Secure Interaction-based Feature Selection for Vertical Federated Learning

Zhenyu Meng[1], Wenmiao Zhang[1], Shuaiqi Shen[2], Chong Yu[3], Kuan Zhang[1]
1. Department of Electrical and Computer Engineering, University of Nebraska-Lincoln, Lincoln, USA
2. Department of Electrical Engineering, University of Wisconsin Milwaukee, Milwaukee USA
3. Department of Computer Science, University of Cincinnati, Cincinnati, USA
Email:{zmeng4@huskers.unl.edu, wzhang40@huskers.unl.edu, shen8@uwm.edu, yuc5@ucmail.uc.edu, kuan.zhang@unl.edu}

***Abstract*—Federated learning enables decentralized data owners to collaborate and train models in a distributed manner. A special type is Vertical Federated Learning (VFL), where each of the participated data owners only has a portion of the data features. To maintain a high accuracy and reasonable computational cost, selecting a set of features among the entire dataset is essential. Although some existing work selects features by calculating their individual contributions to the learning outcomes, knowing the joint contribution from multiple features becomes necessary but challenging. Meanwhile, security concerns are raised when calculating the joint contribution of a set of features where the feature data are stored by different owners. Using homomorphic encryption or secure computing over encrypted data is possible, but it may cost too much when complex calculations are involved and repeated. To this end, this paper proposes a privacy-preserving feature selection protocol that considers the interactions between features stored across different data owners. Specifically, we first propose an interaction-based feature selection algorithm for vertically distributed datasets. This algorithm estimates the features' joint contributions to the model training outcomes. Then, we propose a privacy-preservation protocol to prevent the semi-honest cloud server from obtaining or inferring the raw data when aggregating the knowledge and calculating the complex interaction measure for feature selection. We create a new approximation method for interaction measures to address the high computational cost when securely calculating the interaction measure while maintaining the training accuracy. The security discussions show that the proposed protocol preserves data owner's privacy. The extensive simulations validate the achieved training accuracy and efficiency.**

***Index Terms*—Vertical federated learning, feature selection, feature interactions, privacy-preserving,**

I. Introduction

Federated learning (FL) enables decentralized users to collaboratively train a machine learning (ML) model without sending their local data samples to a centralized cloud server. FL can be categorized into Horizontal FL (HFL) and Vertical FL (VFL) depending on the patterns of data partitioning [1] . VFL [2] operates when data sources possess a portion of different features from the same data samples, i.e., in a complementary manner. In contrast, HFL operates when all data sources have the same features but different data samples. For example, in the e-health system, a hospital wants to develop ML models that provide personalized treatment plans for patients with specific medical conditions. But the data sources may include daily activity data measured outside of the hospital and electronic health records stored in the hospital. VFL can help the hospital train the ML model together with various patients without centrally streaming all patient information. This can dramatically reduce transmission costs and involve diverse data sources to provide a robust and generalized model for healthcare services.

However, there are several technical challenges in VFL. For example, the training data may contain many features, but some of them are irrelevant [1]. Combining these irrelevant features has trivial contributions to the learning outcomes. It increases the complexity of ML models with additional training overhead but degrades the accuracy with overfitting problems. Meanwhile, the interactions among features are simply ignored by many existing feature selections [3] [4]. As multiple features jointly determine the model outcomes, their joint contributions to the model training process can hardly be measured independently. Increasing the values of certain features may alter the significance of others. This poses the demand of selecting relevant features for VFL by considering feature interactions, i.e., "chemical reaction" among features.

In addition to the above efficiency and accuracy problems, security concerns are raised, especially when conducting interaction-based feature selection in VFL [5]. To measure the joint contribution of features, data sources are required to share their training data to combine all features related to the samples and evaluate the feature interactions. This may disclose the private data in some sources [6], [7]. Although data may be encrypted before feature selection, calculating the interaction measure involves some complicated operations instead of a combination of addition and multiplication, which are possibly performed over encrypted data. Therefore, a new secure feature selection protocol for VFL is in urgent demand to preserve user privacy and enable to calculate the complicated interaction measures during feature selection.

In this paper, we propose a privacy-preserving interaction-based feature selection method for VFL. The main contributions are summarized as follows.

- We investigate VFL with feature selection where each data owner holds a portion of features but does not want to share his training data. The security threats of a semi-honest cloud server in the system are analyzed and the design goals of the protocol are identified for privacy-preserving feature selection.
- To improve the VFL accuracy, we design an interaction-

ICC 2024 - IEEE International Conference on Communications | 978-1-7281-9054-9/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICC51166.2024.10622749

Authorized licensed use limited to: University of Cincinnati. Downloaded on January 23,2025 at 01:44:42 UTC from IEEE Xplore. Restrictions apply.

based feature selection algorithm. This allows the feature selection by considering the joint significance among features, so that the model's accuracy can be improved.
- To preserve privacy when calculating the interaction measure among feature data, we propose a privacy-preserving interaction-based feature selection protocol. We create an approximation method to measure the interactions with simplified operation but maintain its accuracy.
- Our newly created approximation method fits the homomorphic encryption so that it transmits the pre-processed ciphertexts that carry all related information of feature interactions from data owners to the cloud server. The cloud server operates over encrypted data, and returns feature selection results without any inferred knowledge.
- We discuss the security features of our proposed protocol that preserves data owner privacy during the interaction-based feature selection in VFL. We also conduct simulations to validate that the proposed protocol achieves identical model accuracy with fewer selected features for VFL compared with existing works.

