VFLens: Co-design the Modeling Process for Efficient Vertical Federated Learning via Visualization

Yun Tian

School of Information Science and Technology, ShanghaiTech University Shanghai, China tianyun@shanghaitech.edu.cn He Wang

School of Information Science and Technology, ShanghaiTech University Shanghai, China wanghe1@shanghaitech.edu.cn Laixin Xie

School of Information Science and Technology, ShanghaiTech University Shanghai, China xielx@shanghaitech.edu.cn

Xiaojuan Ma

Quan Li* School of Information Science and

Technology, ShanghaiTech University

Department of Computer Science and Engineering, The Hong Kong University of Science and Technology Hong Kong, China mxj@cse.ust.hk

ABSTRACT

As a decentralized training approach, federated learning enables multiple organizations to jointly train a model without exposing their private data. This work investigates vertical federated learning (VFL) to address scenarios where collaborating organizations have the same set of users but with different features, and only one party holds the labels. While VFL shows good performance, practitioners often face uncertainty when preparing non-transparent, internal/external features and samples for the VFL training phase. Moreover, to balance the prediction accuracy and the resource consumption of model inference, practitioners require to know which subset of prediction instances is genuinely needed to invoke the VFL model for inference. To this end, we co-design the VFL modeling process by proposing an interactive real-time visualization system, VFLens, to help practitioners with feature engineering, sample selection, and inference. A usage scenario, a quantitative experiment, and expert feedback suggest that VFLens helps practitioners boost VFL efficiency at a lower cost with sufficient confidence.

CCS CONCEPTS

• Human-centered computing \rightarrow Visualization; Human computer interaction (HCI); Interaction design.

KEYWORDS

Federated Learning, Visual Analytics, Feature Interpretation, Sample Selection

Chinese CHI 2022, October 22–23, 2022, Guangzhou, China and Online, China

© 2022 Association for Computing Machinery.

ACM ISBN 978-1-4503-9869-5/22/10...\$15.00

https://doi.org/10.1145/3565698.3565765

Shanghai, China liquan@shanghaitech.edu.cn ACM Reference Format:

Yun Tian, He Wang, Laixin Xie, Xiaojuan Ma, and Quan Li. 2022. VFLens: Co-design the Modeling Process for Efficient Vertical Federated Learning via Visualization. In *The Tenth International Symposium of Chinese CHI (Chinese CHI 2022), October 22–23, 2022, Guangzhou, China and Online, China.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3565698.3565765

1 INTRODUCTION

There is hope that all industries will benefit from big data-driven artificial intelligence (AI), especially after the huge success of AlphaGo. However, with the exception of a few industries, most fields lack sufficient data or have data quality issues to support the construction of a reliable and robust machine learning (ML) model. At the same time, companies are reluctant to share or aggregate their valuable data in a centralized manner due to industry competition and privacy and security concerns, leaving the data often existing as a set of isolated data silos. As a viable decentralized solution that can potentially break down barriers between data sources while preserving privacy and security, federated learning (FL) enables users to collaboratively learn an ML model while keeping all data that may contain private information on their local device [5, 52]. Depending on how the data is partitioned between parties and application scenarios, FL can be divided into two main categories, namely horizontal FL (HFL) and vertical FL (VFL) [52]. The focus of this study is VFL, also known as feature-based FL. VFL can be applied to situations where two datasets have considerable overlap in sample IDs but differ in feature space [11]. A typical example of VFL is a collaboration between an e-commerce retail company and a financial institution in the same city. Their customer set may contain the majority of residents in the area; therefore, the intersection of their customer spaces is huge. However, since the financial institution records its customers' income, spending behavior, and credit rating, while the e-commerce retailer retains its customers' browsing and purchasing history, their feature spaces are quite different. In this case, VFL allows both parties to train a joint ML model for product purchase prediction based on customer and product information under privacy-preserving security conditions (Figure 1).

^{*}The corresponding author.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.



Virtually Aggregated Training & Testing Dataset

(a) A Local financial institution

(b) An e-commerce retail company

Figure 1: Suppose two parties, i.e., a local financial institution (a) and an e-commerce retail company (b) want to co-build a ML model for product purchase prediction. Only the financial institution has the label Y: loan or not and neither party wants to expose their features X. The two parties has an overlapping sample IDs, i.e., user id:1 – 4. The target is to establish a joint model under the condition of protecting privacy and the effect of the joint model is better than that of unilateral data modeling.

