concatenate the output hidden vectors from the bottom discriminators of Parties A and B. It contains two hidden layers, with each layer using the LeakyReLU activation function. In this paper, the negative slope of the LeakyReLU function is set to 0.2, and the Dropout ratio is set to 0.5. The vectors H_{bt}^A and H_{bt}^B output from the bottom discriminators of Parties A and B are input into the top discriminator D_{top} to obtain its output probability vector:

$$p_D^{A\infty B} = D_{top}\left(\tilde{H}_{A\infty B}, M_{hint}^{A\infty B}\right) \tag{11}$$

$\widetilde{H}_{A\infty B} = H_{bt}^A \oplus_\infty H_{bt}^B$, $M_{hint}^{A\infty B} = M_{hint}^B \oplus_\infty M_{hint}^A$. $\tilde{H}_{A\infty B}$ denotes the concatenation of vectors H_{bt}^A and H_{bt}^B under vertical federated learning, obtained from the bottom discriminators of Parties A and B. $M_{hint}^{A\infty B}$ denotes the concatenation of hint mask vectors M_{hint}^A and M_{hint}^B under vertical federated learning from Parties A and B. The output $p_D^{A\infty B}$ is the probability vector of the discriminator distinguish the imputed and real data of elements in the joint samples.

(3) Loss function The loss function of the whole vertical federated imputation consists of both generator loss and discriminator loss. The optimization process follows the adversarial training mechanism of Generative Adversarial Network. This training objective can be formally expressed as:

$$\min_G \max_D \mathcal{L}(D, G) \tag{12}$$

Firstly, we fix the generator and train the discriminator. The minimum batch size k_D is selected in each round from the joint sample set $\tilde{X}_{A\infty B}$. Each sample $\tilde{X}_{A\infty B}(i)$ in the minimum batch includes $\tilde{X}_g^A(i)$ and $M_{hint}^A(i)$ in Party A, $\tilde{X}_g^B(i)$ and $M_{hint}^B(i)$ in Party B. The total loss function of the discriminator is:

$$\min_D -\sum_{i=1}^{k_D} L_{D_{top}}\left(M_{hint}^{A\infty B}(i), \hat{M}^{A\infty B}(i)\right) \tag{13}$$

$$L_{D_{top}} = \sum_{j:m_j^{A\infty B}=0} \left[m_j^{A\infty B} \log\left(\hat{m}_j^{A\infty B}\right) + (1 - m_j^{A\infty B}) \log\left(1 - \hat{m}_j^{A\infty B}\right)\right] \tag{14}$$

$\hat{M}^{A\infty B}(i) = D_{top}(\widetilde{H}_{A\infty B}(i), M_{hint}^{A\infty B}(i))$, $m_j^{A\infty B}$ is the j-th element in the hint mask vector $M_{hint}^{A\infty B}(i)$ for the i-th sample, and $\hat{m}_j^{A\infty B}$ is the j-th element in $\hat{M}^{A\infty B}(i)$ for the i-th sample. It should be noted that, for VF - CGAIN, the category labels need to be added because its bottom model is consistent with that of CGAIN; i.e., $\hat{M}^{A\infty B}(i) = D_{top}(\widetilde{H}_{A\infty B}(i), M_{hint}^{A\infty B}(i)|C(i))$, where, $C(i)$ denotes the label vector of the i-th sample.

Subsequently, we fix the discriminator and train the generator. The minimum batch size k_G is selected in each round from the joint sample set $\tilde{X}_{A\infty B}$. Each

sample $\tilde{X}_{A\infty B}(i)$ in the minimum batch includes $\tilde{X}_g^A(i)$, $\tilde{X}^A(i)$ and $M_{hint}^A(i)$ in Party A, $\tilde{X}_g^B(i)$, $\tilde{X}^B(i)$ and $M_{hint}^B(i)$ in Party B. The total loss function of the generator is:

$$\min_G \sum_{i=1}^{k_G} \mathcal{L}_{GT_{top}}(M_{hint}^{A\infty B}(i), \hat{M}^{A\infty B}(i), \tilde{X}_g^{A\infty B}(i), \tilde{X}_{A\infty B}(i)) \tag{15}$$