The remainder of this paper is organized as follows. Section II reviews the related work. Section III introduces the system and attack models. Section IV presents the proposed scheme. Section V discusses security features. Section VI shows simulation results, and Section VII draws conclusions.

II. Related Work

In this section, we present the related work on feature selection and secure computing. We identify the unique challenges of integrating secure computing into feature selection.

Feature selection is critical to VFL, as irrelevant features may increase the complexity of ML models but reduce the accuracy due to overfitting problems [8]. The features can be evaluated based on either the single feature's significance [8] or the joint significance derived from the features' interaction measure. To estimate the single feature's significance, Gini score or Gini impurity [1], [9] may be used to estimate the possibility of the sample matching an incorrect label according to this feature. Then, the features are selected by ranking each feature's Gini score. However, the single feature's metrics cannot indicate the joint significance or interaction between features, which is critical to select a set of features. Shen et al. [10] studied feature interaction to determine a set of features to detect malicious websites. Zeng et al. [6] proposed an Interaction Weight-based Feature Selection algorithm (IWFS) to select the redundant features. The interaction weight factor is defined to reflect whether a feature is redundant or interactive.

However, selecting or evaluating features needs a cloud server to compute over the raw data. When the cloud server is not fully trusted, security and privacy concerns are raised [5], [11]. Encrypting the raw data before sending it to the cloud server is a possible solution that homomorphic encryption can support. Zhang et al. [1] proposed a secure feature selection for VFL based on Gini scores. In this method, the feature data owner and the label data owner collaborate compute, and rank the Gini score of each feature without sharing any raw

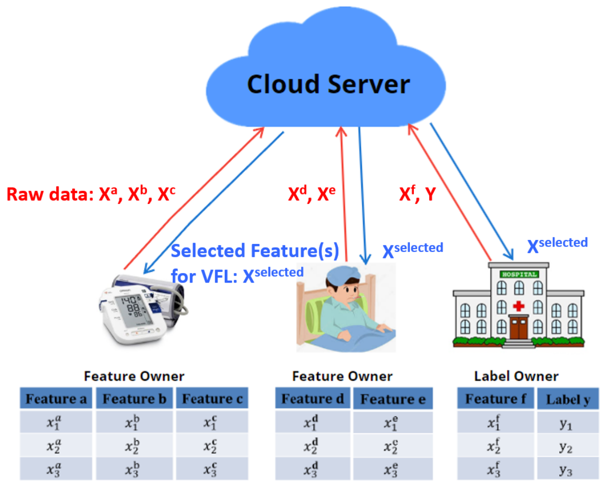

Fig. 1. Selected feature number comparison

training data. Other works [3], [9], [12] also designed secure computing methods for a single feature's significance.

Although existing interaction measures or functions can select features by considering feature interactions, they need to calculate the complex interaction functions, such as logarithm, prior probability, or mutual information. It dramatically increases the secure computing cost and even prevents from using homomorphic encryption and other secure computing solutions. The above technical challenges motivate us to design an efficient secure interaction measure protocol to select features for VFL.

III. System Model Overview

A. System Model

In vertical feature selection, the system mainly consists of a cloud server and multiple data owners. The labels of the training dataset are privately owned by only one label data owner, and the other owners have their own features. The capability of each entity is described as follows.

- *Dataset owners:* Data owners have distinct sets of features and usually store a large amount of data samples that can be utilized for model training. Prior to participating in VFL, data owners want to identify several most significant features to improve training efficiency and model accuracy. The raw data including features and label values are transmitted to the cloud server. As the feedback, each data owner receives the decision from the cloud on which features from its local dataset, which is evaluated to be the most relevant for inferring label values, are selected to perform VFL.
- *Cloud Server:* The computing power of the cloud server is leveraged by the data owners to accurately measure feature interactions with transmitted raw data by applying specific interaction-based feature selection methods [6], [10]. Then, the results of selected features for VFL are returned to corresponding data owners.

B. Attack model

The attack is launched via the semi-honest cloud server, which honestly follows the feature selection process by receiv-

Authorized licensed use limited to: University of Cincinnati. Downloaded on January 23,2025 at 01:44:42 UTC from IEEE Xplore. Restrictions apply.

ing the raw data from various data owners and computing the interactions among features. However, the cloud server is also curious about private information, such as the feature values and labels of the training data. With full access to the raw data, the attackers from the semi-honest server can leak data owners' privacy without their awareness.

C. Design goals

Considering the aforementioned attack model, our work aims to achieve the following design goals.

- **Interaction-based feature selection**: With the features and labels vertically distributed in different data owners, the interactions among features should be evaluated to characterize their joint impacts on the learning outcomes of the federated learning model. By selecting a limited number of the most significant features, the VFL process should consume less training overhead while maintaining desired model accuracy.
- **Privacy preservation**: Private information and raw data of the data owners should be protected against the honest-but-curious cloud server during the feature selection. The cloud server should be prohibited from obtaining the raw data but still capable of leveraging feature significance knowledge for evaluating the feature interactions.