Although VFL has shown good performance in scenarios such as financial risk management [13, 51, 60], healthcare [44], and ecommerce ad recommendation [50], real world practitioners have encountered the following challenges when trying to use VFL for their application domains [24]: 1) Uncertainty in sample selection for training. Traditional VFL practitioners mainly utilize two methods to prepare for the VFL training phase. First, when there is not much available data with labels, they may utilize all the overlapping samples with labels to train a joint VFL model for convenience. However, it is well known that the training speed of FL is much slower than that of the local model due to the design of data encryption and communication mechanisms. In some cases, utilizing all overlapping samples with labels for VFL model training can lead to much longer training time. Second, when the amount of data available for training is large, they may select some labeled data samples for training and evaluate the model based on general metrics such as accuracy, loss, and mAP. As a slang expression in classical ML terminology, "garbage in, garbage out" [20] indicates the samples used for prediction should have a high-quality match with their specific jointly trained VFL models. Although the role of interactive data iteration in ML is emphasized and domain experts acknowledge that "data samples need to have appropriate signals for the model to be useful" [20], there is little support for their fine-tuning of training data samples in VFL scenarios. Both practices rely heavily on feedback from VFL model performance for further evaluation, which is sometimes too time-consuming and expensive. In particular, things get worse when the communication of the VFL training process is not so stable, as domain experts have to repeatedly re-upgrade the model training phase by trial and error. Therefore, an intuitive sample data evolution interaction mechanism that allows domain experts to compare the data characteristics and performance of different sample training datasets in a VFL scenario is necessary. 2) Non-transparent feature selection and assessment. Successful ML applications require an

iterative process to create models that provide the desired performance. One of the key processes involves feature engineering and in this study, we focus on feature selection and assessment. However, unlike traditional centralized ML modeling or HFL in which all data features are available and easy to assess, in VFL, participants update only their internal feature parameters during training, and external features from other parties are not visible to them due to the design of privacy-preserving mechanisms, which poses unique challenges for internal/external feature selection and assessment. That is, in addition to selecting the necessary internal features or transforming the original internal features into other powerful alternatives, practitioners are exploring how to assess external features from other parties [39, 45]. However, the lack of comprehensive consideration of the contribution of internal and external features while protecting privacy still undermines the use of VFL in production. 3) Costly and time-consuming inference. The inference phase of VFL modeling requires online coordination between two (or more) parties to accomplish the inference task, which inevitably poses a challenge to computational resources and raises costs. According to our collaborating domain experts, the cost and deployment efficiency of federated modeling are issues that require rational planning for practical applications, and the use of homomorphic encryption in VFL can lead to a significant reduction in the computational speed and information transfer speed of federated modeling compared to centralized ML modeling [22]. To solve this problem, in addition to optimizing the computational modeling process, another intuitive approach is to reduce the overall data volume. That is, not all samples to be predicted need to be truly predicted with the help of external features from other parties. For example, those samples in which practitioners have relatively high confidence in their labels do not need to be predicted by invoking an online trained VFL model because, e.g., the sample features are poor, and these samples can be safely ignored. Thus, how to visually help domain experts distinguish samples with different

confidence in their labels is a desirable capability for real-world VFL deployments.

In this study, we co-design the modeling process to help VFL practitioners improve the efficiency of VFL modeling from the perspective of visualization. We first conduct an observational study of the current practices of collaborating domain experts to identify their main needs and concerns regarding VFL applications. Then, we streamline the analysis pipeline of feature and sample spaces and propose an interactive visualization system called *VFLens. VFLens* helps domain experts to interactively participate in feature selection, assessment, and sample data iteration processes before the VFL model training phase, in feature interpretation after the VFL model inference phase. A case study and expert feedback confirm the efficacy of *VFLens.* Our main contributions are summarized below.

- We describe the problem in the VFL context from the perspective of feature and sample space through an observational study and in-depth discussions of design requirements with VFL domain experts.
- We co-design the VFL modeling process to support domain experts to interactively participate in the data iteration, feature selection and assessment, and sample prediction processes. To the best of our knowledge, *VFLens* is the first such effort in the VFL scenario.
- We evaluate *VFLens* through a usage scenario, a quantitative experiment and expert interviews.

2 RELATED WORK

The literature that overlaps this work can be categorized into four groups, namely, *federated learning*, *visualizations for federated learning*, *feature selection and assessment*, and *sample selection in machine learning*.

