Cybersecurity

RESEARCH

Open Access

IHVFL: a privacy-enhanced intention-hiding vertical federated learning framework for medical data

Fei Tang[1,2*], Shikai Liang[1], Guowei Ling[1] and Jinyong Shan[3]

Abstract

Vertical Federated Learning (VFL) has many applications in the field of smart healthcare with excellent performance. However, current VFL systems usually primarily focus on the privacy protection during model training, while the preparation of training data receives little attention. In real-world applications, like smart healthcare, the process of the training data preparation may involve some participant's intention which could be privacy information for this participant. To protect the privacy of the model training intention, we describe the idea of Intention-Hiding Vertical Federated Learning (IHVFL) and illustrate a framework to achieve this privacy-preserving goal. First, we construct two secure screening protocols to enhance the privacy protection in feature engineering. Second, we implement the work of sample alignment bases on a novel private set intersection protocol. Finally, we use the logistic regression algorithm to demonstrate the process of IHVFL. Experiments show that our model can perform better efficiency (less than 5min) and accuracy (97%) on Breast Cancer medical dataset while maintaining the intention-hiding goal.

Keywords Medical data, Vertical federated learning, Privacy-presserving, Intention-hiding, Logistic regression

Introduction

Driven by the availability of big data, machine learning plays an essential role in the filed of smart healthcare (Garg and Mago 2021; Magoulas and Prentza 1999). There are many related applications such as prediction of disease progression (Huang et al 2019; Brisimi et al 2018), medical image analysis (Li et al 2019; Roth et al 2020), ancillary diagnosis (Qayyum et al 2020), and so on. However, we have to consider following things. For one thing, more data need to be collected to improve the performance of models. For another, medical data such as health records, gene sequences, biometric data, medical image, are very sensitive and private, and it is difficult to gather or transfer between different organizations. What is more, with the increasing awareness of data security and user privacy, the behavior is almost impossible occurred and even forbidden by relevant laws and regulations, such as the General Data Protection Regulation (GDPR).

To solve the problem of "isolated data island", the concept of federated learning is proposed (McMahan et al 2017). Depending on how data are split across parties, the idea was expanded to three categories (Yang et al 2019a): Horizontal Federated Learning (HFL) (Shokri and Shmatikov 2015; Liu et al 2022a; Aono et al 2017), Vertical Federated Learning (VFL) (Abuadbba et al 2020; Chen et al 2021; Liu et al 2022b), and Federated Transfer Learning (FTL) (Liu et al 2020a; Gao et al 2019). As an example, several hospitals have similar patient feature data but few patient samples overlap, then they can perform HFL tasks to obtain a common global medical model by sharing the

*Correspondence:
Fei Tang
tangfei@cqupt.edu.cn
[1] College of Computer Science and Technology, Chongqing University of Posts and Telecommunications, No. 2, Chongwen Road, Nan'an District, Chongqing 400065, China
[2] School of Cyber Security and Information Law, Chongqing University of Posts and Telecommunications, Chongqing 400065, China
[3] Sudo Technology Co., LTD., Beijing 100083, China

same feature space.. Similarly, the VFL situation is when data is vertically split. For instance, hospitals and medical institutions usually have different features space but same samples spaces, they can perform VFL tasks and get a shared model. FTL solutions can be used when data differs not only in samples but also in features space. There have lots of works on FL, most of them focus on HFL. There is a gap in the research on VFL. In this paper, the VFL is the main topic.

VFL has a great future for applications in industries such as finance and healthcare. However, motivated by the rapid growth in VFL research and real-world applications, VFL is dealing with more demands for customized privacy protection. For example, Fig. 1 illustrates the additional security requirements in medical scenarios. To train the federated model for medical applications, the medical company combines the hospital. Both of them contain unique sensitive input data, meanwhile, the medical company, as the requester of VFL, has a sensitive intention for the model training. More concretely, this intention is to find the target data for training safely, which includes target features and target samples. Furthermore, if the privacy of the intention is compromised, it may negatively impact the requester's self-interest and result in the task's failure. A medical company's training program, as an illustration, aims to create a new medicine. If the plan is revealed to competitors, it will inevitably result in the medical company's interests being lost. In addition, in the financial field, for instance, the intention of a financial company's training model is to predict the credit ability of a specific target customer (e.g. one with an annual salary $100,000), if the intention is reveled to the user, it will inevitably result in a loss of trust between the user and the financial company. We can see that the training intention is private information for the requester of VFL and cannot be disclosed to anyone, including the participants. Therefore, the additional privacy issue in VFL models must be addressed.

To the best of our knowledge, existing papers about VFL usually only focus on the privacy protection of model training, which ensure that original data do not compromised. In this paper, we consider an additional privacy requirement, intention-hiding of the training model in VFL systems. For example, there are two medical institutions $\mathcal{C}$ and $\mathcal{S}$, $\mathcal{C}$ is a medical company, while $\mathcal{S}$ is a hospital with a large patient database. Now, $\mathcal{C}$ combines $\mathcal{S}$ to train a model to improve the quality of its products and services. Meanwhile, $\mathcal{C}$ hopes that $\mathcal{S}$ cannot obtain its intention and $\mathcal{S}$ hopes $\mathcal{C}$ cannot obtain its data, since they have their own privacy protection requirements. For $\mathcal{S}$, it need protect the patient data from being compromised. For $\mathcal{C}$, the intention of model training involves its business interests. Therefore, this is a typical VFL scenario with additional requirements. Compared

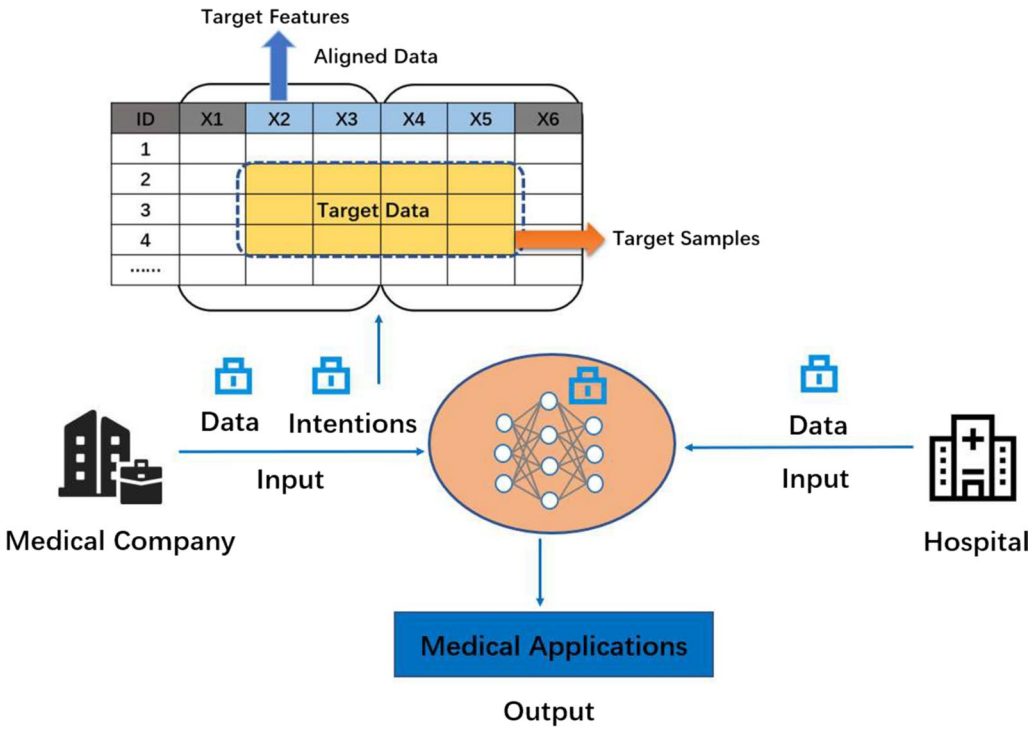


Fig. 1 The privacy-preserving requirement of intention-hiding for medical data scenarios

with the traditional VFL scheme, we not only protect the data during the model training process, but also protect the intention of model training.

To achieve the goal of intention-hiding in medical data applications, we propose the idea of intention-hiding vertical federated learning and construct a framework to meet the privacy-preserving requirements. Our main contributions in this paper are summarized as follows:

- We propose and characterize the idea of intention-hiding vertical federated learning (IHVFL). Compared traditional VFL, it satisfies the need for additional privacy requirement of intention-hiding.
- We enhance the work of privacy-preserving feature engineering and propose a new PSI protocol to implement the sample alignment in VFL.
- We construct the logistic regression algorithm with intention-hiding for two parties to describe the process of IHVFL in details.
- We provide extensive experiments on public medical data to validate the feasibility of our proposed scheme. For example, the results show that our model has better efficiency (less than 5min) and accuracy (97%) on Breast Cancer dataset.

The remainder of this paper is organized as followers. "Introduction" section presents the relevant background of this paper as well as our motivation and contributions. "Related works" section describes the related work on FL for medical data and the directions and concerns in VFL. "Preliminaries" section introduces the preliminaries of our work. "Definitions" section defines the concept of Intention-Hiding VFL and its security and privacy requirements. "IHVFL with Logistic Regression" section demonstrates the details of our proposed framework. "Security analysis" section presents the security proofs for the proposed protocol. "Experiments" section shows the results of the comparison experiment and evaluate the performance of our scheme. Finally, the conclusions of this research and future directions are summarized in "Conclusions" section.

Related works

Federated learning applications for medical data

Federated Learning (FL) has many applications in smart healthcare (Rieke et al 2020). For example, electronic health records (EHR) contain a lot of clinical medical information, and it has great use in medical diagnosis. Huang et al (2019) made use of the EHRs across different hospitals to predict mortality rate for heart disease patients. Brisimi et al (2018) used the cluster Primal Dual Splitting (cPDS) algorithm to predict whether a patient with heart disease will be hospitalized. Moreover, FL has emerged as a promising solution for supporting medical imaging tasks by learning from multi-source datasets without sharing data. Li et al (2019) used the deep neural networks (DNNs) to support brain tumour segmentation, and to protect patient privacy leakage, a differential privacy technique is adopted during the model training. A real-world implementation of FL for medical imaging was presented in Roth et al (2020), they used the FL framework to make breast density classification and the performance is better than the standalone learning approaches.

Directions and concerns in VFL

VFL has the excellent performance in smart healthcare (Sun et al 2019; Brisimi et al 2018). In most works of VFL (Hardy et al 2017; Yang et al 2019a), there is a third party to assist the model training, which is a collaborator role in the VFL systems. However, the centralized VFL suffers from a single point of failure and increases the potential information leakage risk. In addition, it is difficult to find a fair and credible third-party in practice. To handle this problem, decentralized VFL (Yang et al 2019b; Chen et al 2021) was proposed. There is no collaborator involved during model training process.

In order to prevent inference of local data from intermediate results (Zhu et al 2019), most existing works are based on Homomorphic Encryption (HE) to achieve the security goals (Aono et al 2016, 2017). Besides, Secret Sharing (SS) technologies are also popularly used to build the VFL systems (Mohassel and Zhang 2017). Another line of works uses Differential Privacy (DP) (Dwork 2008; Sun et al 2020), which usually involve a trade-off between accuracy and privacy. However, They just focus on the privacy protection of model training, little attention is paid on the other additional privacy protection requirements.

More and more security goals are receiving attention. Recently, considering the asymmetry of participants data, Liu et al (2020b) proposed the concept of asymmetrical vertical federated learning. They divide the participants into 'weak' and 'strong' parties based on the amount of data in the system. They construct an asymmetrical sample alignment protocol to protect the privacy of weak participant. Similarly, Sun et al (2021) started from protecting intersection membership of all parties, proposed a private set union protocol to solve the problem. Instead of identifying the intersection of samples, they take the generated union of samples as training instances. From the perspective of improving data quality, Chen et al (2022) proposed an explainable VFL framework and provided the importance rate as the metric for evaluating the importance of the features. Considering the label privacy in medical scenarios, Fu et al (2022) proposed a new label

attack method and reveal hidden privacy issues in VFL system.

To better understand the current related work, we summarize the above schemes and make a comparison table, and list the addressed challenges in Table 1, respectively. We can see that with the applications of VFL in real-world situations, more and more potential privacy protection concerns are being considered.

Preliminaries

In this section, we describe the setting and threat model of our proposal, and present some background knowledge. All the main notations used in this paper are shown in Table 2.