IV. Proposed Protocol

A. Preliminaries

1) Interaction Weight-based Feature Selection (IWFS): An Interaction Weight-based Feature Selection Algorithm (IWFS) was proposed to solve the feature selection by considering feature interactions. In the algorithm, the adjusted relevant measure is defined to reflect whether a feature is redundant or interactive. The feature with the largest adjusted relevant measure is first selected, and then the interaction weight factor to update the weight of the rest features before the next round of selection. The selection process is iterated several times to determine the most relevant features.

We assume the training dataset includes a feature set $\mathbf{F}$ and the label set $\mathbf{Y}$. Firstly, we initialize the weight for each feature $w_0 = 1$, and the adjusted relevance measure $R(\mathbf{F_i}; \mathbf{Y})$ for feature $\mathbf{F_i}$ is defined as follows:

$$R(\mathbf{F_i}; \mathbf{Y}) = w_0(\mathbf{F_i}) \times (1 + SU(\mathbf{F_i}; \mathbf{Y})), \tag{1}$$

where the Symmetrical Uncertainty of feature $\mathbf{F_i}$ is

$$SU(\mathbf{F_i}; \mathbf{Y}) = \frac{2 \times I(\mathbf{F_i}; \mathbf{Y})}{S(\mathbf{F_i}) \times S(\mathbf{Y})}. \tag{2}$$

Here $I(\mathbf{F_i}, \mathbf{Y})$ denotes the mutual information of the feature $\mathbf{F_i}$, and S denotes the entropy which is a measure of uncertainty. The mutual information and entropy are defined as:

$$I(\mathbf{F}; \mathbf{Y}) = \sum_{i=1}^{n} \sum_{j=1}^{m} \mathsf{Prob}(f_i, y_j) \times log \frac{\mathsf{Prob}(f_i|y_j)}{\mathsf{Prob}(f_i)}, \tag{3}$$

$$S(\mathbf{X}) = \sum_{i=0}^{1} \mathsf{Prob}(x = i) \times log\mathsf{Prob}(x = i), \tag{4}$$

$$S(\mathbf{Y}) = \sum_{j=1}^{m} \mathsf{Prob}(y_j) \times log\mathsf{Prob}(y_j), \tag{5}$$

where $\mathsf{Prob}(f_i)$ denotes the probability of a sample with the feature value $f = i$, $\mathsf{Prob}(f_i, y_j)$ donates the probability of a sample with the feature value $f = i$ as well as having the label $y = j$, and $\mathsf{Prob}(f_i|y_j)$ donates the probability of a sample with the feature value $f = i$ when the label $y = j$.

Then the feature $\mathbf{F_i}$ with the largest $R(\mathbf{F_i}; \mathbf{Y})$ is selected in the first round. After the selection, the weight of the rest features $\mathbf{F_j}$ will be updated as follows:

$$w_1(\mathbf{F_j}) = w_0(\mathbf{F_j}) \times IW(\mathbf{F_i}; \mathbf{F_j}), \tag{6}$$

where $IW(\mathbf{F_i}; \mathbf{F_j})$ denotes the interaction weight factor between two features.

Finally, the adjusted relevant measure in the next round could be computed as Eq.(1) by using w_1 instead of w_0. The selection will be iterated in several rounds to select the most relevant features.

2) The BGV Scheme: Our protocol in this work is constructed based on the homomorphic encryption technique. Without loss of generality, we employ the BGV scheme in [13] as the homomorphic encryption scheme, which can support homomorphic addition and multiplication over ciphertexts, i.e., $[m_1] + [m_2]$ mod N $\rightarrow [m_1 + m_2]$, $[m_1] + m_2$ mod N $\rightarrow [m_1 + m_2]$, $[m_1] * [m_2]$ mod N $\rightarrow [m_1 * m_2]$, and $[m_1] * m_2$ mod N $\rightarrow [m_1 * m_2]$.

B. Proposed Protocol

To achieve our design goals, we propose a privacy-preserving interaction-based feature selection protocol for VFL. Our protocol consists of initialization, feature selection, weight updating, and round iteration processes. We first introduce the system setup and then present our protocol in detail.

1) Initialization: In our protocol, every feature data owner ($\mathcal{FO}$) has the feature vector $\vec{D}$, and the label data owner $\mathcal{LO}$ has the label vector $\vec{L}$.

Firstly, $\mathcal{FO}$ and $\mathcal{LO}$ compute $Init(\vec{D})$ and $Init(\vec{L})$ locally to obtain the matrix $\mathbf{X}$, $\mathbf{A}$, $\mathbf{B}$, $\mathbf{Y}$, and entropy $S(\mathbf{X})$, $S(\mathbf{Y})$. As shown in **Algorithm 1**, $\mathcal{FO}$ firstly computes the mean value of their feature data as the threshold θ and transforms the feature vector into a binary matrix $\mathbf{X}$ by comparing each feature data with θ. In $\mathbf{X}$, $x_i = 1$ when $d_i \geq \theta$, and $x_i = 0$ when $d_i < \theta$. Then, $\mathcal{FO}$ computes a diagonal matrix $\mathbf{A}$ by the binary matrix $\mathbf{X}$ and expands the $\mathbf{X}$ by adding its inversion as a new column and generates a double-column binary matrix $\mathbf{B}$. Finally, $\mathcal{FO}$ assumes the weight $w_0(\mathbf{X}) = 1$ and calculates the entropy of the feature data $S(\mathbf{X})$ by Eq.(4).