$$\begin{aligned} &L_{GT_{top}}\left(M_{hint}^{A\infty B}(i), \hat{M}^{A\infty B}(i), \tilde{X}_g^{A\infty B}(i), \tilde{X}_{A\infty B}(i)\right) \\ &\quad = L_{G_{top}}\left(M_{hint}^{A\infty B}(i), \hat{M}^{A\infty B}(i)\right) + \alpha L_M\left(\tilde{X}_g^{A\infty B}(i), \tilde{X}_{A\infty B}(i)\right) \end{aligned} \tag{16}$$

Where, $\mathcal{L}_{G_{top}} = -\sum_{j:m_j^{A\infty B}=0}[(1-{m_j}^{A\infty B})\log(\widehat{m}_j^{A\infty B})]$, $\mathcal{L}_M = \frac{1}{2}(\mathcal{L}_{M^A} + \mathcal{L}_{M^B})^2$
The loss functions $\mathcal{L}_{M^A}$ in Party A and $\mathcal{L}_{M^B}$ in Party B are:

$$\mathcal{L}_{M^A} = \sum_{j=1}^{d_A} m_j^A \mathcal{L}_M(x_j^A, x_j'^A), \mathcal{L}_{M^B} = \sum_{j=1}^{d_B} m_j^B \mathcal{L}_M(x_j^B, x_j'^B) \tag{17}$$

$$\begin{aligned} \mathcal{L}_M(x_j^A, x_j'^A) &= \begin{cases} (x_j'^A - x_j^A)^2, & \text{if } x_j^A \text{ is continuous,} \\ -x_j^A \log x_j'^A, & \text{if } x_j^A \text{ is binary.} \end{cases} \\ \mathcal{L}_M(x_j^B, x_j'^B) &= \begin{cases} (x_j'^B - x_j^B)^2, & \text{if } x_j^B \text{ is continuous,} \\ -x_j^B \log x_j'^B, & \text{if } x_j^B \text{ is binary.} \end{cases} \end{aligned} \tag{18}$$

Where, i refers to the i-th sample in the joint sample set. j is the j-th element in the i-th sample of Party A or Party B, x_j^A and x_j^B are the observed data, $x_j'^A$ and $x_j'^B$ are the generated data, x_j^A and $x_j'^A$ represent the j-th elements in the i-th samples $\tilde{X}_g^A(i)$ and $\tilde{X}^A(i)$, x_j^B and $x_j'^B$ represent the j-th elements in the i-th samples $\tilde{X}_g^B(i)$ and $\tilde{X}^B(i)$, respectively. The parameter α is a hyperparameter. In the experiments presented in this paper, the value of α is referenced from the GAIN model [18].

4 Experiments

4.1 Datasets and Data Preparation

In our experiments, we utilized four datasets from the UCI repository [22] to evaluate the proposed method: the Bank Marketing Dataset, the German Credit Dataset, the Letter Recognition Dataset and Online News Popularity Dataset: ①The Bank Marketing Dataset pertains to the direct marketing campaigns of a Portuguese banking institution, which contains 45,211 examples and 16 attribute features, along with an ID column and a label column. It aims to classify whether a client will subscribe to a term deposit. ②The German Credit Dataset originates from a credit scoring system, and its objective is to classify applicants as having either "good" or "bad" credit. The dataset includes 1000 samples, and

each sample includes 21 attributes covering financial, demographic, and social characteristics: 13 categorical attribute features, 7 numerical attribute features, and a label column. ③The Letter Recognition dataset is used for character recognition tasks and contains 20,000 instances and 16 attribute features. Each instance includes statistical features of handwritten letter images, such as size, shape, and contour. This dataset aims to classify each sample into one of the 26 English letters. ④The Online News Popularity dataset is used to predict the popularity of online news articles and contains 39,644 instances and 60 attribute features. The dataset includes information such as article content, social media interactions, and publishing time. For simplicity, we refer to the four datasets as 'Bank' , 'Credit', 'Letter' and 'News', respectively.