Vertical federated learning

In the federated learning settings, when the data are distributed vertically, i.e., they share the same sample ID space but differ in feature space, we call it vertical federated learning. Let $\mathcal{D} = (\mathcal{I}, \mathcal{X}, \mathcal{Y})$ denotes a complete dataset with $\mathcal{I}, \mathcal{X}$ and $\mathcal{Y}$, which represent the sample ID space, feature space and label space, respectively (Shokri and Shmatikov 2015). In the classic two-party vertical federated learning scenario, there are two datasets $\mathcal{D}_c = (\mathcal{I}_c, \mathcal{X}_c, \mathcal{Y}_c)$ and $\mathcal{D}_s = (\mathcal{I}_s, \mathcal{X}_s, \mathcal{Y}_s)$, which satisfy $\mathcal{X}_c \neq \mathcal{X}_s, \mathcal{Y}_c \neq \mathcal{Y}_s, \mathcal{I}_c \cap \mathcal{I}_s \neq \varnothing$. We call the party with labels as active party $\mathcal{C}$, and the party without labels as passive party $\mathcal{S}$.

In VFL systems, distributed parties should share the same sample ID space. Therefore, the preparation work is to find the matching sample ID among the parties, and get the same sample space $\mathcal{R} = \mathcal{I}_c \cap \mathcal{I}_s$. This phase of sample alignment is commonly did by Private Set Intersection (PSI) protocol. Next, both parties train the model collaboratively by exchanging the intermediate results for gradient or model. What is more, the intermediate results are masked by encryption, differential privacy or secret sharing techniques. Finally, each party holds a share of model associated to their features.

Table 2 Notations and descriptions

Notations	Descriptions
$\mathcal{C}$	Active party in VFL
$\mathcal{S}$	Passive party in VFL
$\mathcal{F}$	Intention of active party
d	Target features
s	Target samples
σ	Features statement
ρ	Screening vector
τ	Conditional vector
λ	Random scalar
I	The data set with target shares
L_1	The index set with target samples
L_2	The index set with aligned samples
π, π^{-1}	Permutations and inverses of permutations
$[\![x]\!]$	The ciphertext of x
$\langle x \rangle$	The shares of x

Logistic regression

Logistic regression is a classical machine learning algorithm and has been used extensively in medical statistical analysis. The key components of the two parties vertical logistic regression can describe as follows, active party $\mathcal{C}$

Table 1 Comparisons for related work with proposed scheme

References	Description	Addressed challenges
Hardy et al (2017)	This article firstly proposed the FL situation when the data is distributed vertically, and gave a solution based on a collaborator	Privacy protection
Yang et al (2019b)	This article removed the trusted third-party role, proposed a parallel LR model, and improves the scalability of the VFL system	Scalability in VFL
Chen et al (2021)	This article proposed a VFL model CAESAR to solve the problem of high-dimensional sparse data in the field of risk control	Sparse data in specific domains
Chen et al (2022)	According to the complex and impractical problem of data interpretation and evaluation in VFL, this article proposed a explainable VFL framework to evaluate the importance of the features	Explainability and evaluability
Fu et al (2022)	This article revealed the hidden privacy risk in VFL model training and proposed a novel label inference attack method	Label privacy and attacks
Liu et al (2020b)	For the problem of unbalanced distribution of data,This article proposed an asymmetric VFL method to protect the ID privacy of weak parties	ID privacy
Sun et al (2021)	For the intersection membership privacy across privacy-sensitive organizations, this article proposed a VFL framework, allowing each party to preserve private sensitive membership information	Intersection membership privacy
This article	For the privacy protection requirements of hiding model training intention, this article proposed an Intention-Hiding VFL framework to achieve privacy enhancement in VFL	Intention privacy

has $D_c = \{x_i^c, y_i | i \in D_c, y_i \in \{-1, 1\}\}$, passive party $\mathcal{S}$ has $D_s = \{x_i^s | i \in D_s\}$. They aim to learn a model $\mathbf{w}$ by minimizing the loss function:

$$\mathcal{L} = -\frac{1}{n}\sum_{i=1}^{n} y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i) \tag{1}$$

where the $\hat{y}_i = \sigma(\mathbf{X}_i \cdot \mathbf{w}) = \frac{1}{1+e^{-\mathbf{X}_i \cdot \mathbf{w}}}$, and $\sigma(u) = \frac{1}{1+e^{-u}}$ is known as the sigmoid function. In this paper, we use the second order Taylor expansion (Hardy et al 2017) to make the sigmoid function is cryptographically friendly. To efficiently learn the model, the mini-batch SGD algorithm for model trains and updates as follows:

$$\mathbf{w} \leftarrow \mathbf{w} - \frac{\alpha}{|\mathbf{B}|} \cdot \frac{\partial \mathcal{L}}{\partial \mathbf{w}} \tag{2}$$

where $|\mathbf{B}|$ is the batch size, α is the learning rate, and the gradient is denoted as $g = \frac{\partial \mathcal{L}}{\partial \mathbf{w}} = (\hat{Y}_B - Y_B)^T \cdot X_B$.

Private set intersection

Private Set Intersection (PSI) is a preparation work for the VFL, which is to find the public sample intersections. Considering the different application scenarios, There have been many PSI protocols proposed. Interested readers can refer to Meadows (1986), Kolesnikov et al (2016), Debnath and Dutta (2015), Buddhavarapu et al (2020). Here we introduce an intersection-hiding PSI.

Let $R_c = (c_i, x_i^c)$ be the set of tuples of (*identifier*, *values*) associated with active party, and the x_i represents a vector of i-th records. Similarly, $R_s = (s_j, x_j^s)$ be the set of passive party. Let the intersection be $I = \{(x_i^c, x_j^s)\}$ for all i, j where $c_i = s_j$ and the size of intersection is $|I| = k$. The both parties learn the sets $R_{c,I} = \{(r_i^c, r_i^s)\}_{i=1}^k$ and $R_{s,I} = \{(p_i^c, p_i^s)\}_{i=1}^k$ for random values in $[0, 2^\ell)$ where $r_i^c + p_i^c = x_i^c \mod 2^\ell$ and $r_i^s + p_i^s = x_i^s \mod 2^\ell$, for an agreed upon integer ℓ. Finally, the intersections are distributed between the participants as indistinguishable shares, which achieves the purpose of intersection-hiding.

Homomorphic encryption

Homomorphic Encryption (HE) (Paillier 1999; Tang et al 2022) is an encryption scheme that allows computations on ciphertexts and the computation results are matched those of plaintext computations. Due to its significantly greater compute efficiency, additive homomorphic encryption is widely used in the field of federated learning. It mainly has following steps:

- **ParamGen**$(1^\lambda) \rightarrow pp$: λ is a security parameter, and the public parameter pp is implicitly fed in following algorithms.
- **KG**$(pp) \rightarrow (pk, sk)$: Input a public parameter, output a key pair (pk, sk). And pk is public key, while sk is secret key.
- **Enc**$(pk, m) \rightarrow c$: Given a plaintext message m, it is encrypted with pk and generate a ciphertext c.
- **Dec**$(sk, c) \rightarrow m$: Given a ciphertext c, it is decrypted with sk and return a plaintext m.
- **Homomorphic operation**: Given a ciphertext $[\![a]\!]$ and $[\![b]\!]$, the addition is $[\![a]\!] + [\![b]\!] = [\![a + b]\!]$; Given a ciphertext $[\![a]\!]$ and a plaintext b, the multiplication is $[\![a]\!] * \text{b} = [\![a * b]\!]$.

Secret sharing

Secret sharing (SS) is a classic method in Multi-Party Computation (MPC) (Shamir 1979). For example, Alice and Bob want to share a secret. Let the secret x is ℓ-bit, Alice randomly generates an integer $r \in \mathbb{Z}_{2^\ell}$ as $\langle x \rangle_1$, then calculate and send $\langle x \rangle_2 = x - r \mod 2^\ell$ to Bob. At Last, Alice and Bob get the secret shares that meet $\langle x \rangle_1 + \langle x \rangle_2 = x \mod 2^\ell$, respectively. Similarlly, Bob shares a secret y and then Alice gets $\langle y \rangle_1$, Bob gets $\langle y \rangle_2$.

Additive secret sharing (ASS)

ASS is used to compute the result of $x + y$. Assume that Alice has $\langle x \rangle_1, \langle y \rangle_1$ and calculates $\langle z \rangle_1 = \langle x \rangle_1 + \langle y \rangle_1 \mod 2^\ell$. Similarlly, Bob calculate $\langle z \rangle_2 = \langle x \rangle_2 + \langle y \rangle_2 \mod 2^\ell$ and each of them get the shares of results. At last, they exchange their shares and get the result of $x + y = \langle z \rangle_1 + \langle z \rangle_2 \mod 2^\ell$.

Multiplicative secret sharing (MSS)

MSS is used compute the result of $x \cdot y$ by using their shares. The implementation of MSS usually requires the help of Beaver triples (Beaver 1991). A Beaver triple consists of three random numbers a, b, c such that $c = a \cdot b$, and it is private and secure for its owners. Now, Alice and Bob take shared secrets $\langle x \rangle$ and $\langle y \rangle$ as input and get the $z = x \cdot y$ as output. Firstly, they calculate $\langle e \rangle = \langle x \rangle - \langle a \rangle$ and $\langle f \rangle = \langle y \rangle - \langle b \rangle$, respectively. Next, they exchange the shares and reconstruct the e and f. One is mention that since a and b are random and private number, open e and f does not leak information about x and y. Finally, Alice calculates $\langle z \rangle_1 = \langle c \rangle + f \cdot \langle a \rangle + e \cdot \langle b \rangle$, Bob calculates $\langle z \rangle_2 = \langle c \rangle + f \cdot e + f \cdot \langle a \rangle + e \cdot \langle b \rangle$. Finally, they can get the $\langle z \rangle_1 + \langle z \rangle_2 = x \cdot y$.

Additive secret resharing (ASR)

By modifying the protocol so that the results of additive secret sharing can continue to be used for multiplicative secret sharing (Xia et al 2021). In other words, the shared secret over ASS is converted to the shared secret over MSS by additive secret resharing. As mentioned above, Alice and Bob take $\langle x\rangle_1$ and $\langle x\rangle_2$ as input, get the $\langle z\rangle_1$,$\langle z\rangle_2$ as output thus that $\langle z\rangle_1 \cdot \langle z\rangle_2 = x$. First, Alice calculates and sends $e = (\langle x\rangle_1 - \langle c\rangle_1)/a$. Second, Bob calculates $\langle z\rangle_2 = e + b$, $d = (\langle x\rangle_2 - \langle c\rangle_2)/\langle z\rangle_2$, and sends d to Alice. Then Alice calculates $\langle z\rangle_1 = d + a$. Finally, they can get $\langle z\rangle_1$ and $\langle z\rangle_2$, respectively, thus that $\langle z\rangle_1 \cdot \langle z\rangle_2 = \langle x\rangle_1 + \langle x\rangle_2$.

Definitions

In this section, we formally describe the notion of Intention-hiding Vertical Federated Learning.

IHVFL for medical data

Let $\mathcal{C}$, $\mathcal{S}$ represent active and passive party in VFL, respectively. $\mathcal{C}$ combines $\mathcal{S}$ to train a model. For example, the intention of $\mathcal{C}$ is to train a diabetes prediction model on the older population. To do this, $\mathcal{C}$ needs to get the target features about diabetes with the protocol of secure features screening. Meanwhile, $\mathcal{C}$ also needs to obtain the target samples that are older than 60 years old with the protocol of secure samples screening. Next, they use the aligned target data to train the model jointly. In this process, besides the data privacy, additional privacy-preserving requirements are the target features and target samples, which represent the intention of model training.

Formally, let $\mathcal{F} = \{d, s\}$ denote the intention of $\mathcal{C}$, where d represents the features of target data and s represents the samples of target data. For example, in our demo above, d is the features of diabetes model and s is the samples that satisfy the age older than 60. If $\mathcal{C}$ does not leak the $\mathcal{F}$ to $\mathcal{S}$ in the process of VFL, we consider that it has achieved the goal of intention -hiding, and denote it as Intention-Hiding Vertical Federated Learning (IHVFL). We achieve the goal in semi-honest model (Chen et al 2021; Mohassel and Zhang 2017) and illustrate the architecture of IHVFL in Fig. 2.