Secondly, in **Algorithm 2**, $\mathcal{LO}$ computes $S(\mathbf{Y})$ and creates matrix $\mathbf{Y}$, where the row name is the sample numbers, and the column name is all possible classes. In matrix $\mathbf{Y}$, values y_{ij}=1 when the sample d_i belongs to the class j.

Finally, a Trusted Authority ($\mathcal{TA}$) allocates the homomorphic encryption key. After $\mathcal{TA}$ sending homomorphic encryption key to both $\mathcal{FO}$ and $\mathcal{LO}$, they encrypt $\mathbf{X}, \mathbf{A}, \mathbf{B}, \mathbf{Y}, S(\mathbf{X}), S(\mathbf{Y})$, and $w_0(\mathbf{X})$ by the Fully Homomorphic Encryption Key and send the ciphertexts $[\mathbf{X}], [\mathbf{A}], [\mathbf{B}], [\mathbf{Y}], [S(\mathbf{X})], [S(\mathbf{Y})], [w_0(\mathbf{X})]$ to $\mathcal{CS}$.

Authorized licensed use limited to: University of Cincinnati. Downloaded on January 23,2025 at 01:44:42 UTC from IEEE Xplore. Restrictions apply.

Algorithm 1 Feature Data Pre-processing: $Init(\vec{D})$

```
Input: Feature vector $\vec{D} = (d_i)_n$, $n$ is sample number.
Output: $\mathbf{X} = (x_i)_n, \mathbf{A} = (a_{ij})_{n\times n}, \mathbf{B} = (b_{ij})_{n\times 2}$, feature data entropy: $S(\mathbf{X})$
 1: Compute threshold $\theta = \frac{1}{n}\sum_{i=1}^{n} d_i$
 2: Compute binary matrix $\mathbf{X} = (x_i)_n$ as follows:
 3: for $i = 1, 2, \ldots, n$ do
 4:     $x_i = \begin{cases} 0 & \text{if } d_i \geq \theta \\ 1 & \text{if } d_i < \theta \end{cases}$
 5: end for
 6: Compute the diagonal binary matrix $\mathbf{A} = (a_{ij})_{n\times n}$ as follows:
 7: if $i \neq j$ then
 8:     $a_{ij} = 0$
 9: else
10:     $a_{ij} = x_i$
11: end if
12: $(i = 1, 2, \ldots, n; j = 1, 2, \ldots, n$
13: Compute the matrix $\mathbf{B} = (b_{ij})_{n\times 2}$ as follows:
14: for $i = 1, 2, \ldots, n$ do
15:     $b_{i1} = x_i$
16:     $b_{i2} = 1 - x_i$
17: end for
18: Compute entropy of $\vec{X}$ as follows:
19: $S(\mathbf{X}) = \sum_{i=0}^{1} \mathsf{Prob}(x_i) \times log\mathsf{Prob}(x_i)$
```

Algorithm 2 Label Data Pre-processing: $Init(\vec{L})$

```
Input: Label vector $\vec{L} = (l_i)_n$, $n$ is sample number.
Output: label data entropy: $S(\mathbf{Y})$, $\mathbf{Y} = (y_{ij})_{n\times m}$ $m$ is the class number.
 1: Compute binary matrix $\mathbf{Y} = (y_{ij})_{n\times m}$ as follows:
 2: $y_{ij} = \begin{cases} 1 & \text{if } l_i = j \\ 0 & \text{else} \end{cases}$
 3: $(i = 1, 2, \ldots, n,\ j = 1, 2, \ldots, m)$
 4: Compute $S(\mathbf{Y}) = \sum_{j=1}^{m} \mathsf{Prob}(y_j) \times log\mathsf{Prob}(y_j)$
```

2) Feature Selection: After obtaining [**X**], [**A**], [**B**], [**Y**], and $[S(\mathbf{X})]$, $[S(\mathbf{Y})]$, $\mathcal{CS}$ computes the distributed matrix $[\mathbf{Z}] = [\mathbf{B}]^T \times [\mathbf{Y}]$. The distributed matrix reflects the feature and label distribution. All the elements z_{1j} in $\mathbf{Z}$ represent the number of samples with $x = 0$ and $y = j$, while the element z_{2j} means the number of samples with $x = 1$ and $y = j$. Then, as shown in **Algorithm 3**, $\mathcal{CS}$ computes $[I(\mathbf{X};\mathbf{Y})]$, $[SU(\mathbf{X};\mathbf{Y})]$, and $[R(\mathbf{X};\mathbf{Y})]$ for each feature. In the algorithm, $TL(\mathsf{Prob})$ is approximately calculated $log(\mathsf{Prob})$ by the Taylor Series. $TL(x)$ is defined as:

$$TL(x) = 0.4 \times ((x-1) - \frac{(x-1)^2}{2}).$$

Finally, $\mathcal{CS}$ returns $[R(\mathbf{X})]$ of each feature to the $\mathcal{LO}$, $\mathcal{LO}$ can decrypt the ciphertext and select the feature with the largest $R(\mathbf{X})$ in this round.