In the experiments on the 'Bank' and 'Credit' datasets, the attributes are vertically partitioned between Parties A and B based on feature ownership. Specifically, Party A contains customer information, while Party B contains bank information. In the Bank dataset, Party A has 8 attributes while Party B has 8 attributes excluding the ID column excluding the ID column. In the Credit dataset, Party A has 9 attributes and Party B has 11 attributes excluding the ID column. 'Letter' and 'News' datasets are used to simulate a vertical - federated learning scenario. And, we divide the original attribute columns equally between Party A and Party B according to their order.

To verify the validity of the proposed method, we thoroughly conducted sufficient experimental demonstrations for the case where $\mathrm{D}^{\mathrm{B}}_{\mathrm{UnAlign}} = \varnothing$ in Fig. 1. The case when $\mathrm{D}^{\mathrm{B}}_{\mathrm{UnAlign}} = \varnothing$ can be inferred in the same way. In the experiments of this paper, we set different missing sample ratios for Party B when $\mathrm{D}^{\mathrm{B}}_{\mathrm{UnAlign}} = \varnothing$. For example, a missing sample ratio of 0.2 indicates that 80% of the samples in Party B can be aligned with those in Party A, while 20% of the samples are missing in Party B, relative to Party A. In the experiments, Cnum represents the number of correlated attributes generated for the missing samples in Party B. And, we denote the missing sample ratio of Party B relative to Party A as MisR-B.

4.2 Experiment 1: Different settings in FedPSG-CAG

In Experiment 1, the evaluation metric employed to assess the effectiveness of sample generation is RMSE. RMSE [23] is used to indicate the average degree of deviation between the data in the generated sample and the real data. Experiment 1 includes three experimental settings, utilizing the 'Bank' and 'Credit' datasets mentioned in Subsection 4.1. These datasets represent two scenarios: one with a relatively large sample size and the other with a relatively small sample size. In Experiment 1, for the federated or non-federated imputation models based on GANs and the generation models involved in our method of this paper, the total number of training iterations is set to 10,000, the number of epochs is set to 10, the learning rate is set to 0.001, and the optimizer is set to Adam.

Experimental Setting 1: In FedPSG-CAG, we fix the generation model and imputation model, and set different Cnum to verify the final effect of our method. Specifically, TabDDPM is used as the generation model and VF-GAIN is the imputation model. Fig. 4 demonstrates the experimental results for Experimental Setting 1. As shown in Fig. 4, under the different values of MisR-B, when Cnum=4, 5, 6, the RMSE values obtained by FedPSG-CAG perform better. When Cnum=0, the RMSE values obtained by FedPSG-CAG perform worse. In FedPSG-CAG, Cnum=0 indicates that all attributes of the missing samples in Party B are generated using vertical federated imputation models. These findings suggest that adopting this strategy is feasible, which first generate some highly correlated attributes and then impute the remaining attributes by vertical federated imputation models. Moreover, according to the attribute correlation of samples in Party B, it can generate more high correlation attributes for the missing samples, then perform federated imputation, and the RMSE of the generated samples is lower, which helps obtain a sample set that is closer to the real one. But, it is worth noting that the RMSE will also increase as the Cnum of Party B increases as shown in Fig. 4. These results indicate that a larger number of correlated attribute columns generated for Party B is not necessarily better. In fact, some non-highly correlated attributes are more suitable for generation through federated imputation. Federated imputation can fully leverage the data collaboration among multiple participants to improve the accuracy of generated data. Consequently, generating some highly correlated attributes and combining federated imputation to impute the remaining attributes, can help ensure the quality of the generated samples to a certain extent.