2.1 Federated Learning

Federated learning was first proposed by Google, which prevents data from being transmitted by distributing model training to each mobile terminal [5]. Later, they released the first commercial FL application, GBoard [17], which uses a recursive neural language model to predict the next word in a keyboard application. GBoard allows each local mobile device to train the model using local data from the same distributed ML model. The global model can be updated by averaging the model parameters collected over all local models. Along the same lines, many studies have reshaped different ML models into a federated framework, including decision trees [31, 58], linear/logistic regression [32, 36], and neural networks [46, 56]. These works are categorized as HFL because the clients share the same feature space but differ in the sample space. Unlike HFL, VFL is applicable to scenarios where we have many overlapping instances but few overlapping features [51]. For example, an insurance company and an online retailer in a local city have many overlapping users, but each has its own feature space. VFL "merges" features and uses homomorphic encryption to protect the data privacy of the participating parties, and requires a more sophisticated mechanism to decompose the loss function of each party. This study focuses on VFL, "virtually aggregation" of different features to compute training losses and gradients in

a privacy-preserving manner, and jointly build an ML model [11] with data from both parties.

2.2 Visualizations for Federated Learning

Researchers from academia and industry are using visualizations to demonstrate, explain, and monitor the process of federated learning. For example, in industry, Lenovo has simulated the industrial revolution in factories by demonstrating the process of horizontal federated learning to predict the internal pressure of hardware [38]. Similarly, Cloudera Fast Forward Labs released an interactive simulation prototype, Turbofan Tycoon, which takes advantage of visualization to examine the federated model and predict when a turbofan will fail [35]. FATEBoard¹ utilizes dashboard visualizations to display modeling logs, metrics, and evaluation results, including information on data sets, job status, computational plots, and model output [12]. While FATEBoard can help domain experts understand the ranking of features and the performance of models, it does not support detailed and interactive inspection of the sample and feature spaces. On the other hand, in academia, Wei et al. [47] developed a game to demonstrate the superiority of HFL and built a visualization prototype to help understand the operation of HFL. However, this work assumes that client-side data can be witnessed by the server-side. Li et al. [30] proposed HFLens, which strictly follows a data privacy-preserving design and supports comparative visual interpretation at the overview, communication round, and client instance levels. HFLens facilitates the investigation of the overall HFL process involving all clients, the correlation analysis of client information in one or different communication rounds, the identification of potential anomalies, and the evaluation of the contribution of each HFL client. However, the pain point for VFL is not the anomaly detection like HFLens, because for VFL there are generally not as many data collaborators as for HFL, and the collaborators partnerships with common interests. In this work, we do not focus on the operational process of FL, but rather improve the efficiency of VFL modeling by involving domain experts in the sample and feature space.

2.3 Feature Selection and Assessment

There is a large amount of existing work related to feature selection [4, 6], which has two main difficulties. First, a large number of features are used in the process of building machine learning models; however, if several features are linearly correlated with each other, many of them will be redundant, which adds additional computational effort and leads to more complex parameters. Second, common feature analysis methods use feature correlation metrics, but correlation metrics cannot measure nonlinear relationships. Isabelle et al. [15] performed a survey of automatic feature selection methods. The authors abstracted the core problem of feature selection, which is to find a minimal subset of features from a large number of features. The authors also argued that there are many options for feature selection and that there is no one universal and unique solution. There are other types of feature selection methods, such as wrappers [25], which iteratively eliminate features by regression or classification models to find the ideal subset of features. There are also metric-based methods [2, 14], where users pick the

¹https://fate.fedai.org/

top k best features. However, they also suffer from the problems described earlier. As for feature assessment, it will be different in VFL modeling and traditional centralized machine learning modeling. In VFL, Host B does not have direct access to the features of Guest A, so in practice, the feature importance of Guest A is obtained by encrypting the IV values [8, 9, 57] (Host B and Guest A will be introduced in subsection 4.1). In traditional centralized machine learning modeling, there are many methods to calculate feature importance, such as impurity-based feature importance [41] and permutation feature importance [1]. In this study, we implement several different types of alternative feature selection techniques for choosing the internal features of Host B. We allow the user to decide whether the feature ranking is desirable or whether to focus on one of the features, while for the external features of Guest A, we use the encrypted IV values [8, 9, 57] to obtain the feature importance metrics.