3) Weight Updating: After selecting the feature with the largest $R(\mathbf{X})$, $\mathcal{CS}$ needs to update the weight of each feature

Algorithm 3 Feature Selection

```
Input: Distributed matrix $[\mathbf{Z}] = ([z_{ij}])_{2\times m}$, $[S(\mathbf{X})]$, $[S(\mathbf{Y})]$, $[w_0(\mathbf{X})]$
Output: $[I(\mathbf{X};\mathbf{Y})]$, $[SU(\mathbf{X};\mathbf{Y})]$, $[R(\mathbf{X};\mathbf{Y})]$
 1: Compute:
 2: $[\mathsf{Prob}(x_{i-1}, y_j)] = \frac{[z_{ij}]}{\sum_{i=1}^{2}\sum_{j=1}^{m}[z_{ij}]}$
 3: $[\mathsf{Prob}(x_{i-1})] = \frac{\sum_{j=1}^{m}[z_{ij}]}{\sum_{i=1}^{2}\sum_{j=1}^{m}[z_{ij}]}$
 4: $[\mathsf{Prob}(x_{i-1}|y_j)] = \frac{[z_{1j}]}{[z_{ij}]+[z_{2j}]}$
 5: $(i = 1, 2)$
 6: Compute $[I(\mathbf{X};\mathbf{Y})] = \sum_{i=0}^{1}\sum_{j=1}^{m} \mathsf{Prob}(x_i, y_j) \times (TL([\mathsf{Prob}(x_i|y_j)]) - TL([\mathsf{Prob}(x_i)]))$
 7: Compute $[SU(\mathbf{X};\mathbf{Y})] = \frac{2\times[I(\mathbf{X},\mathbf{Y})]}{[S(\mathbf{X})]\times[S(\mathbf{Y})]}$
 8: Compute $[R(\mathbf{X};\mathbf{Y})] = [w_0(\mathbf{X})] + [w_0(\mathbf{X})] \times [SU(\mathbf{X};\mathbf{Y})]$
```

by the interaction weight factors between the selected feature and the rest features. As shown in **Algorithm 4**, to compute the interaction weight factor between $\mathbf{X_1}$ and $\mathbf{X_2}$, $\mathcal{CS}$ first calculates the feature distributed matrix $[\mathbf{F_{11}}]$,$[\mathbf{F_{10}}]$,$[\mathbf{F_{01}}]$,$[\mathbf{F_{00}}]$ by the Hadamard production of the feature matrix or its inverted matrix. The Hadamard product is a binary operation that takes in two matrices of the same dimensions and returns a matrix of the multiplied corresponding elements. For example, for two matrices $\mathbf{A}$ and $\mathbf{B}$ of the same dimension, $(\mathbf{A} \odot \mathbf{B})_{ij} = \mathbf{A}_{ij} \times \mathbf{B}_{ij}$.

Therefore, for example, in matrix $\mathbf{F_{01}}$, the value $f_{i1} = 1$ means the i th sample has $\mathbf{X_1} = 0$, and $\mathbf{X_2} = 1$. Then $\mathcal{CS}$ computes four feature-label matrixs by $\mathbf{G} = \mathbf{F}^T \times \mathbf{Y}$. For example, elements g_{1j} in $\mathbf{G_{01}}$ means the counts of samples that $x_1 = 0$ and $x_2 = 1$, and the class of the is j. Finally, the interaction weight factor $IW(\mathbf{X_1};\mathbf{X_2})$ can be computed and the weight of the rest features would be updated as follows:

$$[w_1(\mathbf{X_{rest}})] = [w_0(\mathbf{X_{rest}})] \times [IW(\mathbf{X_{rest}};\mathbf{X_{selected}})].$$

4) Round Iteration: After updating the weight of the rest features, the relevance measure R in the second round is

$$[R(\mathbf{X};\mathbf{Y})] = w_1(\mathbf{X}) + w_1(\mathbf{X}) \times (SU(\mathbf{X};\mathbf{Y})).$$

Then, the remaining feature with the largest $R(\mathbf{X};\mathbf{Y})$ will be selected in the second round. Then the $w_1(\mathbf{X})$ of the rest features will be updated again based on the interaction weight factor between the selected feature in round two and the rest features. The algorithm will be iterated until selecting several relevant features. The details of the proposed privacy-preserving feature selection protocol are shown in Protocol 1.

V. Security Discussions

In this section, we analyze the security of our protocol. Our security analysis follows the real world/ideal world paradigm: in the real world, data owners and the cloud server interact according to the protocol specification; while in the ideal world, they follow our proposed protocol. The executions in both worlds are coordinated by the environment ***Env***. We will show that the real-world distribution is computationally indistinguishable from the ideal-world distribution.

Authorized licensed use limited to: University of Cincinnati. Downloaded on January 23,2025 at 01:44:42 UTC from IEEE Xplore. Restrictions apply.