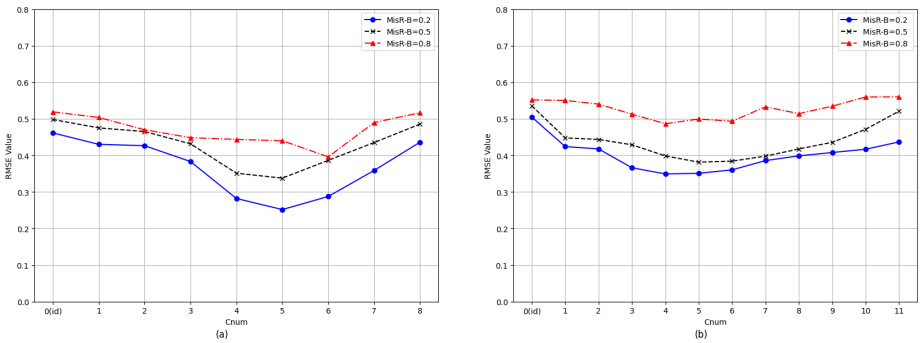

Fig. 4. RMSE line graph obtained by FedPSG-CAG with different Cum of Party B: (a) Bank Dataset; (b) Credit Dataset

Experimental Setting 2: Experimental Setting 2: We fix the Cnum in Party B, and adopt different imputation models to generate the remaining attributes to verify the final effect of our method. Specifically,Cum is set to 5, which is the relatively optimal number of columns in Experimental Setting 1. TabDDPM is still used as the generation model in FedPSG-CAG. The imputation models used for comparison include Mean [24], MissFI (The number of trees in the Random

Forest is set to 100) [25], MICE [26], GAIN [18], CGAIN [19], VGAIN [20].. In the experiment, they are deployed and implemented within a vertical federated framework, which are represented as VF-MissFI, VF-MICE, VF-GAIN, VF-CGAIN, VF-VGAIN. Table 1 demonstrates the experimental results for Experimental Setting 2. Under the different values of MisR-B, the optimal and suboptimal RMSEs are obtained by FedPSG-CAG when the imputation models are VF-GAIN, VF-VGAIN, VF-CGAIN. Particularly, the optimal RMSEs of Bank dataset occur when imputation models are VF-VGAIN and VF-CGAIN, and the optimal RMSEs of Credit dataset occur when imputation model is VF-VGAIN. The results demonstrate the effectiveness of vertical federated imputation models based on GANs. The worst RMSEs of the two datasets occur when imputation model is Mean. Because Mean imputation is a mean calculation on all the values of the current attribute column, and during the imputation process, it neither references nor combines the data from other parties.

Table 1. RMSE obtained by FedPSG-CAG with different imputation models for Bank and Credit Dataset (The optimal and suboptimal results are highlighted in red and blue, respectively.MisR-B=0.2,0,5,0,8)

Dataset	Imp	Mean	VF-MissFI	VF-MICE	VF-GAIN	VF-VGAIN	VF-CGAIN
Bank	0.2	0.4077	0.4036	0.4214	0.2518	0.2372	0.2691
	0.5	0.4744	0.4563	0.4047	0.3378	0.2997	0.2722
	0.8	0.5287	0.4776	0.4520	0.3468	0.3537	0.3832
Credit	0.2	0.4815	0.4481	0.4512	0.3509	0.3495	0.4376
	0.5	0.5006	0.4516	0.4911	0.3815	0.3610	0.4522
	0.8	0.5437	0.4928	0.5210	0.4994	0.4621	0.4657

Experimental Setting 3: Experimental Setting 3: We fix Cnum=5 in Party B, and verify the generation effect of multi-party vertical federated imputation and local non-federated imputation for the remaining attributes. Local non-federated imputation: The remaining attributes in Party B are imputed based on the original data of Party B and the attribute columns generated by Party B. Multi-party vertical federated imputation: The remaining attributes in Party B are imputed based on the original data of Party A and Party B, and the attribute columns generated by Party B. TabDDPM is still used as the generation model in FedPSG-CAG. The imputation models adopt the federated and non-federated implementation of GAIN, CGAIN, VGAIN, which are the outstanding models in Experimental Setting 2. As shown in Table 2, for the different values of MisR-B, the federated imputation methods VF-GAIN, VF-CGAIN, VF-VGAIN overall outperform the local non-federated imputation methods GAIN, CGAIN, VGAIN. Specifically, in FedPSG-CAG, VF-VGAIN performs relatively better among the three federated imputation methods based on GANs. Since federated imputation incorporates multi-party data for collaborative learning, the inclusion of additional data features enhances the quality of the remaining attributes generated for Party B in the joint samples.