Security and privacy requirements of IHVFL

In the system of IHVFL, a key point is the stage of data preparation. First, $\mathcal{C}$ performs the privacy-preserving screening protocols on the data set of $\mathcal{S}$ to get the target features of diabetes. In this process, $\mathcal{C}$ cannot get the row feature data of $\mathcal{S}$, and $\mathcal{S}$ cannot know the features that $\mathcal{C}$ selected. Next, $\mathcal{C}$ gets the target samples that meet the condition of person aged older than 60. In this stage, $\mathcal{C}$ cannot get the row sample data of $\mathcal{S}$, and $\mathcal{S}$ cannot know the condition. More specifically, the security and privacy requirements of the IHVFL are constructed via following aspects:

- The features that $\mathcal{C}$ selected in the model training are hiding for $\mathcal{S}$. To do this, a secure features screening

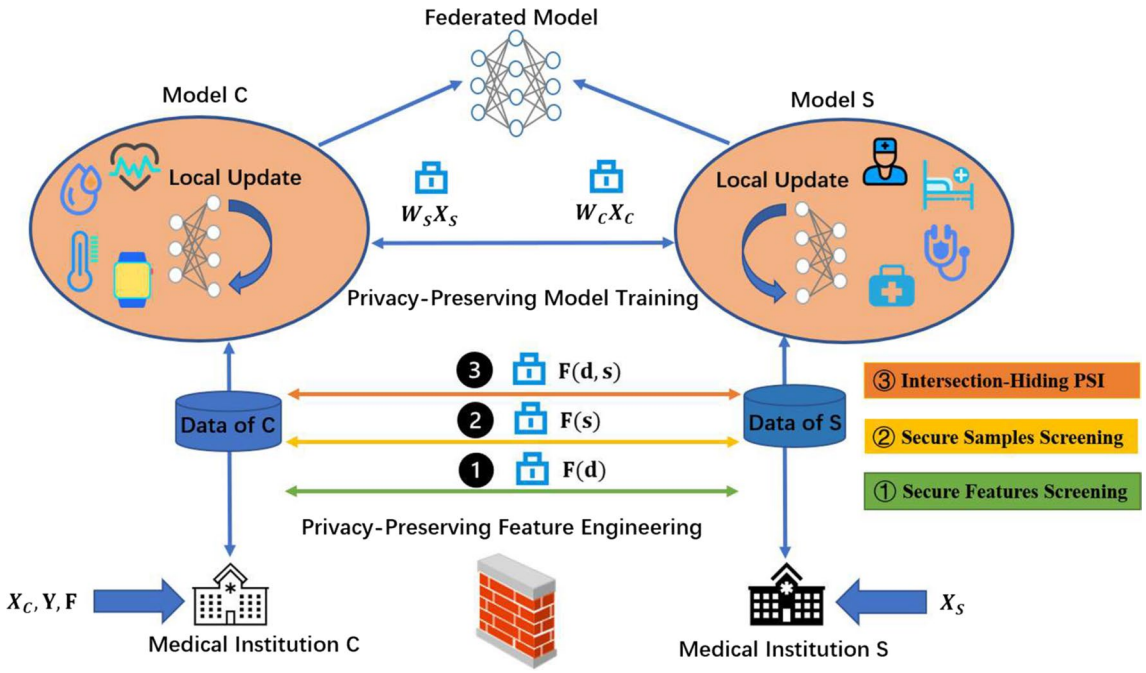


Fig. 2 Architecture of the intention-hiding vertical federated learning for medical data scenarios

protocol is needed to ensure that $\mathcal{C}$ can securely get the target features from $\mathcal{S}$.

- The samples that meet the condition of $\mathcal{C}$ are hiding for $\mathcal{S}$. To do this, a secure samples screening protocol is needed to ensure that $\mathcal{C}$ can securely get the target samples without leaking conditions.

IHVFL with logistic regression

To achieve the intention-hiding in VFL system, in this section, we investigate the privacy-preserving feature engineering and the intersection-hiding PSI protocol in the process of data preparation, and propose a novel and general approach to training the model in IHVFL. First, we construct a secure screening protocol by combing HE and SS to enhance the ability of privacy-preserving in data preparation. Next, we describe the solution of private set intersection with secret shares. Finally, as an example, we chose logistic regression, a classical algorithm widely used in medical data (Caruana et al 2015; Jothi et al 2015), to describe the procedure of intention-hiding federated model training.

Privacy-preserving feature engineering

To get the target data, the passive party $\mathcal{S}$ needs to publish a feature statement to the active party $\mathcal{C}$. Next, $\mathcal{C}$ selects the target data for model training. It is worth mentioning that $\mathcal{C}$ and $\mathcal{S}$ should determine the target features by secure federated feature engineering (Fang et al 2020).

Let $\mathcal{D}_{m\times n}$ be the data of $\mathcal{S}$, m is the number of samples and n is the number of features. $\mathcal{C}$ obtains the target data $\mathcal{D}_{i\times j}$ from $\mathcal{S}$ by the privacy-preserving feature engineering, which includes two stages, features screening and samples screening. In the first, $\mathcal{C}$ gets the shares obtaining the target features. In the second, $\mathcal{C}$ screens the shares with a secret condition, and gets the shares that satisfy the condition. Finally, both parties get the target shares for downstream computation. A key point to note that we construct the protocol based on secret sharing and permutations techniques, it is secure under the DDH problem (Buddhavarapu et al 2020).

Algorithm 1 Secure Features Screening (SFS)

Input: $\mathcal{C}$: $\{\sigma = (\sigma_1, ..., \sigma_m)\}$; $\mathcal{S}$: $\{\mathcal{D} \in \mathbb{R}^{m\times n}\}$
Output: $\mathcal{C}$: $\{I_c\}$; $\mathcal{S}$: $\{I_s\}$ for $I_c \cup I_s = \mathcal{D}_{m,j}$
1: Lets $HE = (\mathbf{KG}, \mathbf{Enc}, \mathbf{Dec})$ be a HE scheme.
2: $\mathcal{S}$ runs $(pk_s, sk_s) \leftarrow$ **KG** (1^λ), for d $\in \mathcal{D}$, calculates and sends $[\![d^c]\!] =$ **Enc** $(pk_s, d^{m\times n})$ to $\mathcal{C}$
3: $\mathcal{C}$ selects j target features from n based on the σ
4: $\mathcal{C}$ generates a random matrix $r^c_{m,j}$ as $\langle d_{m,j}\rangle_1$, gets $I_c = \langle d_{m,j}\rangle_1$ and calculates $[\![d^s]\!] = d^c_{m,j} - r^c_{m,j}$, then shuffles $[\![d^s]\!]$ with a permutation π_c and sends to $\mathcal{S}$
5: $\mathcal{S}$ calculates $\langle d_{m,j}\rangle_2 =$ **Dec** $(sk_s, [\![d^s]\!])$, gets $I_s = \langle d_{m,j}\rangle_2$

Features screening

In order to achieve the purpose of features screening, $\mathcal{C}$ needs to know a statement σ, which is about the feature definition and declaration of the passive party $\mathcal{S}$ in advance. For example, the σ = (age,glucose,bmi,blood pressure,cholesterol), $\mathcal{C}$ selects features about diabetes using the σ, calculates and sends shares to $\mathcal{S}$. We assume that $\mathcal{S}$ has one share $\{\langle d_1\rangle_1, \langle d_2\rangle_1, ... \langle d_j\rangle_1\}$, while $\mathcal{C}$ has another share $\{\langle d_1\rangle_2, \langle d_2\rangle_2, ... \langle d_j\rangle_2\}$. Particularly, to better protect the target features in features screening, data is need to be disrupted by $\mathcal{C}$ with a predefined permutation π. The process of recovering this permutation is denoted as π^{-1}. This step is to ensure that the party being screened cannot distinguish which features are chosen. We describe the process in Algorithm 1.

Algorithm 2 Secure Samples Screening (SSS)

Input: $\mathcal{C}$: $\{\rho = (\rho_1, ..., \rho_j), \tau = (\tau_1, ..., \tau_j), I_c\}$; $\mathcal{S}$: $\{I_s\}$
Output: $\mathcal{C}$: $\{I_c\}$; $\mathcal{S}$: $\{I_s\}$ for $I_c \cup I_s = \mathcal{D}_{i,j}$
1: Let Beaver Triples $\boldsymbol{c} = \mathbf{a} \times \mathbf{b}$, $\mathcal{C}$ has ($\mathbf{a}$, $\boldsymbol{c_1}$), $\mathcal{S}$ has ($\mathbf{b}$, $\boldsymbol{c_2}$)
2: $\mathcal{C}$ shares ρ with $\mathcal{S}$, and $\mathcal{C}$ gets $\langle\rho\rangle_1$, $\mathcal{S}$ gets $\langle\rho\rangle_2$
3: For each $\langle d_{m,j}\rangle_1 \in I_c$, $\mathcal{C}$ calculates $\upsilon_1 = \langle\rho\rangle_1 - \langle d_{m,j}\rangle_1$
4: For each $\langle d_{m,j}\rangle_2 \in I_s$, $\mathcal{S}$ calculates $\upsilon_2 = \langle\rho\rangle_2 - \langle d_{m,j}\rangle_2$
5: $\mathcal{C}$ calculates $e = (\upsilon_1 - c_1)/a$, sends e to $\mathcal{S}$
6: $\mathcal{S}$ calculates $\mu_2 = e + b, d = (\upsilon_2 - c_2)/\mu_2$, sends (μ_2, d) to $\mathcal{C}$
7: $\mathcal{C}$ calculates $\mu_1 = d + a$, and gets $\mu = \mu_1 \cdot \mu_2 = \upsilon_1 + \upsilon_2$
8: For each $i \in m$, $\mathcal{C}$ determines whether $\mu_{i,j}$ satisfies τ
If true, let $I_c = I_c \cup \langle d_{i,j}\rangle_1, L_1 = L_1 \cup i$
If false, let $I_c = I_c \cup \varnothing, L_1 = L_1 \cup \varnothing$
9: $\mathcal{C}$ sends L_1 to $\mathcal{S}$, for each $i \in L_1$, $\mathcal{S}$ calculates $I_s = I_s \cup \langle d_{i,j}\rangle_2$

Samples screening

In some scenarios, we still need to screen the samples. For example, to build a prediction model for diabetes inadults above 60, we need to screen the samples with age older than 60. For this purpose, we designed a secure samples screening protocol based on the ASR. With the shares of features screening, $\mathcal{C}$ takes a screening vector $\rho = \{60, 0, 0, 0, 0, 0\}$, which means that $\mathcal{C}$ wants to screen the first feature of age and the value to be screened is '60'. Meanwhile, $\mathcal{C}$ has a conditional vector $\tau \leftarrow \{-1, 0, 1\}^*$, which means that the feature at the associated index is whether satisfy the condition or not. Such as, '-1' denotes the value < '60', '0' denotes value = '60' and '1' denotes value > '60'. Therefore, $\mathcal{C}$ can get the samples shares of $\mathcal{S}$ that meet the condition by secure comparing. The details of secure samples screening are presented in Algorithm 2.

We describe the execution process of the above two protocols in Fig. 3. We can observe that $\mathcal{C}$ gets the shares that obtained the target features after executing the protocol of features screening. Next, $\mathcal{C}$ gets the shares containing the target samples through a protocol of secure samples screening. Finally, the target data is distributed between two parties in the form of shares.

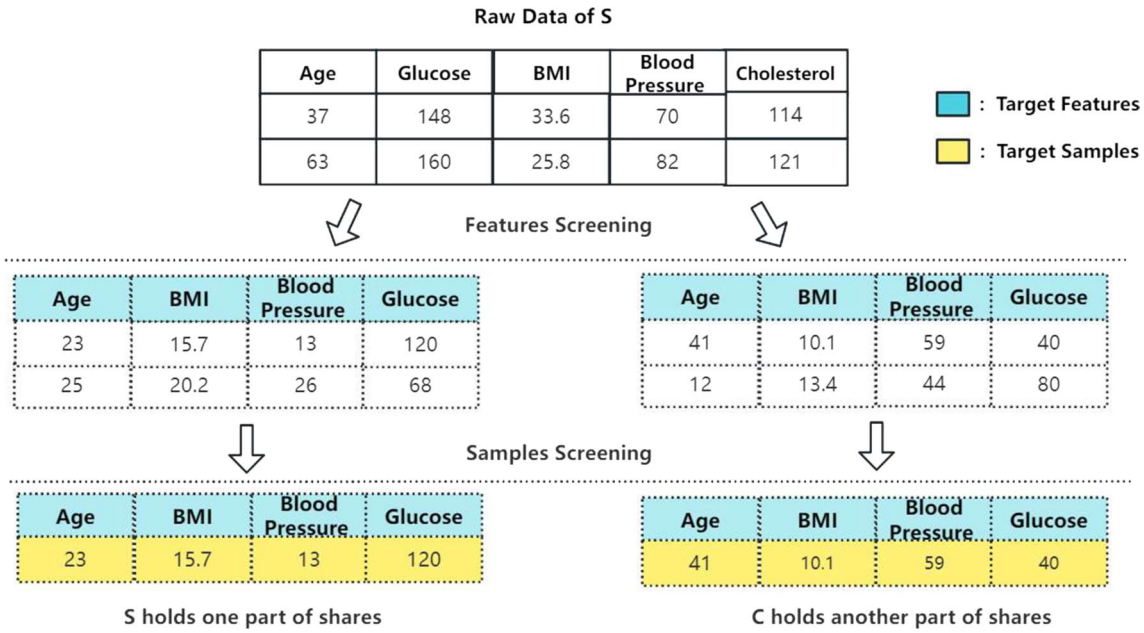


Fig. 3 Process of secure screening protocols

PSI based on secret shares

We construct a PSI protocol based on secret shares. Unlike with the traditional PSI, our approach not only protects samples outside the intersection, but also the intersection ID. That is to say, all participants do not know the specific information about the intersection, such as whether a sample exists in the intersection. This is because the intersection is distributed among the participants in the form of indistinguishable shares.