Algorithm 4 Secure Weight Updating

```
Input: $[\mathbf{X_1}]$, $[\mathbf{X_2}]$, $[\mathbf{Y}]$, $[S(\mathbf{X_1})]$, $[S(\mathbf{X_2})]$
Output: $[IW(\mathbf{X_1}, \mathbf{X_2})]$
1: Compute feature distributed matrix:
2: $[\mathbf{F_{11}}]$=$[\mathbf{X_1}]\odot$ $[\mathbf{X_2}]$
3: $[\mathbf{F_{10}}]$=$[\mathbf{X_1}]\odot$ $[1-\mathbf{X_2}]$
4: $[\mathbf{F_{01}}]$=$[1-\mathbf{X_1}]\odot$ $[\mathbf{X_2}]$
5: $[\mathbf{F_{00}}]$=$[1-\mathbf{X_1}]\odot$ $[1-\mathbf{X_2}]$
6: Compute feature-label matrix $[\mathbf{G_{ik}}]$=$[\mathbf{F_{ik}}]^T\times[\mathbf{Y}]$
7: $(i=0,1;k=0,1)$
8: for $j=0,1,\ldots,m$ do
9:     Compute:
10:    $[\mathsf{Prob}(x_{1i},x_{2k}|y_j)]$=$\frac{[g_{ik}]_{1j}}{\sum_{i=0}^{1}\sum_{k=0}^{1}[g_{ik}]_{1j}}$
11:    $[\mathsf{Prob}(x_{1i},x_{2k},y_j)]$=$\frac{[g_{ik}]_{1j}}{\sum_{i=0}^{1}\sum_{k=0}^{1}\sum_{j=1}^{m}[g_{ik}]_{1j}}$
12:    $[\mathsf{Prob}(x_{1i},x_{2k})]$=$\frac{\sum_{j=1}^{m}[g_{ik}]_{1j}}{\sum_{i=0}^{1}\sum_{k=0}^{1}\sum_{j=1}^{m}[g_{ik}]_{1j}}$
13:    $(i=0,1;j=0,1)$
14: end for
15: Compute:
16: $[I(\mathbf{X_1},\mathbf{X_2};\mathbf{Y})]$=$\sum_{i=0}^{1}\sum_{k=0}^{1}\sum_{j=1}^{m}[\mathsf{Prob}(x_{1i},x_{2k},y_j)]$
    $\times(TL([\mathsf{Prob}(x_{1i},x_{2k}|y_j)])-TL([\mathsf{Prob}(x_{1i},x_{2k})]))$
17: $[I(\mathbf{X_1};\mathbf{X_2};\mathbf{Y})]$=$[I(\mathbf{X_1},\mathbf{X_2};\mathbf{Y})]$-$[(\mathbf{X_1};\mathbf{Y})]$-$[I(\mathbf{X_2};\mathbf{Y})]$
18: $[IW(\mathbf{X_1},\mathbf{X_2})]$=1+$\frac{[I(\mathbf{X_1};\mathbf{X_2};\mathbf{Y})]}{[S(\mathbf{X_1})]+[S(\mathbf{X_2})]}$
```

Protocol 1 Privacy-preserving Feature Selection Protocol

```
Input: Feature vector $\vec{D}=(d_i)_n$, Label vector $\vec{Y}=(y_j)_n$
Output: Selected feature: $\mathbf{X_{selected}}$,
    Feature interactions:$[IW(\mathbf{X},\mathbf{X_{selected}})]$
1: $\mathcal{FO}$, $\mathcal{LO}$ receive Homomorphic encryption key from $\mathcal{TA}$
2: $\mathcal{FO}$ obtains $\mathbf{X},\mathbf{A},\mathbf{B},S(\mathbf{X})=Init(\mathbf{D})$
3: $\mathcal{LO}$ obtains $\mathbf{Y},S(\mathbf{Y})=Init(\mathbf{Y})$
4: $\mathcal{FO}$ initializes $w_0(\mathbf{X})=1$;
   encrypts $\mathbf{X},\mathbf{A},\mathbf{B},S(\mathbf{X}),w_0(\mathbf{X})$ with Homomorphic encryption key locally;
   sends $[\mathbf{X}]$, $[\mathbf{A}]$, $[\mathbf{B}]$, $[S(\mathbf{X})]$, $[w_0(\mathbf{X})]$ to $\mathcal{CS}$
5: $\mathcal{LO}$ encrypts $\mathbf{Y},S(\mathbf{Y})$ with Homomorphic key;
   sends ciphertext $[\mathbf{Y}],[S(\mathbf{Y})]$ to $\mathcal{CS}$
6: $\mathcal{CS}$ computes $[\mathbf{Z}]=[\mathbf{B}]^T\times[\mathbf{Y}]$;
   obtains $[R(\mathbf{X};\mathbf{Y})]$ by Algorithm 3.
7: $\mathcal{CS}$ sends $[R(\mathbf{X};\mathbf{Y})]$ to $\mathcal{LO}$
8: $\mathcal{LO}$ decrypts $[R(\mathbf{X};\mathbf{Y})]$
   selects the feature with the largest $R(\mathbf{X};\mathbf{Y})$
   sends $\mathbf{X_{selected}}$ to $\mathcal{CS}$
9: $\mathcal{CS}$ computes $[IW(\mathbf{X},\mathbf{X_{selected}})]$ by Algorithm 4
   updates $[w_1(\mathbf{X})]=[w_0(\mathbf{X})]\times[IW(\mathbf{X};\mathbf{X_{selected}})]$
```