2.4 Sample Selection in Machine Learning

As one of the most critical infrastructures for building AI systems [16], data has a significant impact on the performance, fairness, robustness, and scalability of AI systems. However, data is often the "the least motivated aspect, considered 'operational" relative to the lionized work in building new models and algorithms [16, 34, 40]. Akrong et al. [40] reported on data practices of high-risk AI by interviewing AI practitioners around the world. They identified compound events of adverse and downstream effects caused by data problems named data cascades. Yee et al. [53] proposed Faceted browsing that allows users to use metadata to extract subsets of data that share desired attributes. The rank-by-feature framework allows users to examine low-dimensional projections of multidimensional data based on their statistics [42]. Hohman et al. [20] proposed CHAMELEON, which allows users to compare data features, training/test splits, and performance of multiple data versions. Facets² helps developers examine ML datasets, including training/test segmentation, observe feature shapes, and explore individual observations. For data selection in FL, recent studies select relevant data distributively based on a benchmark model prior to training, regardless of other data quality factors or batch composition during training. Li et al. [26] provided a systematic analysis of the underlying data factors that affect FL model performance and propose an overall design to privately and efficiently select high quality data samples. However, the focus of these studies is on HFL. Inspired by their work, we emphasize the importance of training data in VFL and propose several interactive visualization schemes to facilitate sample selection prior to training of VFL models. In VFL inference, we separate the samples to be predicted based on the different confidence levels of their labels. To the best of our knowledge, VFLens is the first attempt in this regard.

3 OBSERVATIONAL STUDY

3.1 Background

To understand the application of VFL in practice, we worked with a team of domain experts from a collaborating local financial and AI organization, including a FL project manager (E1, male, age: 31), a

VFL researcher (E2, male, age: 33), two VFL engineers (E3, male, age: 27, E4, male, age: 28), and one business contact (E5, female, age: 29). A large part of their work is to provide federated learning (both HFL and VFL) solutions to clients to meet their specific business needs. They shared with us a recent encounter in which they designed and developed a precision marketing strategy in a real estate scenario. Notably, the real estate company wanted to leverage the features of other parties through VFL to jointly train an ML model to predict whether a particular customer would come to visit the real estate sales office and understand the customer's characteristics. By taking this jointly trained VFL model, the real estate company can find more suitable customers from a large pool of other customers who may visit the sales office. An illustrative pipeline for this case is shown in Figure 2.

In this case of the approval and start-up phases of VFL, the experts encountered several problems. First, the domain experts had to identify training samples from an overlapping set of users between the real estate (host) and the external data provider (guest, i.e., an online e-commercial and financial company). Although the total amount of available training samples has reached about 25,000, not all of the samples are good enough. Most records were collected from field sales representatives and describe a rough profile of customer characteristics such as "gender", "age", "career", "income level", "marital status", and "family structure". E1 commented that "there may be some outliers in the training samples because sometimes salepeople cannot guarantee the veracity of all characteristics." E3 said that "we are not experts in real estate," so he did not have a clear idea about how to identify the right sample of customers for model training. Besides the local feature aspect, the limited computational resources and communication bandwidth between the host and the guest is another issue in the modeling process. In other words, dumping all overlapping training samples to train the VFL model is unrealistic because it may consume a lot of communication resources. Considering the characteristics of training samples and resources, domain experts would like to get some intuitive tips on how to select suitable and sufficient training samples. Second, when discussing how model knowledge can be used to guide their offline marketing strategies when contacting potential customers, E1 and E5 said, "we should at least know what the model has learned so that we can understand the characteristics of customers who are likely to visit our real estate sales office." Third, when it comes to inference and prediction using the trained VFL model, the initial expectation of the business requirement was that predictions needed to be made for all contacts in the real estate pool to obtain the likelihood that they would visit the sales office. However, the reality is that the pool is quite large, i.e., about 100, 000 and each prediction requires a paid call to the online VFL model. "If the model predicts all the customers in the pool, the budget may not cover this cost." said E4, "can we just predict those customers who need the help of the VFL model without running all of them?"

3.2 Requirement Analysis

To ensure that our approach was in line with the tasks and requirements, we interviewed all experts (E1 – E5) to identify their main concerns about improving VFL modeling efficiency and have summarized their requirements below.