As mentioned above, after executing the secure screening protocol in "Privacy-preserving feature engineering" section, $\mathcal{C}$ gets the target shares of $\mathcal{S}$. Meanwhile, $\mathcal{S}$ needs to get the target shares of $\mathcal{C}$. Therefore, $\mathcal{C}$ performs the same strategy to screen its features and samples. Next, $\mathcal{C}$ encrypts and sends them to $\mathcal{S}$. At this time, $\mathcal{S}$ calculates the shares, and sends them to $\mathcal{C}$ after a shuffle with π_s. Finally, $\mathcal{C}$ decrypts them and gets another part of shares. It is worth mentioning that this shuffle step is to prevent $\mathcal{C}$ from knowing the sequence of samples, which ensure that $\mathcal{C}$ dose not know the intersection ID in the process of PSI. In fact, during the implementation of the protocol, the samples of both parties are disturbed by the other party, and the restoration is finished prior to the intersection comparison, so as to prevent the other party from inferring additional information according to the sequence of the samples. The process of the PSI based on target shares following next manners:

1. **Calculate and exchange markings:**

   (i) $\mathcal{C}$ computes $H(c_i)^{\lambda_c}$ using a random scalar λ_c for all i and sends them to $\mathcal{S}$

   (ii) $\mathcal{S}$ computes $H(s_m)^{\lambda_s}$ using a random scalar λ_s for all m, then computes $H(c_i)^{\lambda_c \times \lambda_s}$, shuffles with π_s and sends them to $\mathcal{C}$

   (iii) $\mathcal{C}$ selects the target records from $H(s_m)^{\lambda_s}$ with L_1 and π_c^{-1}, gets $H(s_j)^{\lambda_s}$ and computes $H(s_j)^{\lambda_s \times \lambda_c}$

2. **Calculate intersections and output shares:**

   (i) $\mathcal{C}$ determines whether the shares are aligned by $H(s_j)^{\lambda_s \times \lambda_c}$ and $H(c_i)^{\lambda_c \times \lambda_s}$, then calculates $R_{c,I} = R_{c,I} \cup (\langle x_i^c \rangle_1, \langle x_j^s \rangle_1)$ and sends the index (i, j) to $\mathcal{S}$

   (ii) $\mathcal{S}$ finds the corresponding shares by (i, j) and gets $R_{s,I} = R_{s,I} \cup (\langle x_i^c \rangle_2, \langle x_j^s \rangle_2)$

It is important to note that we do not seek to improve PSI performance and we just present a shares-based PSI approach that meets our demands for privacy preservation. Therefore, we construct a PSI protocol that based the DDH with random shuffling (Buddhavarapu et al 2020). A formal description of the protocol shows in Algorithm 3.

Algorithm 3 Intersection-Hiding PSI (IH-PSI)

Input: $[\mathcal{C}: \{c = (c_1, ..., c_I), \langle x^c \rangle_1 = (\langle x_1^c \rangle_1, ..., \langle x_I^c \rangle_1), \langle x^s \rangle_1 = (\langle x_1^s \rangle_1, ..., \langle x_J^s \rangle_1), \pi_c(c_i, \langle x_i^c \rangle_1, \langle x_i^c \rangle_2\}; \mathcal{S}: \{s = (s_1, ..., s_m), \langle x^c \rangle_2 = (\langle x_1^c \rangle_2, ..., \langle x_I^c \rangle_2), \langle x^s \rangle_2 = (\langle x_1^s \rangle_2, ..., \langle x_J^s \rangle_2), \pi_s(s_j, \langle x_j^s \rangle_2, \langle x_j^s \rangle_2)\}]$ for $x_i^c, x_j^s \in [0, 2^\ell)$, a random oracle $H(\cdot): \{0,1\}^* \to \mathbb{G}$

Output: $[\mathcal{C}: R_{c,I} = \{(\langle x^c \rangle_1, \langle x^s \rangle_1)\}^k; \mathcal{S}: R_{s,I} = \{(\langle x^c \rangle_2, \langle x^s \rangle_2)\}^k]$, I is the intersection, and k is the size of intersection

$\mathcal{C}$: **Calculate and Exchange markings**

1: Given $\lambda_c \xleftarrow{R} \mathbb{Z}_q$ and $U_c \leftarrow \emptyset$. For each $c_i \in c$, $\mathcal{C}$ calculates $u_i^c = H(c_i)^{\lambda_c}$ and let $U_c = U_c \cup \{u_i^c\}$, sends U_c to $\mathcal{S}$

$\mathcal{S}$: **Calculate and Exchange markings**

2: Given $\lambda_s \xleftarrow{R} \mathbb{Z}_q$ and $U_s, E_c \leftarrow \emptyset$. For each $s_j \in s$, $\mathcal{S}$ calculates $u_j^s = H(s_j)^{\lambda_s}$ and $U_s = U_s \cup \{u_j{}^s\}$

3: For each $u_i^c \in U_c$, $\mathcal{S}$ calculates $e_i^c = (u_i^c)^{\lambda_s}$ and $E_c = E_c \cup \{e_i^c\}$, then shuffles U_s with π_s and sends U_s, E_c to $\mathcal{C}$

$\mathcal{C}$: **Calculate set intersection and output shares**

4: Let $L, E_s \leftarrow \emptyset$. For each $u_j^s \in U_s$, $\mathcal{C}$ selects the target records u_i^s by L_1 and π_c^{-1}, calculates $e_i^s = (u_i^s)^{\lambda_c}$ and gets $E_s = E_s \cup \{e_i^s\}$

5: For every (i, j) where $e_i^c = e_j^s \in (E_c, E_s)$, $\mathcal{C}$ calculates $L_2 = L_2 \cup (i, j)$ and let $R_{c,I} = R_{c,I} \cup (\langle x_i^c \rangle_1, \langle x_j^s \rangle_1)$

6: $\mathcal{C}$ sends L_2 to $\mathcal{S}$, and outputs shares $R_{c,I}$

$\mathcal{S}$: **Output shares**

7: For each $(i, j) \in L_2$, $\mathcal{S}$ gets $R_{s,I} = R_{s,I} \cup (\langle x_i^c \rangle_2, \langle x_j^s \rangle_2)$ and outputs shares $R_{s,I}$

Intention-hiding vertical logistic regression

The downstream computational procedure can be easily constructed based on the aligned secret shares. Now, we introduce the process of intention-hiding model training using logistic regression.

Secure matrix multiplication overview

As we describe above, matrix multiplication operations plays a key role in logistic regression, so we construct a protocol by combing the homomorphic encryption and secret sharing, which implements the secure matrix multiplication between two parties when data is distributed vertically. Similarly, let $\mathcal{C}$ and $\mathcal{S}$ represent the active party and passive party, respectively. They want to compute the product of matrices $\mathbf{X}$ and $\mathbf{Y}$ securely. First, $\mathcal{C}$ encrypts the $\mathbf{X}$ and sends $[\![\mathbf{X}]\!]$ to $\mathcal{S}$. Next, $\mathcal{S}$ calculates $[\![\mathbf{X}]\!] \cdot \mathbf{Y}$, and shares it in the ciphertext additive operation. Finally, $\mathcal{C}$ decrypts and gets the shares of the product of matrices. More details can see Algorithm 4.

Algorithm 4 Secure Matrix Multiplication (SecMM)

Input: Party $\mathcal{C}$ holds matrix $\mathbf{X}$, Party $\mathcal{S}$ holds matrix $\mathbf{Y}$, HE scheme $HE = (\mathbf{KG}, \mathbf{Enc}, \mathbf{Dec})$ and key pair (pk_c, sk_c) for $\mathcal{C}$

Output: $\mathbf{Z}_1$ for $\mathcal{C}$ and $\mathbf{Z}_2$ for $\mathcal{S}$ thus that $\mathbf{Z}_1 + \mathbf{Z}_2 = \mathbf{X} \cdot \mathbf{Y}$

1: $\mathcal{C}$ calculates $[\![\mathbf{X}]\!] \leftarrow \mathbf{Enc}\ (pk_c, \mathbf{X})$ and sends $[\![\mathbf{X}]\!]$ to $\mathcal{S}$

2: $\mathcal{S}$ lets $\mathbf{Z}_2 \leftarrow \mathbb{Z}_{2^\ell}$, calculates $[\![\mathbf{Z}_1]\!] = [\![\mathbf{X}]\!] \cdot \mathbf{Y} - \mathbf{Z}_2$, and sends $[\![\mathbf{Z}_1]\!]$ to $\mathcal{C}$

3: $\mathcal{C}$ calculates $\mathbf{Z}_1 \leftarrow \mathbf{Dec}\ (sk_c, [\![\mathbf{Z}_1]\!])$

4: **return** $\mathbf{Z}_1$ for $\mathcal{C}$ and $\mathbf{Z}_2$ for $\mathcal{S}$

Algorithm 5 Intention-Hiding Vertical LR (IH-VLR)

Input: shares for party $\mathcal{C}$: $I_c = \{\langle \mathbf{X}_c \rangle_1, \langle \mathbf{X}_s \rangle_1, \langle \mathbf{y} \rangle_1\}$, shares for party $\mathcal{S}$: $I_s = \{\langle \mathbf{X}_c \rangle_2, \langle \mathbf{X}_s \rangle_2, \langle \mathbf{y} \rangle_2\}$

Output: models for party $\mathcal{C}(\mathbf{w}_c)$ and models for party $\mathcal{S}(\mathbf{w}_s)$

1: **Initialization:** $\mathcal{C}$ and $\mathcal{S}$ locally initialize their model, i.e., $\mathbf{w}_c$ and $\mathbf{w}_s$, learning rate α and max iterations T.

2: **Training Model:**

3: **for** each $t \in [1, T]$ **do**

4: **Calculate prediction:**

5: $\mathcal{C}$ calculates $\langle u_c \rangle_1 = \mathbf{w}_c \cdot \langle \mathbf{X}_c \rangle_1$, and securely calculates $\langle u_c \rangle_2 = \mathbf{w}_c \cdot \langle \mathbf{X}_c \rangle_2$ by $SecMM$ with $\mathcal{S}$, and after that $\mathcal{C}$ gets $\langle \langle u_c \rangle_2 \rangle_1$ and $\mathcal{S}$ gets $\langle \langle u_c \rangle_2 \rangle_2$

6: $\mathcal{S}$ calculates $\langle u_s \rangle_2 = \mathbf{w}_s \cdot \langle \mathbf{X}_s \rangle_2$, and securely calculates $\langle u_s \rangle_1 = \mathbf{w}_s \cdot \langle \mathbf{X}_s \rangle_1$ by $SecMM$ with $\mathcal{C}$, and after that $\mathcal{C}$ gets $\langle \langle u_s \rangle_1 \rangle_1$ and $\mathcal{S}$ gets $\langle \langle u_s \rangle_1 \rangle_2$

7: $\mathcal{C}$ calculates $\langle u \rangle_1 = \langle u_c \rangle_1 + \langle \langle u_c \rangle_2 \rangle_1 + \langle \langle u_s \rangle_1 \rangle_1$

8: $\mathcal{S}$ calculates $\langle u \rangle_2 = \langle u_s \rangle_2 + \langle \langle u_c \rangle_2 \rangle_2 + \langle \langle u_s \rangle_1 \rangle_2$

9: **Calculate shared error:**

10: $\mathcal{C}$ gets $\langle \mathbf{e} \rangle_1 = \langle u \rangle_1 - 2 \cdot \langle \mathbf{y} \rangle_1$, $\mathcal{S}$ gets $\langle \mathbf{e} \rangle_2 = \langle u \rangle_2 - 2 \cdot \langle \mathbf{y} \rangle_2$

11: **Calculate gradients:**

12: $\mathcal{C}$ calculates $\langle \langle \mathbf{g}_c \rangle_1 \rangle_1 = \langle \mathbf{e} \rangle_1^T \cdot \langle \mathbf{X}_c \rangle_1$ and securely calculates $\langle \langle \mathbf{g}_c \rangle_1 \rangle_2 = \langle \mathbf{e} \rangle_1^T \cdot \langle \mathbf{X}_c \rangle_2$ by $SecMM$ with $\mathcal{S}$, and after that $\mathcal{C}$ gets $\langle \langle \langle \mathbf{g}_c \rangle_1 \rangle_2 \rangle_1$, $\mathcal{S}$ gets $\langle \langle \langle \mathbf{g}_c \rangle_1 \rangle_2 \rangle_2$