A. Privacy preservation in feature interactions

In our protocol, we assume $\mathcal{CS}$ is semi-honest and we use homomorphic Encryption(HE) to protect data transmission. In the real world, $\mathcal{CS}$ receives $[\mathbf{X}],[\mathbf{A}],[\mathbf{B}],[S(\mathbf{X})],[w_0(\mathbf{X})]$ from $\mathcal{FO}$, and received $[\mathbf{Y}],[S(\mathbf{Y})]$ from $\mathcal{LO}$ in the section IV-B1. In the ideal world, $\mathcal{FO}$ encrypts the zero-element matrix $\mathbf{X}'$, $\mathbf{A}',\mathbf{B}'$, as well as the zero-value $S'(\mathbf{X}),w_0(\mathbf{X})'$. $\mathcal{LO}$ encrypts the zero-element matrix $\mathbf{Y}'$ and the zero-value $S'(\mathbf{Y})$. Then $\mathcal{FO}$ and $\mathcal{LO}$ sends the ciphertext to $\mathcal{CS}$. According to the security of HE, the output distribution of ***Env*** in the real world and the ideal world are not computationally indistinguishable, and the transmitted message will not be leaked to $\mathcal{CS}$.

However, there are two Technical issues when $\mathcal{CS}$ trying to compute the symmetrical uncertainty SU for every feature in Section IV-B2 and compute the interaction weight factor for two features in SectionIV-B3.

- The first one is that homomorphic Encryption does not allow any operations on the encrypted data to recover the logarithm calculation on raw data. This is because $\mathcal{CS}$ needs to calculate the logarithm when computing the symmetrical uncertainty and the interaction weight factor in IWFS. However, in our protocol, $\mathcal{CS}$ can only compute some basic operations on the data when obtaining the encrypted dataset. It is essential to find out how to calculate the logarithm when given encrypted datasets.
- The second one is that $\mathcal{CS}$ can not compute the probabilities under the ciphertext. This is because $\mathcal{CS}$ needs to calculate the probabilities when computing the symmetrical uncertainty and the interaction weight factor in IWFS. In IWFS, $\mathcal{CS}$ can compute the probabilities easily by counting the number of every situation. However, when $\mathcal{CS}$ obtains the encrypted dataset, it can not compute the probabilities by counting directly because it does not know the raw dataset. Therefore, we need to find how to compute the probabilities when given encrypted datasets.

In our protocol:

- To solve the first issue, we explore the Taylor Series to solve the calculation of the logarithm in our protocol. From the Taylor Series, we know that when $(0<x<2)$:

  $$ln(x)=(x-1)-\frac{(x-1)^2}{2}+\ldots+(-1)^{n+1}\frac{(x-1)^n}{n}.$$

  We can approximate the value of $log(x)$ by the first two terms of the Taylor Series as follows:

  $$log(x)=\frac{ln(x)}{ln(10)}\approx 0.4\times(x-1)-\frac{0.4\times(x-1)^2}{2}.$$

  When $\mathcal{CS}$ obtains an encrypted dataset, it can still compute the result, and the dataset owners can decrypt the result to obtain the logarithm.
- We solve the second issue by the distributed matrix. Distributed matrix reflects how the feature data and label data are distributed in the raw dataset. Data in the distributed matrix means the number of samples in the corresponding situations. It means $\mathcal{CS}$ can use the distributed matrix to compute the probabilities of every situation directly. In this way, after obtaining the encrypted distributed matrix by multiplying two encrypted matrices, $\mathcal{CS}$ can compute the result that is decrypted to obtain the probabilities.

Our proposed protocol securely selects features based on feature interactions for VFL in the semi-honest attack model.

Authorized licensed use limited to: University of Cincinnati. Downloaded on January 23,2025 at 01:44:42 UTC from IEEE Xplore. Restrictions apply.

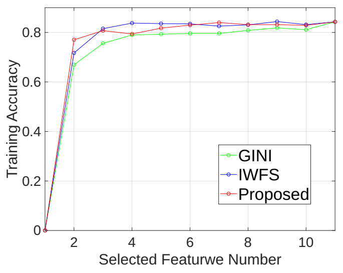

Fig. 2. Training accuracy comparison

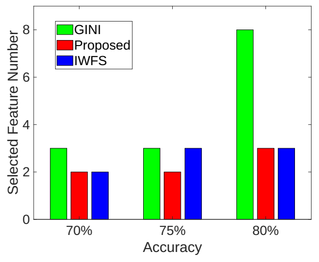

Fig. 3. Selected feature comparison

VI. Performance Evaluation

A. Simulation Setup

We perform experiments on a real-world dataset with the K-Nearest Neighbor (KNN) classification algorithm to analyze the accuracy and efficiency of our protocol. The dataset *white wine-quality* has 4898 instances and 11 raw features. We test all the cases where we select different numbers of features and compare our protocol to another two feature selection methods: IWFS (interaction-based feature selection) and GINI (non-interaction-based feature selection).

B. Simulation Results

Figure 2 shows the accuracy gains after feature selection. Our protocol achieves the same accuracy as the IWFS and better accuracy than the GINI. The results validate that the proposed protocol leverages feature interaction for proper selection and gains higher accuracy than non-interaction-based feature selection methods. In Figure 3, we compare the number of selected features to maintain the same accuracy. When achieving the same accuracy such as $70\%, 75\%$, and 80%, our proposed protocol selects 33% to 62% fewer features than the non-interaction-based feature selection. Fewer features indicate the computational and storage cost reduction.