Table 2. After generating correlated attributes, RMSE obtained by federated and non-federated imputation based on different GANs when Cnum=5 for Bank and Credit Dataset (The optimal and suboptimal results are highlighted in red and blue, respectively.MisR-B=0.2,0,5,0,8)

Dataset	VF(Y/N)	VF-GAIN	GAIN	VF-VGAIN	VGAIN	VF-CGAIN	CGAIN
Bank	0.2	0.2518	0.3305	0.2372	0.2764	0.2691	0.3531
	0.5	0.3378	0.3524	0.2997	0.3236	0.2722	0.3336
	0.8	0.3468	0.4401	0.3537	0.386	0.3832	0.4018
Credit	0.2	0.3509	0.5095	0.3495	0.4289	0.4376	0.4618
	0.5	0.3815	0.5108	0.3610	0.4573	0.4522	0.5003
	0.8	0.4994	0.5314	0.4621	0.4846	0.4657	0.5059

4.3 Experiment 2: Comparison between FedPSG-CAG and the SOTA baseline models

To validate the effectiveness of our method for participants sample generation, based on Experiment 1, we compare FedPSG-CAG with local generation methods in Party B. The baseline models used for comparison are the current state-of-the-art models in the field of data generation: CTGAN [9], TableGAN [10], CTAB-GAN [11], TVAE [9], and TabDDPM [14]. This experiment is also conducted with varying MisR-B. The evaluation metric for this experiment is RMSE. In FedPSG-CAG, Cum is set to 5, TabDDPM is still used as the generation model, and the federated imputation models are VF-GAIN and VF-VGAIN which performed relatively better in Experiment 1. When Cum= 5, our method is denoted as FedPSG-CAG-5. In this experiment, the total number of training iterations for each generation method is set to 10,000, with 10 epochs, a learning rate of 0.001, and the Adam optimizer.

The first group of experiments is conducted on the two datasets, 'Bank' and 'Credit'. As shown in Table 3, under the different values of MisR-B, the RMSE values of TabDDPM are superior when the generation model is used locally to generate samples for Party B. In the two datasets, when MisR-B is relatively high, FedPSG-CAG-0, i.e., Cum= 0, is superior to methods where the missing samples are generated by local generation models in Party B. However, when MisR-B is relatively low, local generation models in Party B can better learn the distribution of local data for generating the missing samples. However, regardless of whether the MisR-B is high or low, the RMSE of FedPSG-CAG-5 is superior to all the methods in which the missing samples are generated by local generation models in Party B and the method FedPSG-CAG-0. These results further confirm the effectiveness of our method, which first generates highly correlated attributes and then imputes the remaining attributes using vertical federated imputation models.

The second group of experiments is conducted on 'Letter' and'News' datasets, which were mentioned in Section 4.1. Excluding the financial domain, Letter and

News represent different application scenarios as well as different sample sizes and feature numbers.These two datasets equally divide the attribute columns between Parties A and B to simulate the vertical federated scenario. The data correlation, sample size, and number of attribute columns differ from those in 'Bank' and 'Credit' datasets. Therefore, Cnum=5 is not necessarily the optimal setting for FedPSG-CAG. As shown in Table 4, across different values of MisR-B, TabDDPM consistently achieves lower RMSE values compared to other local generation models. However, the RMSE of FedPSG-CAG based on VF-GANs outperforms all models in which the missing samples are generated locally in Party B.