²https://pair-code.github.io/facets/



Figure 2: A case the experts encountered: designing and developing a precision marketing strategy in a real estate scenario. (1) The real estate company provides training samples from one sales office and (2) the samples are divided into two parts: those who have visited the sales office and those who have no interest in visiting the sales office. (3) The real estate company (host) jointly trains a VFL model with the data from an external data provider (guest). (4) The real estate company predicts whether a particular customer will visit the sales office from its large customer pool. (5) The real estate company learns the characteristics of its potential buying customers to guide offline telemarketing.

R.1 Evaluate the quality of the training samples from the host. The first pressing problem that experts encounter when building VFL models is to prepare sufficiently good training samples. While previous studies have proposed various methods to support data iteration for better model training [20, 40, 42, 53], there is little support for this in VFL. E1 and E5 had little work experience in ML and, given the specific business requirements in the VFL scenario, they felt that their domain knowledge could be useful in selecting training samples. Therefore, both technologists (E3 and E4) and business personnel (E1 and E5) wanted to assess the quality of samples used by the host for model training in an intuitive and interactive way.

R.2 Understand the internal/external features of hosts and guests. Although the well-established automatic feature selection techniques allow analysts to confirm the contribution of each feature to the final prediction, especially in datasets with many features [7], these techniques may produce significantly inconsistent results. According to E2, in a typical VFL scenario, the feature space is distributed in two (or multiple) parties. Thus, understanding internal/external features consists of two stages: 1) comparing alternative feature selection techniques based on their ranking of all internal features of the host, and 2) simulating external features of different batches of guests. Notably, the experts indicated that they would like answers to the following questions: "which internal features are consistently ranked high?"; "How much does the technology vary in terms of feature ranking?" Our approach should allow experts to respond to such queries.

R.3 Compare performance between models. Inspired by recent studies that use similarities between model representations to correct for local training of the parties, such as conducting contrastive learning in model-level [28] or comparing differences between the global HFL model and the local model [30], E3 and E4 wished to understand the differences between each locally trained model and the global VFL model. For example, using standard validation metrics such as *accuracy, loss, Kolmogorov-Smirnov (KS), Area under the curve (AUC)*, and *mean Average Precision (mAP)* to understand performance fluctuations. In addition, experts wanted

to have an overview of the history of the operations they performed so that they could identify "*critical points that might correspond to model performance improvements*" [28]. Therefore, it is desirable to have an intuitive representation of the model performance per attempt and the performance differences between models.

R.4 Obtain VFL forecasts at low cost. The most critical business requirement raised by E1 and E5 is the conflict between the large number of forecasting requirements and the limited monetary budget and computational/communication resources. They commented, "usually, the pool has a large number of samples to forecast, but blindly feeding all of them into the joint model will inevitably result in a waste of time and money." Therefore, given the limited resources, experts need a strategy to balance the prediction sample size with a good enough prediction accuracy.

R.5 Checking the prediction results. As mentioned earlier, since our method makes a trade-off between the size of the VFL prediction sample and the prediction accuracy, experts are curious about the effectiveness of our strategy. Therefore, we should perform a comparative evaluation of the prediction results of our strategy. This evaluation will allow domain experts, especially E5, to understand the efficacy of our method and the reasons why those samples with relatively high confidence in the labels do not need to call the online trained VFL model for prediction.

4 CO-DESIGN THE MODELING PROCESS OF VFL

Inspired by the observational study, we propose a co-design process for VFL modeling, as shown in Figure 3. Our approach, named *VFLens*, consists of an LR-based back-end VFL configuration module and a front-end visualization module. Specifically, the back-end module collects the necessary logs from the embedded VFL model and processes the information required for feature and data sample selection. The output of the back-end engine module is fed to the front-end visualization module for further analysis. The back-end engine module also receives interactive commands from the user through the front-end visualization module during the feature and sample selection.

Chinese CHI 2022, October 22-23, 2022, Guangzhou, China and Online, China

Yun Tian, He Wang, Laixin Xie, Xiaojuan Ma, and Quan Li



Figure 3: Overview of our approach. We divide the pipeline of our approach into two phases, namely modeling stage and inference stage. In the modeling stage, we first configure the backend using *LR-based vertical federated learning*. Features distributed on two parties, i.e., host and guest, are selected and fine-tuned in *Features for Modeling* and *Feature Selection View*. After determining the features, we select the appropriate samples for modeling via *Sample Selection View*. Then, we fine-tune the local model via *Model Performance View* until we are satisfied and run the VFL model using the identified features and the samples initialized by the host side. Note that the local model can be fine-tuned several times. During the inference phase, all instances used for inference are compared and sampled via *Inference View*. Finally, in *Inference View*, samples with low confidence are fed into the VFL model for prediction; otherwise, samples with high confidence are provided to the local model for prediction.