13: $\mathcal{S}$ calculates $\langle \langle \mathbf{g}_c \rangle_2 \rangle_2 = \langle \mathbf{e} \rangle_2^T \cdot \langle \mathbf{X}_c \rangle_2$ and securely calculates $\langle \langle \mathbf{g}_c \rangle_2 \rangle_1 = \langle \mathbf{e} \rangle_2^T \cdot \langle \mathbf{X}_c \rangle_1$ by $SecMM$ with $\mathcal{C}$, and after that $\mathcal{C}$ gets $\langle \langle \langle \mathbf{g}_c \rangle_2 \rangle_1 \rangle_1$, $\mathcal{S}$ gets $\langle \langle \langle \mathbf{g}_c \rangle_2 \rangle_1 \rangle_2$

14: $\mathcal{C}$ gets $\langle \mathbf{g}_c \rangle_1 = \langle \langle \mathbf{g}_c \rangle_1 \rangle_1 + \langle \langle \langle \mathbf{g}_c \rangle_1 \rangle_2 \rangle_1 + \langle \langle \langle \mathbf{g}_c \rangle_2 \rangle_1 \rangle_1$

15: $\mathcal{S}$ gets $\langle \mathbf{g}_c \rangle_2 = \langle \langle \mathbf{g}_c \rangle_2 \rangle_2 + \langle \langle \langle \mathbf{g}_c \rangle_1 \rangle_2 \rangle_2 + \langle \langle \langle \mathbf{g}_c \rangle_2 \rangle_1 \rangle_2$

16: Similarly, $\mathcal{C}$ and $\mathcal{S}$ get $\langle \mathbf{g}_s \rangle_1$,$\langle \mathbf{g}_s \rangle_2$, exchange the shared gradients and calculate $\mathbf{g}_c = \langle \mathbf{g}_c \rangle_1 + \langle \mathbf{g}_c \rangle_2$, $\mathbf{g}_s = \langle \mathbf{g}_s \rangle_1 + \langle \mathbf{g}_s \rangle_2$, respectively

17: **Update model:**

18: $\mathcal{C}$ updates $\mathbf{w}_c \leftarrow \mathbf{w}_c - \alpha \cdot \mathbf{g}_c$

19: $\mathcal{S}$ updates $\mathbf{w}_s \leftarrow \mathbf{w}_s - \alpha \cdot \mathbf{g}_s$

20: **end for**

21: **return** models for $\mathcal{C}(\mathbf{w}_c)$ and models for $\mathcal{S}(\mathbf{w}_s)$

Logistic regression with shares

We now introduce the procedure of intention-hiding vertical logistic regression. First, $\mathcal{C}$ and $\mathcal{S}$ input their shares that generated from the secure screening protocol and the initialized models. Next, they collaboratively calculate the shares of prediction with the *SecMM* protocol. After they get the shares of error, they can calculate the shares of gradients to finish model updating. This procedure is described in Algorithm 5. Finally, they can obtain their particular model, respectively. It is important to note that although they can get access to their own gradients and models during each iteration, they do not have additional access to each other's private information since that the data and labels are shared consistently.

Security analysis

In this section, the security of the proposed IHVFL framework will be analyzed and proved in detail.

Security definition

We use simulation-based definitions of security for secure two-party computation to prove that the protocol is secure against a semi-honest adversary. Let $\mathcal{F}$ be the functionality computed by the two-party protocol Π, P_i and x_i represents the party and party's input, where $i \in (1, 2)$. The view of P_i's consists of its input, randomness r and the exchanged messages throughout the protocol Π, which is denoted as $VIEW^{\Pi}_{P_i}$.

Definition 1 Protocol Π securely computes function $\mathcal{F}$ against a semi-honest adversary if there exists two probabilistic polynomial-time (PPT) simulators SIM_1 and SIM_2, such that

$$SIM^{\Pi}_{P_i}(1^{\mathcal{K}}, x_i, \mathcal{F}(x_1, x_2)) \cong VIEW^{\Pi}_{P_i}(x_1, x_2, \mathcal{K}) \tag{3}$$

where $\mathcal{K}$ is security parameter, and $\cong$ denotes computationally indistinguishablity.

We prove the above equations for a semi-honest $\mathcal{C}$ and a semi-honest $\mathcal{S}$, respectively.

Security analysis of SFS

In Algorithm 1, we can see that the messages obtained by $\mathcal{C}$ include the ciphertext $[\![d^c]\!]$ and the output I_c, the messages obtained by $\mathcal{S}$ only include the output I_s. For $\mathcal{C}$, its private input do not leave local. For $\mathcal{S}$, its private input is protected by HE technology, as long as the private key is not disclosed, the private input is safe. Formally, we have the following theorem.

Theorem 1 (Security of Π^{SFS} against a semi-honest $\mathcal{C}$). *Assume that additive HE scheme is indistinguishable under chosen-plaintext attacks. Then the protocol of SFS is secure in Definition* 1.

Proof We Construct a PPT simulator $SIM_{\mathcal{C}}$ to simulate the view of $\mathcal{C}$ in the protocol execution. For the functionality $\mathcal{F}_{SFS}$, $VIEW^{\Pi}_{\mathcal{C}}(\sigma, \mathcal{D}, \mathcal{K}, I_c, I_s)$ consists of $\mathcal{C}$'s input σ, randomness r_c, the obtained ciphertext $[\![d^c]\!]$ and I_c.

Given $\mathcal{K}$, σ, I_c, $SIM_{\mathcal{C}}$ generates a simulation of $VIEW^{\Pi}_{\mathcal{C}}(\sigma, \mathcal{D}, \mathcal{K}, I_c, I_s)$ as follows. It randomly select a matrix $d^{c\prime}$, encrypts it with pk_s and obtain $[\![d^c]\!]'$. Then, it generates $(\sigma, I_c, r_c, [\![d^c]\!]')$ as the output. Therefore, we can get the following two equations:

$$VIEW^{\Pi}_{\mathcal{C}}(\sigma, \mathcal{D}, \mathcal{K}, I_c, I_s) = (\sigma, I_c, r_c, [\![d^c]\!]) \tag{4}$$

$$SIM_{\mathcal{C}}(1^{\mathcal{K}}, \sigma, I_c) = (\sigma, I_c, r_c, [\![d^c]\!]') \tag{5}$$

It is observed that both $[\![d^c]\!]$ and $[\![d^c]\!]'$ serve as the ciphertext for d^c, and they appear indistinguishable to $\mathcal{C}$. Consequently, the probability distributions of $\mathcal{C}$'s view and $SIM_{\mathcal{C}}$'s output are identical. Hence, we claim that Eq. (3) holds. This completes the proof of security of Π^{SFS} in case of a semi-honest $\mathcal{C}$. □

Theorem 2 (Security of Π^{SFS} against a semi-honest $\mathcal{S}$). *Assume that additive HE scheme is indistinguishable under chosen-plaintext attacks. Then the protocol of SFS is secure in Definition* 1.

Proof We Construct a PPT simulator $SIM_{\mathcal{S}}$ to simulate the view of $\mathcal{S}$ in the protocol execution. For the functionality $\mathcal{F}_{SFS}$, $VIEW^{\Pi}_{\mathcal{S}}(\sigma, \mathcal{D}, \mathcal{K}, I_c, I_s)$ consists of $\mathcal{S}$'s input $\mathcal{D}$, randomness r_s, the obtained ciphertext $[\![d^s]\!]$ and I_s.

Given $\mathcal{K}$, $\mathcal{D}$, I_s, $SIM_{\mathcal{S}}$ generates a simulation of $VIEW^{\Pi}_{\mathcal{S}}(\sigma, \mathcal{D}, \mathcal{K}, I_c, I_s)$ as follows. It encrypts $\mathcal{D}$ with pk_s, shuffle them randomly and obtain $[\![d^s]\!]'$. Then, it generates $(\mathcal{D}, I_s, r_s, [\![d^s]\!]')$ as the output. Therefore, we can get the following two equations:

$$VIEW^{\Pi}_{\mathcal{C}}(\sigma, \mathcal{D}, \mathcal{K}, I_c, I_s) = (\mathcal{D}, I_s, r_s, [\![d^s]\!]) \tag{6}$$

$$SIM_{\mathcal{C}}(1^{\mathcal{K}}, \mathcal{D}, I_s) = (\mathcal{D}, I_s, r_s, [\![d^s]\!]') \tag{7}$$

It is observed that both $[\![d^s]\!]$ and $[\![d^s]\!]'$ serve as the ciphertext for d^s, and they appear indistinguishable to $\mathcal{S}$. Consequently, the probability distributions of $\mathcal{S}$'s view and $SIM_{\mathcal{S}}$'s output are identical. Hence, we claim that Eq. (3)

holds. This completes the proof of security of Π^{SFS} in case of a semi-honest $\mathcal{S}$. □

Security analysis of SSS

In Algorithm 2, we can see that the messages obtained by $\mathcal{C}$ include d and μ_2. For $\mathcal{C}$, its local calculation data includes ρ_1 and υ_1, as long as the triples held by $\mathcal{S}$ are not leaked, then d and μ_2 held by $\mathcal{C}$ are indistinguishable random values. For $\mathcal{S}$, the messages it obtained include ρ_2, e and L_1. Similarly, as long as the triples held by $\mathcal{C}$ are not leaked, then the messages held by $\mathcal{S}$ are indistinguishable. It is worth mentioning that since both $\mathcal{C}$ and $\mathcal{S}$ hold L_1, they will both know the size of the target samples set, but cannot determine whether a certain sample is in the set. Formally, we have the following theorem.

Theorem 3 (Security of Π^{SSS} against a semi-honest $\mathcal{C}$). *Assume that the triples are random and secure. Then the protocol of SSS is secure in Definition* 1.

Proof We Construct a PPT simulator $SIM_{\mathcal{C}}$ to simulate the view of $\mathcal{C}$ in the protocol execution. For the functionality $\mathcal{F}_{SSS}$, $VIEW_{\mathcal{C}}^{\Pi}(\rho, \tau, I_c, I_s)$ consists of $\mathcal{C}$'s input ρ, τ, randomness r_c, the obtained value d, μ_2 and output $I_c{}'$.
Given $\mathcal{K}$, ρ, τ, I_c and $I_c{}'$, $SIM_{\mathcal{C}}$ generates a simulation of $VIEW_{\mathcal{C}}^{\Pi}(\rho, \tau, I_c, I_s)$ as follows. It randomly select a value d', μ_2', and generates $(\rho, \tau, I_c, I_c{}', r_c, d', \mu_2')$ as the output. Therefore, we can get the following two equations:

$$VIEW_{\mathcal{C}}^{\Pi}(\rho, \tau, I_c, I_s) = (\rho, \tau, I_c, I_c{}', r_c, d, \mu_2) \tag{8}$$

$$SIM_{\mathcal{C}}(1^{\mathcal{K}}, \rho, \tau, I_c, I_c{}') = (\rho, \tau, I_c, I_c{}', r_c, d', \mu_2') \tag{9}$$

It is observed that d, d' and μ_2, μ_2' are indistinguishable random values to $\mathcal{C}$. Consequently, the probability distributions of $\mathcal{C}$'s view and $SIM_{\mathcal{C}}$'s output are identical. Hence, we claim that Eq. (3) holds. This completes the proof of security of Π^{SSS} in case of a semi-honest $\mathcal{C}$. □

Theorem 4 (Security of Π^{SSS} against a semi-honest $\mathcal{S}$). *Assume that the triples are random and secure. Then the protocol of SSS is secure in Definition* 1.