Table 3. RMSE obtained by different methods when generating the missing samples of Party B for Bank and Credit Dataset (The optimal, suboptimal and third-best results are highlighted in red, blue, green, respectively.MisR-B=0.2,0,5,0,8)

Methods \ Dataset	Bank			Credit		
	0.2	0.5	0.8	0.2	0.5	0.8
CTGAN [9]	0.5099	0.5213	0.5456	0.7554	0.6385	0.6600
TableGAN [10]	0.5951	0.6865	0.7312	0.6008	0.6975	0.7838
CTAB-GAN [11]	0.4773	0.5644	0.6454	0.5533	0.6674	0.6976
TVAE [9]	0.4265	0.4862	0.6970	0.4305	0.5534	0.6740
TabDDPM [14]	0.4227	0.4728	0.5236	0.4478	0.5363	0.5750
FedPSG-CAG-0(VF-GAIN)	0.4615	0.4981	0.519	0.5048	0.5354	0.5518
FedPSG-CAG-5(VF-GAIN)	0.2518	0.3378	0.3468	0.3509	0.3815	0.4994
FedPSG-CAG-5(VF-VGAIN)	0.2372	0.2997	0.3537	0.3495	0.3610	0.4621

Table 4. RMSE obtained by different methods when generating the missing samples of Party B for Letter and News Dataset (The optimal, suboptimal and third-best results are highlighted in red, blue, green, respectively.MisR-B=0.2,0,5,0,8)

Methods \ Dataset	Letter			News		
	0.2	0.5	0.8	0.2	0.5	0.8
CTGAN [9]	0.5328	0.5664	0.6218	0.5209	0.5626	0.6034
TableGAN [10]	0.5398	0.6148	0.7068	0.4722	0.5204	0.6234
CTAB-GAN [11]	0.5226	0.561	0.6559	0.4815	0.522	0.6572
TVAE [9]	0.5203	0.5614	0.6634	0.5018	0.5492	0.6348
TabDDPM [14]	0.4921	0.5413	0.5833	0.455	0.5024	0.5671
FedPSG-CAG-5(VF-GAIN)	0.4049	0.4468	0.4897	0.4112	0.463	0.5133
FedPSG-CAG-5(VF-VGAIN)	0.3698	0.3823	0.4768	0.4341	0.4579	0.5021

The excellence of FedPSG-CAG lies in combining correlated attributes generation and vertical federated imputation. Correlated attributes generation effectively captures the intrinsic relationships among the strongly correlated attributes in Party B, ensuring that the data distribution of these generated attributes is more consistent with their original data. In addition, under the framework of vertical federated learning, FedPSG-CAG utilizes the data from multiple

participants, which improves the generation effect while protecting data privacy. To further validate the effectiveness of FedPSG-CAG in addressing the issue of insufficient joint samples after multi-party sample alignment caused by missing samples from some participants, we conducted an additional Experiment 3. In this experiment, joint sample sets are constructed using different methods for handling the missing samples of Party B, which are then used to train vertical federated classification models. The detailed process and analysis can be found in **Supplementary Material A**.

5 Conclusion

In this paper, we propose a novel Participants Sample Generation method based on correlated attributes generation and vertical federated imputation (FedPSG-CAG). FedPSG-CAG first computes and obtains the highly correlated attributes in the set of aligned samples from the participant with missing samples. These attribute columns are used to train the local generation model, generating these correlated attributes of the missing samples for the participants. Then, a vertical federated imputation framework is constructed in this paper to generate the remaining attributes of missing samples. A series of experiments have shown that in the VLF setting, FedPSG-CAG can better address the issue of sample generation for participants with missing samples. Our method can effectively capture the intrinsic relationships among attributes in participants with missing samples and to leverage data associations from multiple participants, thereby improving the generation quality while protecting data privacy. As future work, we would focus on improving the generation of highly correlated attributes.

Acknowledgments. This work was supported by the Chongqing Returned Overseas Scholars Entrepreneurship and Innovation Support Program No. CX2024086; the National Social Science Foundation of China under Grant No. 20BXW097; the Chongqing Technology Innovation and Application Development Special Major Project under Grant No. CSTB2023TIAD-STX0034.