4.1 VFL Architecture

In this subsection, we illustrate the general architecture and basic background knowledge of a VFL system. According to the definition proposed by Yang et al. [52], VFL is suitable for scenarios with many overlapping instances but few overlapping features. For example, suppose two companies, A and B, want to jointly train a machine learning model with their business data (i.e., they have different feature spaces). In addition, B has the label data that the model needs to predict. In this case, B is considered as the Host and A as the Guest. Due to data privacy and security issues, the two companies cannot directly exchange their business data for training. To ensure data confidentiality during model training, an honest third party, usually played by an authority or a secure computing node, is introduced and participates without colluding with either party, i.e., Collaborator C. Both parties (A, B) are honest, but curious about each other's data. It is worth noting that a VFL training process usually consists of the two following phases as shown in Figure 4(a), i.e., Encrypted Entity Alignment and Encrypted Model Training [52]. In the Encrypted Entity Alignment phase, VFL utilizes an encryption-based user ID alignment technique called Private Set Intersection (PSI) [21], based on some encryption techniques such as MD5 [37] and Secure Hash Algorithm-1 (SHA-1) [55] to identify common users who overlap on both sides without exposing their respective data. Note that the VFL system does not tell nonoverlapping users during the encrypted entity alignment process. Regarding the Encrypted Model Training stage, we adopt logistic regression (LR)-based VFL [18, 49] to showcase our approach. Other federated privacy-preserving ML algorithms such as secure linear regression [52] and SecureBoost [11] can also be quickly adopted or replaced with our back-end VFL solution. Since the initiator of

VFL is the host, our proposed co-design process for VFL modeling (i.e., *VFLens*) is oriented to the host side.

4.2 LR-based VFL Configuration

We utilize LR and stochastic gradient descent in cooperation with an additive homomorphic encryption scheme and mask [18, 49]. The phase of encryption model training starts after identifying overlapping entities common to both parties, which are used to train the ML model. Notably, our goal is to have both parties, i.e., the host and the guest, compute the intermediate results of the gradient separately as much as possible, and then get their gradient results through the interaction of encrypted information. As shown in Figure 4(b) (1 - 4), we divide the computational task as follows. That is, in each round of parameter update, each party needs to perform the following computations and interactions in turn and Step 1 - 4 are repeated until the model converges.

Step 1: Guest A and Host B initialize their parameters, and Collaborator C generates a key pair and distributes the public key to A and B.

Step 2: Guest A computes its part of the gradient, encrypts with the public key, and sends it to Host B. Host B calculates its own amount of the gradient, encrypts it with the public key, and sends it to Guest A. All exchanged information is homomorphically encrypted, so parameters can be computed like in non-encrypted procedures, but are not visible. After receiving the corresponding parts, both parties compute their respective parts of the gradient separately.

Step 3: Both parties send the encrypted part of the gradient to Collaborator C for decryption, but to prevent Collaborator C from getting the gradient directly, Guest A and Host B add a random



Figure 4: Architecture for a typical VFL system.

mask to the gradient part and send it to Collaborator C. Thus, the gradient obtained by Collaborator C cannot be used directly.

Step 4: Collaborator C gets the two parts of the encryption gradient, decrypts them, and returns them to Guest A and Host B, respectively. Then, Guest A and Host B subtract the previously added random mask to get the actual gradient and update their parameters.

4.3 VFL Modeling Phase

In this subsection, we first describe the general process of the VFL modeling phase. Then, we describe how *VFLens* supports domain experts to co-design the feature and sample space for VFL modeling.

4.3.1 General Process of VFL Modeling. To train the VFL model, first, both host and guest collide de-identifiable³ sample users id to determine the user intersection set. In the modeling phase, we decide which samples from the intersection will be used as training and testing samples (subsubsection 4.3.4 Samples for Modeling), where the training samples are used to train the VFL model and the testing samples are used to validate the model. Next, both parties train a "semi-model" using the features and labels of the samples. In addition to the samples, the model training also requires the determination of sample features (subsubsection 4.3.2 Features for Modeling) and sample performance (sample labels). Specifically, the guest provides a certain amount of modeling sample features (Featurequest) for VFL model training, and completes its "semi-model" training in its local environment. The host provides labels of the modeling samples for VFL model training, but it is not necessary to provide the features (Featurehost) of the modeling samples for VFL model training. Based on feedback from domain experts, "host does not necessarily have the features of the modeling samples, but must have the labels of the modeling samples." Without loss of generality, we assume that the host also has some features of the modeling samples, i.e., in our case, the host has both some but limited features and all the labels of the modeling samples. Thus, the host can also train a "semi-model" locally by using these features and labels.