Proof We Construct a PPT simulator $SIM_{\mathcal{S}}$ to simulate the view of $\mathcal{S}$ in the protocol execution. For the functionality $\mathcal{F}_{SSS}$, $VIEW_{\mathcal{S}}^{\Pi}(\rho, \tau, I_c, I_s)$ consists of $\mathcal{S}$'s input I_s, randomness r_s, the obtained value ρ_2, e, L_1 and output $I_s{}'$.
Given $\mathcal{K}$, I_s and $I_s{}'$, $SIM_{\mathcal{S}}$ generates a simulation of $VIEW_{\mathcal{S}}^{\Pi}(\rho, \tau, I_c, I_s)$ as follows. It randomly select a value $\rho_2{}'$, e', L_1' and generates $(I_s, I_s{}', \rho_2{}', e', L_1')$ as the output. Therefore, we can get the following two equations:

$$VIEW_{\mathcal{S}}^{\Pi}(\rho, \tau, I_c, I_s) = (I_s, I_s{}', r_s, \rho_2{}', e', L_1') \tag{10}$$

$$SIM_{\mathcal{S}}(1^{\mathcal{K}}, I_s, I_s{}') = (I_s, I_s{}', r_s, \rho_2{}', e', L_1') \tag{11}$$

It is observed that ρ_2, e, L_1 and $\rho_2{}'$, e', L_1' are indistinguishable random values to $\mathcal{S}$. Consequently, the probability distributions of $\mathcal{S}$'s view and $SIM_{\mathcal{S}}$'s output are identical. Hence, we claim that Eq. (3) holds. This completes the proof of security of Π^{SSS} in case of a semi-honest $\mathcal{S}$. □

Security analysis of IH-PSI

In Algorithm 3, we can see that the messages obtained by $\mathcal{C}$ include U_s and E_c, its input includes c, $\langle x^c \rangle_1$, $\langle x^s \rangle_1$, π_c and its output is $R_{c,I}$. Therefore, we have

$$VIEW_{C}^{\Pi} = (c, \langle x^c \rangle_1, \langle x^s \rangle_1, \pi_c, R_{c,I}, E_c, U_s) \tag{12}$$

We can construct a PPT simulator $SIM_{\mathcal{C}}$ to simulate the view of $\mathcal{C}$ in the protocol execution, and it generates a simulation of $VIEW_{C}^{\Pi}$ as follows. First, it generates λ_c honestly. Next, for each $i \in [1, I]$, $SIM_{\mathcal{C}}$ randomly choose $g_i \leftarrow \mathbb{G}$ and let $E_c{}' = E_c{}' \cup \{g_i^{\lambda_c}\}$; for each $j \in [1, J]$, $SIM_{\mathcal{C}}$ randomly choose index j and let $U_s' = U_s' \cup \{g_j\}$. Finally, we have

$$SIM_{\mathcal{C}} = (c, \langle x^c \rangle_1, \langle x^s \rangle_1, \pi_c, R_{c,I}, E_c{}', U_s{}') \tag{13}$$

From $VIEW_{C}^{\Pi}$ and $SIM_{\mathcal{C}}$, we need to discuss that $(E_c, E_c{}')$ and $(U_s, U_s{}')$ are indistinguishable to $\mathcal{C}$. Formally, we have the following theorem.

Theorem 5 (Security of Π^{IH-PSI} against a semi-honest $\mathcal{C}$). *Assume that the DDH problem is hard, then the protocol of IH-PSI is secure in Definition* 1.

Proof Using a sequence of hybrid arguments, we show that the distribution generated by $SIM_{\mathcal{C}}$ is indeed indistinguishable from the $VIEW_{\mathcal{C}}$.
$\mathcal{H}_0$: This is the view of $\mathcal{C}$ in the real execution of Π^{IH-PSI}.

$\mathcal{H}_{1,0}$: Identical to $\mathcal{H}_0$.

$\mathcal{H}_{1,i}$: For $i \in [1, I]$, the same as $\mathcal{H}_{1,i-1}$ except that we replace $H(c_i)^{\lambda_c \times \lambda_s}$ in E_c with $g_i^{\lambda_c}$, and g_i is randomly selected element in $\mathbb{G}$.

$\mathcal{H}_{2,0}$: Identical to $\mathcal{H}_{1,I}$.

$\mathcal{H}_{2,j}$: For $j \in [1, J]$, the same as $\mathcal{H}_{2,j-1}$ except that we replace $H(s_j)^{\lambda_s}$ in U_s with a randomly selected element (e.g. g_j) in $\mathbb{G}$.

$\mathcal{H}_3$: The view of $\mathcal{C}$ output by SIM_C^{IH-PSI}.

To begin with, we argue that $\mathcal{H}_{1,i-1}$ and $\mathcal{H}_{1,i}$ are indistinguishable to $\mathcal{C}$. For any PPT adversary $\mathcal{A}$ who can distinguish the two hybrids, we devise a challenger $\mathcal{B}$ can solve the DDH hard problem. $\mathcal{B}$ is given (g, g^a, g^b, g^c) and needs to decide whether c is random or $c = ab$. First, given input c_i, $\mathcal{B}$ can program $H(\cdot)$ and return g^b. We let $g^a = g^{\lambda_c}$, for the challenge markings m, $\mathcal{B}$ cannot reply m belongs to $\mathcal{H}_{1,i-1}$ or $\mathcal{H}_{1,i}$, and send m to $\mathcal{A}$. For $\mathcal{A}$, the markings $m = g^{c\lambda_s}$ if $c = ab$, otherwise markings $m = g_i^{\lambda_c}$ (since g_i is uniformly random). Therefore, if $\mathcal{A}$ judges m belongs to $\mathcal{H}_{1,i-1}$, then $\mathcal{B}$ outputs $c = ab$; if $\mathcal{A}$ judges m belongs to $\mathcal{H}_{1,i}$, then $\mathcal{B}$ outputs c is random. That is to say, if $\mathcal{A}$ can distinguish which hybrids are markings, then $\mathcal{B}$ can solve the DDH problem with the same probability.

Next, we argue that $\mathcal{H}_{2,j-1}$ and $\mathcal{H}_{2,j}$ are indistinguishable to $\mathcal{C}$. For any PPT adversary $\mathcal{A}$ who can distinguish the two hybrids, we devise a challenger $\mathcal{B}$ can solve the DDH hard problem. $\mathcal{B}$ is given (g, g^a, g^b, g^c) and needs to decide whether c is random or $c = ab$. First, given input s_j, $\mathcal{B}$ can program $H(\cdot)$ and return g^b. We let $g^a = g^{\lambda_s}$, for the challenge markings m, $\mathcal{B}$ cannot reply m belongs to $\mathcal{H}_{2,j-1}$ or $\mathcal{H}_{2,j}$, and send m to $\mathcal{A}$. For $\mathcal{A}$, the markings $m = g^c$ when $c = ab$, otherwise $m = g_j$ (since g_j is uniformly random). Therefore, if $\mathcal{A}$ judges m belongs to $\mathcal{H}_{2,j-1}$, then $\mathcal{B}$ outputs $c = ab$; if $\mathcal{A}$ judges m belongs to $\mathcal{H}_{2,j}$, then $\mathcal{B}$ outputs c is random. That is to say, if $\mathcal{A}$ can distinguish which hybrids are markings, then $\mathcal{B}$ can solve the DDH problem with the same probability.

This completes the proof of security of Π^{IH-PSI} against a semi-honest $\mathcal{C}$. □

Similarly, for $\mathcal{S}$, its obtained messages include U_c and L_2, its input includes $s, \langle x^c \rangle_2, \langle x^s \rangle_2, \pi_s$ and its output is $R_{s,I}$. Therefore, we have

$$VIEW_S^{\Pi} = (s, \langle x^c \rangle_2, \langle x^s \rangle_2, \pi_s, R_{s,I}, U_c, L_2) \tag{14}$$

we can construct a PPT simulator $SIM_{\mathcal{S}}$ to simulate the view of $\mathcal{S}$ in the protocol execution, and it generates a simulation of $VIEW_S^{\Pi}$ as follows. First, it generates λ_s honestly. Next, for each $i \in [1, I]$, $SIM_{\mathcal{S}}$ randomly choose $g_i \leftarrow \mathbb{G}$ and let $U_c{}' = U_c{}' \cup \{g_i\}$. Meanwhile, it generates L_2 using the index of $R_{s,I}$. Finally, we have

$$SIM_{\mathcal{S}} = (s, \langle x^c \rangle_2, \langle x^s \rangle_2, \pi_s, R_{s,I}, U_c{}', L_2) \tag{15}$$

From $VIEW_S^{\Pi}$ and $SIM_{\mathcal{S}}$, we need to discuss that $(U_c, U_c{}')$ is indistinguishable to $\mathcal{S}$. Formally, we have the following theorem.

Theorem 6 (Security of Π^{IH-PSI} against a semi-honest $\mathcal{S}$). *Assume that the DDH problem is hard, then the protocol of IH-PSI is secure in Definition* 1.

Proof Using a sequence of hybrid arguments, we show that the distribution generated by $SIM_{\mathcal{S}}$ is indeed indistinguishable from the $VIEW_{\mathcal{S}}$.
$\mathcal{H}_0$: This is the view of $\mathcal{S}$ in the real execution of Π^{IH-PSI}.

$\mathcal{H}_{1,0}$: Identical to $\mathcal{H}_0$.

$\mathcal{H}_{1,i}$: For $i \in [1, I]$, the same as $\mathcal{H}_{1,i-1}$ except that we replace $H(c_i)^{\lambda_c}$ in U_c with a randomly selected element (e.g. g_i) in $\mathbb{G}$.

$\mathcal{H}_2$: The view of $\mathcal{S}$ output by SIM_S^{IH-PSI}.

We argue that $\mathcal{H}_{1,i-1}$ and $\mathcal{H}_{1,i}$ are indistinguishable to $\mathcal{S}$. For any PPT adversary $\mathcal{A}$ who can distinguish the two hybrids, we devise a challenger $\mathcal{B}$ can solve the DDH hard problem. $\mathcal{B}$ is given (g, g^a, g^b, g^c) and needs to decide whether c is random or $c = ab$. First, given input c_i, $\mathcal{B}$ can program $H(\cdot)$ and return g^b. We let $g^a = g^{\lambda_s}$, for the challenge markings m, $\mathcal{B}$ cannot reply m belongs to $\mathcal{H}_{1,i-1}$ or $\mathcal{H}_{1,i}$, and send m to $\mathcal{A}$. For $\mathcal{A}$, the markings $m = g^c$ if $c = ab$, otherwise markings $m = g_i$ (since g_i is uniformly random). Therefore, if $\mathcal{A}$ judges m belongs to $\mathcal{H}_{1,i-1}$, then $\mathcal{B}$ outputs $c = ab$; if $\mathcal{A}$ judges m belongs to $\mathcal{H}_{1,i}$, then $\mathcal{B}$ outputs c is random. That is to say, if $\mathcal{A}$ can distinguish which hybrids are markings, then $\mathcal{B}$ can solve the DDH problem with the same probability.

This completes the proof of security of Π^{IH-PSI} against a semi-honest $\mathcal{S}$. □

Security analysis of SecMM

In Algorithm 4, we can see that the messages obtained by $\mathcal{C}$ is the ciphertext $[\![\mathbf{Z_1}]\!]$, the messages obtained by $\mathcal{S}$ is the ciphertext $[\![\mathbf{X}]\!]$. For $\mathcal{C}$, its private input is protected by HE technology, as long as the private key is not disclosed, the private input is safe. For $\mathcal{S}$, its private input is hidden in the ciphertext $[\![\mathbf{Z_1}]\!]$. Formally, we have the following theorem.