C. Cost Analysis

We analyze the computational costs of our proposed protocol during initialization, feature selection, and weight updating. Similar to [1], we primarily consider the cost of three major operations: (1) encryption Enc, (2) decryption Dec, and (3) ciphertext multiplication Mul. These operations cost more computations, compared with other operations, such as adding two ciphertexts. Table I and II summarize the computational costs of the proposed protocol and GINI method [1] respectively, from the perspectives of feature owner $\mathcal{FO}$, the label owner $\mathcal{LO}$, and the cloud server $\mathcal{CS}$. Here, n, t, m denote the sample number, feature numbers, and all possible label numbers, respectively. Although our proposed protocol consumes more computational cost than the GINI when selecting features, this sacrificed cost finally increases the model training efficiency substantially due to the fewer features selected by the proposed protocol.

VII. Conclusion

In this paper, we have proposed a privacy-preserving interaction-based feature selection for VFL. In addition to considering feature interactions for feature selection, the proposed protocol can preserve the data owner's privacy during the feature selection. We have also provided detailed security discussions to validate our proposed privacy-preserving protocol. The simulation results demonstrate that our protocol achieves higher accuracy than the non-interaction-based feature selection when selecting the same number of features. In future work, we will further improve the efficiency of secure feature interaction calculation.

TABLE I
Computational cost of proposed protocol

Processes	$\mathcal{FO}$	$\mathcal{LO}$	$\mathcal{CS}$
Initialization	$(n^2+3n+1)\ Enc$	$(nm+1)\ Enc$	-
Feature Selection	-	$t\ Dec$	$(2mn+6m+3)\ Mul$
Weight Updating	-	-	$(4mn+4n+12m+1)\ Mul$
Total	$(n^2+3n+1)\ Enc$	$(nm+t+1)\ Enc$	$(6mn+18m+4n+4)\ Mul$

TABLE II
Computational cost of Gini method

Processes	$\mathcal{FO}$	$\mathcal{LO}$
Initialization	-	$mn\ Enc$
Secure Computation	$(2mnt+2m)\ Mul$	$2mt\ Dec + 2m\ Mul$

References

[1] R. Zhang, H. Li, M. Hao *et al.*, "Secure feature selection for vertical federated learning in ehealth systems," in *Proc. of IEEE ICC*, 2022, pp. 1257–1262.

[2] C. Yu, S. Shen, S. Wang *et al.*, "Efficient multi-layer stochastic gradient descent algorithm for federated learning in e-health," in *Proc. of IEEE ICC*, 2022, pp. 1263–1268.

[3] X. Li, R. Dowsley, and M. De Cock, "Privacy-preserving feature selection with secure multiparty computation," in *International Conference on Machine Learning*. PMLR, 2021, pp. 6326–6336.

[4] X. Ye, H. Li, A. Imakura *et al.*, "Distributed collaborative feature selection based on intermediate representation," in *IJCAI*, 2019, pp. 4142–4149.

[5] J. Yang and Y. Li, "Differentially private feature selection," in *2014 International Joint Conference on Neural Networks (IJCNN)*, 2014, pp. 4182–4189.

[6] Z. Zeng, H. Zhang, R. Zhang *et al.*, "A novel feature selection method considering feature interaction," *Pattern Recognition*, vol. 48, no. 8, pp. 2656–2666, 2015.

[7] P. Zhou, P. Li, S. Zhao *et al.*, "Feature interaction for streaming feature selection," *IEEE Trans. on Neural Networks and Learning Systems*, vol. 32, no. 10, pp. 4691–4702, 2021.

[8] R. Battiti, "Using mutual information for selecting features in supervised neural net learning," *IEEE Trans. on Neural Networks*, vol. 5, no. 4, pp. 537–550, 1994.

[9] A. Li, H. Peng, L. Zhang *et al.*, "Fedsdg-fs: Efficient and secure feature selection for vertical federated learning," in *IEEE INFOCOM*, 2023, pp. 1–10.

[10] S. Shen, C. Yu, K. Zhang *et al.*, "Exploiting feature interactions for malicious website detection with overhead-accuracy tradeoff," in *Proc. of IEEE ICC*, 2021, pp. 1–7.

[11] N. K. Anuar, A. A. Bakar, A. R. Ahmad, S. Yussof, F. A. Rahim, R. Ramli, and R. Ismail, "Privacy preserving features selection for data mining using machine learning algorithms," in *Proc. of ICIMU*, 2020, pp. 108–113.

[12] T. Zhang, T. Zhu, P. Xiong *et al.*, "Correlated differential privacy: Feature selection in machine learning," *IEEE Trans. on Industrial Informatics*, vol. 16, no. 3, pp. 2115–2124, 2020.

[13] Z. Brakerski, C. Gentry, and V. Vaikuntanathan, "(leveled) fully homomorphic encryption without bootstrapping," *ACM Trans. Comput. Theory*, vol. 6, no. 3, pp. 1–36, 2014.

Authorized licensed use limited to: University of Cincinnati. Downloaded on January 23,2025 at 01:44:42 UTC from IEEE Xplore. Restrictions apply.