4.3.2 *Features for Modeling.* As mentioned earlier, we assume that the host holds user data that can be processed as features and labels. The host then needs to transform the user samples used for modeling into user features and user labels in order to complete the host's "semi-model" training locally. Therefore, how to properly co-design the host samples with features is of great interest to federated learning practitioners.

For internal features, we explore the feature importance-based host feature space [7] using the following five representative automatic feature selection techniques. There are three methods for Univariate Feature Selection [23]. 1) The first one uses the method of ANOVA F-value test to select k best features. To avoid automatic feature removal, we always set the value of k to the maximum value of all features retained and let the domain experts decide which features to retain. 2) The X^2 -based method uses a chi-square test to select the k best features. 3) The mutual information method estimates the mutual information of discrete target variables, with higher values implying stronger dependencies. 4) Impurity-based Feature Importance [41]. This is related to the intrinsic nature of the ensemble algorithm, i.e., outputting feature importance after training. Therefore, we derive the feature importance from the best model found so far. 5) Permutation Feature Importance [1]. This is a technique for monitoring the reduction of model scores when individual feature values are randomly shuffled. To put them in the same context for comparison, we normalize the output from 0 to 1. We also calculate their average value to represent the feature performance on an average basis.

For external features, since they are invisible to the host, the data requester does not have access to the data information of the training participants in advance. The simplest way is that if the host wants to perform VFL jointly, they need to try all combinations of external features. This would waste a lot of training time. Existing work usually utilize feature bucketing [8, 9, 57] for external feature engineering and secures the data to measure the importance of external features. In this work, we also utilize the cryptographic communication method of feature bucketing to compute the information values (iv) of external features.

³A process that removes personal identity and makes it impossible to identify or associate the subject of the personal information without additional information.



Figure 5: *VFLens* interface: (a) A data loader for selecting cases of interest; (b) A feature selection view embeds five automatic feature selection methods. Users can select a certain number of local features and external "invisible" features by considering the internal feature importance and balancing the cost of VFL modeling. (c) The sample selection view shows the statistics information of the dataset and 2D embedding projection of all training samples. Users can select a particular data cluster in (c1) for selection and observe the detailed feature distribution of each cluster in (c2). (c3) An interactive scheme to select samples from the center to the periphery of a cluster. (c4) The quality of the training data set is evaluated by two metrics, *homogeneity* and *diversity*. (d1) Parameter selection for modeling. (d2) The model performance view for training the local and VFL model. (e) The summary view presents and visualizes the results of the classification results of all the samples to be predicted with our strategy.



4.3.3 Feature Selection View. Based on the above internal feature selection for modeling, we design a feature selection view Figure 6(a) that supports domain experts to interactively select essential features from internal and external features based on feature importance (**R2**). The view is designed using a heatmap [7], where the horizontal axis refers to different feature selection methods and the vertical axis represents different features. The color depth of

the heatmap represents the importance of the features, and the penultimate column is the average of the first five methods, which serves as a reference for our selection. The last column can be interactively toggled to select and decide which local feature we finally choose for modeling. For external feature selection, due to the privacy mechanism of federated feature modeling, domain experts only have access to the importance of federated features and the anonymous ID of external feature. Therefore, *VFLens* uses the depth of the color to encode the information values (iv) in the VFL feature selection view Figure 5(b).

We initially considered an alternative design solution displayed as a radial stacked bar chart Figure 6(b). We accumulate the scores of the different automatic feature selection methods in a stacked plot, which can directly display the most critical feature. In addition, thin lines connect the features, and their thickness represents the correlation between the two corresponding features. However, this initial design was not accepted by our collaboration experts when comparing the differences between the various feature selection

methods. In addition, we need to consider the contribution of features from the external party. Therefore, we finally chose the first design.



Figure 7: Illustration of interactive sampling stage.