Theorem 7 (Security of Π^{SecMM} against a semi-honest $\mathcal{C}$). *Assume that additive HE scheme is indistinguishable under chosen-plaintext attacks. Then the protocol of SecMM is secure in Definition 1.*

Proof We Construct a PPT simulator $SIM_{\mathcal{C}}$ to simulate the view of $\mathcal{C}$ in the protocol execution. For the functionality $\mathcal{F}_{SecMM}$, $VIEW_{\mathcal{C}}^{\Pi}(X, Y, \mathcal{K}, pk_c, sk_c)$ consists of $\mathcal{C}$'s input X, randomness r_c and the obtained ciphertext $[\![\mathbf{Z_1}]\!]$.
Given $\mathcal{K}$, pk_c, sk_c, $\mathbf{X}$ and $\mathbf{Z}_1$, $SIM_{\mathcal{C}}$ generates a simulation of $VIEW_{\mathcal{C}}^{\Pi}(\mathbf{X}, \mathbf{Y}, \mathcal{K}, pk_c, sk_c)$ as follows. It encrypts $\mathbf{Z_1}$ with pk_c and obtains $[\![\mathbf{Z}_1]\!]'$. Then, it generates $(pk_c, sk_c, r_c, [\![\mathbf{Z}_1]\!]')$ as the output. Therefore, we can get the following two equations:

$$VIEW_{\mathcal{C}}^{\Pi}(\mathbf{X}, \mathbf{Y}, \mathcal{K}, pk_c, sk_c) = (pk_c, sk_c, r_c, [\![\mathbf{Z}_1]\!]) \quad (16)$$

$$SIM_{\mathcal{C}}(1^{\mathcal{K}}, pk_c, sk_c, \mathbf{X}, \mathbf{Z}_1) = (pk_c, sk_c, r_c, [\![\mathbf{Z}_1]\!]') \quad (17)$$

It is observed that both $[\![\mathbf{Z}_1]\!]$ and $[\![\mathbf{Z}_1]\!]'$ serve as the ciphertext for $\mathbf{Z}$, and they appear indistinguishable to $\mathcal{C}$. Consequently, the probability distributions of $\mathcal{C}$'s view and $SIM_{\mathcal{C}}$'s output are identical. Hence, we claim that Eq. (3) holds. This completes the proof of security of Π^{SecMM} in case of a semi-honest $\mathcal{C}$. □

Theorem 8 (Security of Π^{SecMM} against a semi-honest $\mathcal{S}$). *Assume that additive HE scheme HE is indistinguishable under chosen-plaintext attacks. Then the protocol of SecMM is secure in Definition 1.*

Table 3 Vertical partition of the used dataset

Dataset	Samples	Partition settings	
		Features for $\mathcal{C}$	Features for $\mathcal{S}$
Diabetes	768	4	4
Breast cancer	569	10	20

Proof We Construct a PPT simulator $SIM_{\mathcal{S}}$ to simulate the view of $\mathcal{S}$ in the protocol execution. For the functionality $\mathcal{F}_{SecMM}$, $VIEW_{\mathcal{S}}^{\Pi}(\mathbf{X}, \mathbf{Y}, \mathcal{K}, pk_c, sk_c)$ consists of $\mathcal{S}$'s input $\mathbf{Y}$, randomness r_s and the obtained ciphertext $[\![\mathbf{X}]\!]$.
Given $\mathcal{K}$, pk_c, sk_c, $\mathbf{Y}$ and $\mathbf{Z}_2$, $SIM_{\mathcal{S}}$ generates a simulation of $VIEW_{\mathcal{S}}^{\Pi}(\mathbf{X}, \mathbf{Y}, \mathcal{K}, pk_c, sk_c)$ as follows. It randomly selects a matrix $\mathbf{X}'$, encrypts it with pk_c and obtains $[\![\mathbf{X}]\!]'$. Then, it generates $(\mathbf{Y}, r_s, [\![\mathbf{X}]\!]')$ as the output. Therefore, we can get the following two equations:

$$VIEW_{\mathcal{S}}^{\Pi}(\mathbf{X}, \mathbf{Y}, \mathcal{K}, pk_c, sk_c) = (\mathbf{Y}, r_s, [\![\mathbf{X}]\!], \mathbf{Z}_2) \quad (18)$$

$$SIM_{\mathcal{S}}(1^{\mathcal{K}}, \mathbf{Y}, \mathbf{Z}_2) = (\mathbf{Y}, r_s, [\![\mathbf{X}]\!]', \mathbf{Z}_2) \quad (19)$$

It is observed that as the additive HE is indistinguishable under chosen-plaintext attacks, the probability distributions of $\mathcal{S}$'s view and$SIM_{\mathcal{S}}$'s output are computationally indistinguishable. Hence, we claim that Eq. (3) holds. This completes the proof of security of Π^{SecMM} in case of a semi-honest $\mathcal{S}$. □

Security analysis of IH-VLR

In algorithm 5, We can see that during the model training process, $\mathcal{C}$ and $\mathcal{S}$ finish interactive computing tasks by the protocol of *SecMM*, and other computing tasks are finished locally. Therefore, algorithm 5 is secure, if the protocol of *SecMM* is secure. Formally, we have the following theorem.

Theorem 9 (Security of Π^{IH-VLR} against a semi-honest $\mathcal{C}$). *Assume that the protocol of SecMM is secure against a semi-honest $\mathcal{C}$, then the protocol of IH-VLR is secure in Definition 1.*

Theorem 10 (Security of Π^{IH-VLR} against a semi-honest $\mathcal{S}$). *Assume that the protocol of SecMM is secure against a semi-honest $\mathcal{S}$, then the protocol of IH-VLR is secure in Definition 1.*

Proof Based on the above analysis, see the proof of Theorem 7 and Theorem 8 for more details. □

Experiments

In this section, we provided a few experiments to validate the feasibility and performance of our scheme.

Setup

Dataset description

We use two classical classification benchmark datasets from the UCI repository in our experiments: Diabetes (Smith et al 1988) and Breast Cancer (Bache and Lichman 2013).

- **Diabetes:** This dataset consists of medical measurements that correspond to 768 female patients older than 21 years old. Each sample has 7 features which includes blood pressure, body mass index, age and plasma glucose concentration, etc.
- **Breast Cancer:** It contains 569 samples with 30 dimensions, which 357 are benign, and 212 are malignant. It is also a binary classification dataset.

Implementation settings

We implement the system in Python, and all experiments are performed on an Intel Core i7-7500U @ 2.70GHz, 2 CPU cores and 12GB RAM. In our settings, we assume that the secure screening process has been completed, which means that the input to the model is a shares of the data. Meanwhile, we omit the performance analysis of PSI and only focus on the part of model training. Besides, we divide the dataset vertically into two parts and distribute them to party $\mathcal{C}$ and $\mathcal{S}$. Table 3 describes the partition details.

Parameters settings

We use the paillier (Paillier 1999) as our additive HE scheme and set the key length to 1024 bits. For diabetes dataset, we set the batch size, iteration and learning rate are 64, 30, 0.1. For breast cancer dataset, such settings are 32, 30, 0.05, respectively. In addition, we set the training and test sets according to the ratio of 7:3.

Comparison methods

To evaluate the effectiveness of our scheme, we make some comparison experiments with existing related work, which also use the two-party LR model. First, we use plaintext logistic regression as a baseline and denote it as 'BaselineLR'. Besides, we implement the works (Hardy et al 2017; Yang et al 2019b; Chen et al 2021), and denote them as 'SSHELR', 'HECLR', 'HELR' in our settings, respectively. In the implementation details, Hardy et al (2017) uses HE to protect the gradient. On this basis, Yang et al (2019b) proposed a two party scheme without third-party coordinator, which is more suitable for real-world application scenarios. Chen et al (2021) combines the HE and SS together to solve the problem of high-dimensional and sparse data in the risk control scenario. The code is available at the address https://github.com/hellolsk/IHVFL.

Comparisons results and analysis

We first analyze the complexity of secure screening protocols. Assume that the size of dataset is $n \times m$ and the target feature number is k, and then the time complexity of the secure screening protocols is about $O(nm \times T_{Enc} + nk \times T_{Add} + nk \times T_{Dec})$, where the $T_{Enc}, T_{Add}, T_{Dec}$ represents the single encryption, homomorphic addition and decryption time, respectively. It is clear that the cost of protocol execution increases with the size of dataset, which means that when the size of data is large it can seriously affect the efficiency of protocol. In this way, optimizing the secure screening protocol will be our future work.

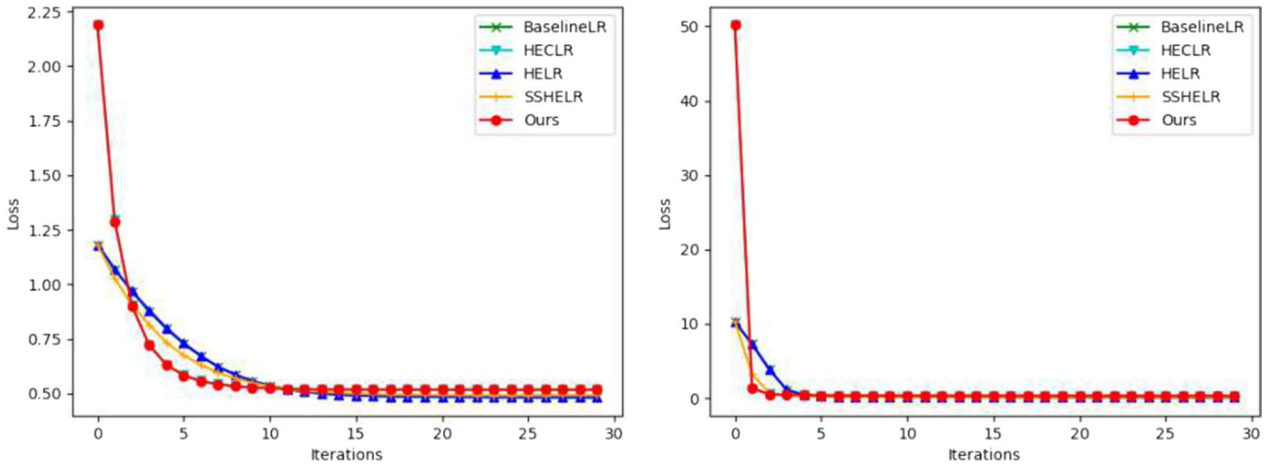
Effectiveness

As shown in Table 4, we test our proposed scheme with related works. In our experiments, the baseline is the model trained in a plaintext manner. It is clear that the baseline has a best performance in all evaluation metrics, which is as expected. We also test the main metrics comparison with respect to iterations on different schemes. In Fig. 4, the loss function converges very fast. What is more, the loss on diabetes Fig. 4a and breast cancer Fig. 4b tends become stable and nearly reach the same with the number of iterations increase. This is because they all use efficient optimization solutions on sigmoid function, i.e., Taylor expansion (Hardy et al 2017) and Minimax approximation (Chen et al 2018). In

Table 4 The performance of IHVLR with different datasets

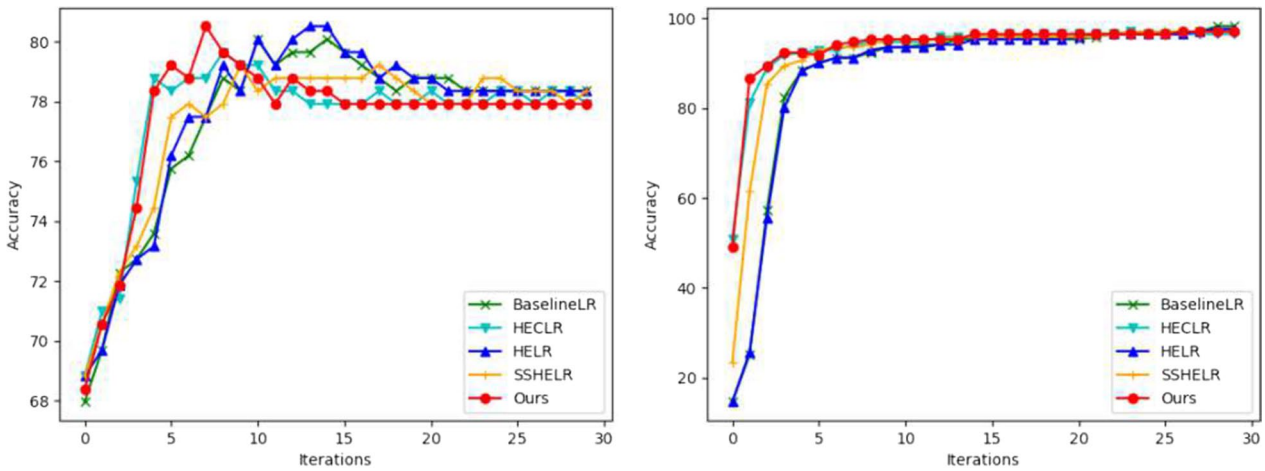
Related works	Scheme	Without coordinator	Diabetes dataset			Breast cancer dataset		
			Accuracy (%)	AUC	Runtime (s)	Accuracy (%)	AUC	Runtime (s)
Baseline	Plaintext	–	78.355	0.864	0.095	98.246	0.999	0.069
HECLR (Hardy et al 2017)	HE	✗	77.922	0.864	75.552	96.491	0.999	107.251
HELR (Yang et al 2019b)	HE	✓	78.355	0.865	33.273	97.661	0.999	25.265
SSHELR (Chen et al 2021)	SS+HE	✓	78.355	0.864	93.014	97.076	0.999	153.946
Ours	SS+HE	✓	**77.922**	**0.864**	**190.423**	**97.076**	**0.999**	**265.623**

In order to distinguish it from the experimental results of other schemes, we display the experimental data of our scheme in bold



(a) Test Loss with different schemes over Diabetes (b) Test Loss with different schemes over Breast Cancer