References

1. Liu Y, Kang Y, Zou T, et al. Vertical federated learning: Concepts, advances, and challenges[J]. IEEE Transactions on Knowledge and Data Engineering, 2024, 36(7): 3615-3634.
2. Zhang J F, Jiang Y C. A data augmentation method for vertical federated learning[J]. Wireless Communications and Mobile Computing, 2022, 2022(1): 6596925.
3. Jiang X, Zhang Y, Zhou X, et al. Distributed gan-based privacy-preserving publication of vertically-partitioned data[J]. Proceedings on Privacy Enhancing Technologies, 2023.
4. Yuan X, Yang Y, Gope P, et al. Vflgan: Vertical federated learning-based generative adversarial network for vertically partitioned data publication[J]. arXiv preprint arXiv:2404.09722, 2024.

5. Zhao Z, Wu H, Van Moorsel A, et al. Gtv: Generating tabular data via vertical federated learning[J]. arXiv preprint arXiv:2302.01706, 2023.
6. Zhao F, Li Z, Ren X, et al. VertiMRF: Differentially Private Vertical Federated Data Synthesis[C]//Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. 2024: 4431-4442.
7. Goodfellow I, Pouget-Abadie J, Mirza M, et al. Generative adversarial nets[J]. Advances in neural information processing systems, 2014, 27.
8. Mirza M, Osindero S. Conditional generative adversarial nets[J]. arXiv preprint arXiv:1411.1784, 2014.
9. Xu L, Skoularidou M, Cuesta-Infante A, et al. Modeling tabular data using conditional gan[J]. Advances in neural information processing systems, 2019, 32.
10. Park N, Mohammadi M, Gorde K, et al. Data synthesis based on generative adversarial networks[J]. arXiv preprint arXiv:1806.03384, 2018.
11. Zhao Z, Kunar A, Birke R, et al. Ctab-gan: Effective table data synthesizing[C]//Asian Conference on Machine Learning. PMLR, 2021: 97-112.
12. Bank D, Koenigstein N, Giryes R. Autoencoders[J]. Machine learning for data science handbook: data mining and knowledge discovery handbook, 2023: 353-374.
13. Kingma D P, Welling M. Auto-encoding variational bayes[EB/OL].(2013-12-20)
14. Kotelnikov A, Baranchuk D, Rubachev I, et al. Tabddpm: Modelling tabular data with diffusion models[C]//International Conference on Machine Learning. PMLR, 2023: 17564-17579.
15. De Cristofaro E, Tsudik G. Practical private set intersection protocols with linear complexity[C]//International Conference on Financial Cryptography and Data Security. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010: 143-159.
16. Zar J H. Spearman rank correlation[J]. Encyclopedia of biostatistics, 2005, 7.
17. Cheon J H, Kim A, Kim M, et al. Homomorphic encryption for arithmetic of approximate numbers[J]. Lecture Notes in Computer Science, 2017, 10624: 409-437.
18. Yoon J, Jordon J, Schaar M. Gain: Missing data imputation using generative adversarial nets[C]//International conference on machine learning. PMLR, 2018: 5689-5698.
19. Awan S E, Bennamoun M, Sohel F, et al. Imputation of missing data with class imbalance using conditional generative adversarial networks[J]. Neurocomputing, 2021, 453: 164-171.
20. Miao X, Wu Y, Chen L, et al. An experimental survey of missing data imputation algorithms[J]. IEEE Transactions on Knowledge and Data Engineering, 2022, 35(7): 6630-6650.
21. Zhou X, Liu X, Lan G, et al. Federated conditional generative adversarial nets imputation method for air quality missing data[J]. Knowledge-Based Systems, 2021, 228: 107261.
22. Dua, D., Graff, C.: UCI machine learning repository (2017), http://archive.ics.uci.edu/ml
23. Chai T, Draxler R R. Root mean square error (RMSE) or mean absolute error (MAE)[J]. Geoscientific model development discussions, 2014, 7(1): 1525-1534.
24. Farhangfar A, Kurgan L A, Pedrycz W. A novel framework for imputation of missing values in databases[J]. IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans, 2007, 37(5): 692-709.
25. Stekhoven D J, Bühlmann P. MissForestnon-parametric missing value imputation for mixed-type data[J]. Bioinformatics, 2012, 28(1): 112-118.
26. Royston P, White I R. Multiple imputation by chained equations (MICE): implementation in Stata[J]. Journal of statistical software, 2011, 45: 1-20.