Fig. 4 Loss comparison with respect to iterations over different schemes



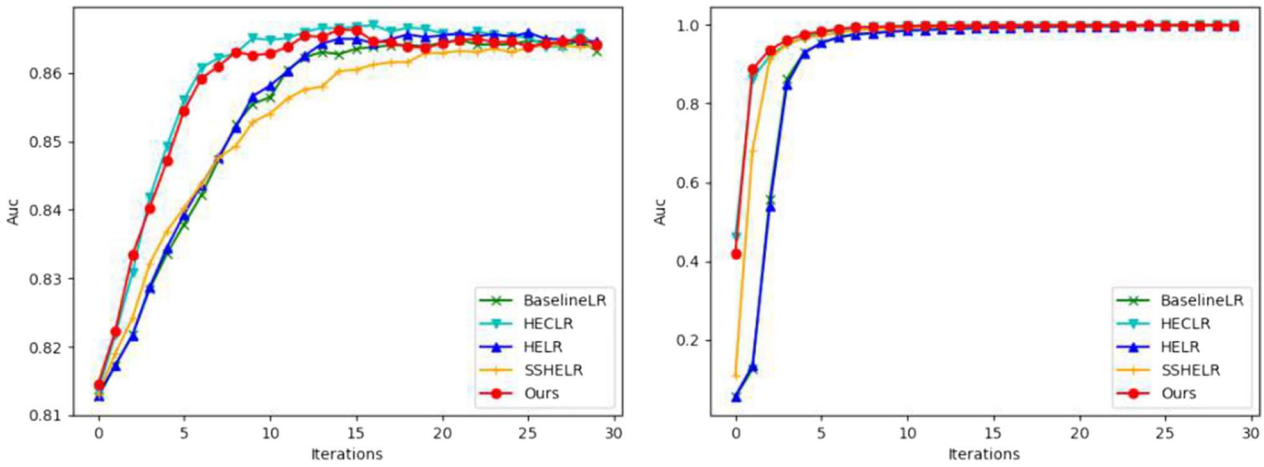
(a) Test Accuracy with different schemes over Diabetes (b) Test Accuracy with different schemes over Breast Cancer

Fig. 5 Accuracy comparison with respect to iterations on different schemes

Figs. 5 and 6, we test the accuracy and AUC of models, there are small differences in performance across datasets. For breast cancer dataset, the two metrics performance well with different schemes in Figs. 5b and 6b. For example, when the iteration reaches 30, the AUC is 0.999 and the accuracy reaches 96%, which the best performance is Baseline, followed by Yang et al (2019b) and finally SS+HE scheme. For diabetes dataset in Fig. 5a, we observe that our scheme just drop by 0.4% compared the baseline and other works (Yang et al 2019b; Chen et al 2021) on accuracy. In Fig. 6a, we can see that our scheme consistently achieve better AUC than others besides work (Hardy et al 2017), although they eventually converged. Therefore, our scheme is able to achieve better performance in a shorter time.

Efficiency

To evaluate the efficiency of our work, we test the runtime of different scheme. From the Table 4, we can see that the work (Yang et al 2019b) has better performance in VFL settings, since that it has less homomorphic operations. For the 'SS+HE' scheme, it has more encryption operations based the shares, which costs much time compared to 'HE' scheme (Hardy et al 2017). In addition, for work (Chen et al 2021), it shares the model while we share the data, so our scheme has much time cost on the matrix multiplication operations. Fortunately, the additional cost is acceptable. Besides, the parties communicate using local sockets in our settings, so the network latency is not measured in experiments.



(a) Test AUC with different schemes over Diabetes (b) Test AUC with different schemes over Breast Cancer

Fig. 6 AUC comparison with respect to iterations on different schemes

Security

To begin with, the plaintext scheme has best performance, but it does not fit the VFL scenarios. Besides, the other schemes satisfy the basic security definition in VFL. For HECLR, it needs a trusted third-party, which is not allowed or has security problems in some scenarios. For HELR and SSHELR,they remove the third-party and solve the problem of data sparse in specific scenarios, respectively. However, none of them consider the security of the data preparation process. Our proposed scheme based the requirement of intention-hiding, not only achieves privacy enhancement but also guarantees model performance.

Conclusions

In this paper, we studied the intention-hiding in model training to solve the privacy-preserving requirements in real-life applications. To do this, we first proposed the idea of intention-hiding vertical federated learning. First, we constructed two secure screening protocols to enhance the feature engineering, and then we presented a new PSI protocol to achieve the sample alignment. Next, we used the logistic regression to present the process of intention-hiding vertical federated learning by combing homomorphic encryption and secret sharing. Finally, we implemented the framework and conducted experiments on it. In future, We will explore more solutions to optimize the efficiency of framework, and expand our framework to more machine learning models.

Acknowledgements
Not applicable.

Author Contributions
All authors read and approved the fnal manuscript.

Funding
This work was supported by the National Key Research and Development Program of China under Grant 2021YFF0704102.

Availability of data and materials
The authors confirm that the data supporting the findings of this study are available within the article.

Declarations

Competing interests
The authors declare that they have no competing interests.

Received: 20 March 2023 Accepted: 8 June 2023
Published online: 04 October 2023

References

Abuadbba S, Kim K, Kim M, Thapa C, Camtepe SA, Gao Y, Kim H, Nepal S (2020) 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

Aono Y, Hayashi T, Wang L, Moriai S et al (2017) Privacy-preserving deep learning via additively homomorphic encryption. IEEE Trans Inf Forensics Secur 13(5):1333–1345

Aono Y, Hayashi T, Trieu Phong L, Wang L (2016) Scalable and secure logistic regression via homomorphic encryption. In: Proceedings of the sixth ACM conference on data and application security and privacy, pp 142–144

Bache K, Lichman M (2013) UCI machine learning repository. http://archive.ics.uci.edu/ml

Beaver D (1991) Efficient multiparty protocols using circuit randomization. In: Annual international cryptology conference. Springer, pp 420–432

Brisimi TS, Chen R, Mela T, Olshevsky A, Paschalidis IC, Shi W (2018) Federated learning of predictive models from federated electronic health records. Int J Med Inform 112:59–67

Buddhavarapu P, Knox A, Mohassel P, Sengupta S, Taubeneck E, Vlaskin V (2020) Private matching for compute. Cryptol ePrint Arch

Caruana R, Lou Y, Gehrke J, Koch P, Sturm M, Elhadad N (2015) Intelligible models for healthcare: predicting pneumonia risk and hospital 30-day readmission. In: Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining, pp 1721–1730

Chen H, Gilad-Bachrach R, Han K, Huang Z, Jalali A, Laine K, Lauter K (2018) Logistic regression over encrypted data from fully homomorphic encryption. BMC Med Genomics 11(4):3–12
Chen P, Du X, Lu Z, Wu J, Hung PC (2022) Evfl: an explainable vertical federated learning for data-oriented artificial intelligence systems. J Syst Architect 126(102):474
Chen C, Zhou J, Wang L, Wu X, Fang W, Tan J, Wang L, Liu AX, Wang H, Hong C (2021) When homomorphic encryption marries secret sharing: secure large-scale sparse logistic regression and applications in risk control. In: Proceedings of the 27th ACM SIGKDD conference on knowledge discovery and data mining, pp 2652–2662
Debnath SK, Dutta R (2015) Secure and efficient private set intersection cardinality using bloom filter. In: International conference on information security. Springer, pp 209–226
Dwork C (2008) Differential privacy: A survey of results. In: International conference on theory and applications of models of computation. Springer, pp 1–19
Fang P, Cai Z, Chen H, Shi Q (2020) Flfe: a communication-efficient and privacy-preserving federated feature engineering framework. arXiv:2009.02557
Fu C, Zhang X, Ji S, Chen J, Wu J, Guo S, Zhou J, Liu AX, Wang T (2022) Label inference attacks against vertical federated learning. In: 31st USENIX security symposium (USENIX Security 22), pp 1397–1414
Gao D, Liu Y, Huang A, Ju C, Yu H, Yang Q (2019) Privacy-preserving heterogeneous federated transfer learning. In: IEEE international conference on big data (big data). IEEE, pp 2552–2559
Garg A, Mago V (2021) Role of machine learning in medical research: a survey. Comput Sci Rev 40(100):370
Hardy S, Henecka W, Ivey-Law H, Nock R, Patrini G, Smith G, Thorne B (2017) Private federated learning on vertically partitioned data via entity resolution and additively homomorphic encryption. arXiv:1711.10677
Huang L, Shea AL, Qian H, Masurkar A, Deng H, Liu D (2019) Patient clustering improves efficiency of federated machine learning to predict mortality and hospital stay time using distributed electronic medical records. J Biomed Inform 99(103):291
Jothi N, Husain W et al (2015) Data mining in healthcare-a review. Procedia Comput Sci 72:306–313
Kolesnikov V, Kumaresan R, Rosulek M, Trieu N (2016) Efficient batched oblivious PRF with applications to private set intersection. In: Proceedings of the 2016 ACM SIGSAC conference on computer and communications security, pp 818–829
Li W, Milletarì F, Xu D, Rieke N, Hancox J, Zhu W, Baust M, Cheng Y, Ourselin S, Cardoso MJ, et al (2019) Privacy-preserving federated brain tumour segmentation. In: International workshop on machine learning in medical imaging. Springer, pp 133–141
Liu Y, Kang Y, Xing C, Chen T, Yang Q (2020) A secure federated transfer learning framework. IEEE Intell Syst 35(4):70–82
Liu P, Xu X, Wang W (2022) Threats, attacks and defenses to federated learning: issues, taxonomy and perspectives. Cybersecurity 5(1):1–19
Liu Y, Kang Y, Zou T, Pu Y, He Y, Ye X, Ouyang Y, Zhang YQ, Yang Q (2022b) Vertical federated learning. arXiv:2211.12814
Liu Y, Zhang X, Wang L (2020b) Asymmetrical vertical federated learning. arXiv:2004.07427
Magoulas GD, Prentza A (1999) Machine learning in medical applications. In: Advanced course on artificial intelligence. Springer, pp 300–307
McMahan B, Moore E, Ramage D, Hampson S, y Arcas BA (2017) Communication-efficient learning of deep networks from decentralized data. In: Artificial intelligence and statistics. PMLR, pp 1273–1282
Meadows C (1986) A more efficient cryptographic matchmaking protocol for use in the absence of a continuously available third party. In: 1986 IEEE symposium on security and privacy. IEEE, p 134
Mohassel P, Zhang Y (2017) Secureml: a system for scalable privacy-preserving machine learning. In: 2017 IEEE symposium on security and privacy (SP). IEEE, pp 19–38
Paillier P (1999) Public-key cryptosystems based on composite degree residuosity classes. In: International conference on the theory and applications of cryptographic techniques. Springer, pp 223–238
Qayyum A, Qadir J, Bilal M, Al-Fuqaha A (2020) Secure and robust machine learning for healthcare: a survey. IEEE Rev Biomed Eng 14:156–180
Rieke N, Hancox J, Li W, Milletari F, Roth HR, Albarqouni S, Bakas S, Galtier MN, Landman BA, Maier-Hein K et al (2020) The future of digital health with federated learning. NPJ Digit Med 3(1):119
Roth HR, Chang K, Singh P, Neumark N, Li W, Gupta V, Gupta S, Qu L, Ihsani A, Bizzo BC, et al (2020) Federated learning for breast density classification: A real-world implementation. In: Domain adaptation and representation transfer, and distributed and collaborative learning. Springer, pp 181–191
Shamir A (1979) How to share a secret. Commun ACM 22(11):612–613
Shokri R, Shmatikov V (2015) Privacy-preserving deep learning. In: Proceedings of the 22nd ACM SIGSAC conference on computer and communications security, pp 1310–1321
Smith JW, Everhart JE, Dickson W, Knowler WC, Johannes RS (1988) Using the ADAP learning algorithm to forecast the onset of diabetes mellitus. In: Proceedings of the annual symposium on computer application in medical care. American Medical Informatics Association, p 261
Sun C, Ippel L, Van Soest J, Wouters B, Malic A, Adekunle O, van den Berg B, Mussmann O, Koster A, van der Kallen C, et al (2019) A privacy-preserving infrastructure for analyzing personal health data in a vertically partitioned scenario. In: MedInfo. pp 373–377
Sun L, Qian J, Chen X (2020) Ldp-fl: practical private aggregation in federated learning with local differential privacy. arXiv:2007.15789
Sun J, Yang X, Yao Y, Zhang A, Gao W, Xie J, Wang C (2021) Vertical federated learning without revealing intersection membership. arXiv:2106.05508
Tang F, Ling GW, Shan JY (2022) Additive homomorphic encryption schemes based on sm2 and sm9. J Cryptol Res 9(3):535–549
Xia Z, Gu Q, Zhou W, Xiong L, Weng J, Xiong N (2021) STR: Secure computation on additive shares using the share-transform-reveal strategy. IEEE Trans Comput
Yang Q, Liu Y, Chen T, Tong Y (2019) Federated machine learning: concept and applications. ACM Trans Intell Syst Technol (TIST) 10(2):1–19
Yang S, Ren B, Zhou X, Liu L (2019b) Parallel distributed logistic regression for vertical federated learning without third-party coordinator. arXiv:1911.09824
Zhu L, Liu Z, Han S (2019) Deep leakage from gradients. Adv Neural Inf Process Syst 32

Publisher's Note