4.3.4 Samples for Modeling. In addition to assessing the contribution of features, another concern of VFL practitioners is to prepare suitable samples for the VFL modeling process. By selecting relevant samples, the labels of the samples can be potentially balanced (**R1**) and the diversity of sample features can be maintained, which can potentially improve model performance [33]. To this end, we propose an interactive sampling and evaluation procedure, which consists of two phases, *projection and clustering* and *interactive sampling for each cluster*.

Projection and Clustering. We utilize *t-SNE* as a dimensionality reduction technique because it is particularly good at conveying meaningful insights about the data, such as clusters and outliers [29]. Nevertheless, it may not be feasible to transform all avaiable samples in the intersection user set (i.e., the samples for training, validation, and prediction) into a two-dimensional representation based on the data features residing in the host and depict them on a two-dimensional space, as the data size may be huge. To deal with this potential problem, we abstract the raw training data into several clusters using *KMeans++* [3], which augments *KMeans* with a random and direct seeding technique that greatly improves its speed. We determine the number of clusters using the Elbow method [43].

Interactive Sampling. Sampling is a simple but practically meaningful method to greatly reduce the data while maintaining certain data properties. We propose an interactive sampling scheme for each cluster. The basic principle is to drop most of the internal samples and keep the boundary samples to ensure strong learning and generalization [33]. Users can interactively control the sampling ratio for a given data interval by adjusting the sampling slider in Figure 5(c3). Figure 7 is a schematic of the sample data around the cluster, and the dashed circles are intervals divided from the origin outward, centered on the center of the cluster. The intuitive idea is that some nodes at the center of the cluster are more concentrated and representative, while sample data that are off-center tend to have some noise. We divide each cluster into intervals of n from the center to the circle margin as $[0:\sqrt{r}], [\sqrt{r}:\sqrt{2r}], \ldots, [\sqrt{(n-2)r}:\sqrt{(n-1)r}]$ and $\left[\sqrt{(n-1)r} : +\infty\right]$ with a constant *r* just as Figure 7 (in this work, we define the value of *n* as 11, *r* as 0.5 after several experiments). This sampling scheme ensures that each individual region within the cluster occupies the same area. We sample the data according to a polar coordinates design because of the clustering mechanism

of *KMeans*, which clusters the samples according to the distance between the samples and the centroids. After completing the sampling of each cluster, we merge the sampled instances to the data pool.

Data Sampling Evaluation. We utilize *target label balance* and *feature diversity* as core metrics to help evaluate the combined data samples after interactive sampling. Regarding the target label of the training data, we find that the performance of the model deteriorates significantly as the imbalance increases [27]. When the classification distributions of the datasets are almost identical, the reduction in content diversity will lead to a considerable loss of accuracy. Thus, larger homogeneity and content diversity are likely to lead to better model performance. These two metrics are defined as follows. (1) *Statistical Homogeneity*. Let \mathcal{Y} is the set of target categories. Cluster C_k has a dataset $\mathcal{D}_k = \{(x_k, y_k)\}$, where each data x_k has a label $y_k.\mathcal{D}_k$ follows a categorical distribution q_k . The uniform categorical distribution over \mathcal{Y} is q_u . The statistical homogeneity of \mathcal{D}_k is defined as [59],

$$\mu_k = 2 - \sqrt{\sum_{y \in \mathcal{Y}} |q_k (y_k = y) - q_u (y_u = y)|^2},$$
(1)

measuring the similarity between distributions q_k and q_u over \mathcal{Y} . (2) *Content Diversity.* Given a dataset \mathcal{D} having M samples or M sub-collections of samples and let v_i be the flattened features vector of the i^{th} sample or i^{th} subcollection of samples, the similarity function of two vectors is $S(v_i, v_j)$. The content diversity of \mathcal{D} is defined as [48],

$$\rho(\mathcal{D}) = 1 - \frac{\sum_{i,j \in [M], i \neq j} 2S\left(v_i, v_j\right)}{M(M-1)}$$
(2)

4.3.5 Sample Selection View. We design a sample selection view to facilitate the mentioned interactive sampling for modeling. As shown in Figure 5(c1), after projecting the clusters into a 2D space, a key issue is to convey the data characteristics of each cluster in terms of label distribution and salient features to facilitate the exploration of the data. Notably, we design a novel glyph to represent the data characteristic of each cluster. It consists of an inner radar chart and an outer ring. The data features are arranged in the clockwise direction of the glyph. The values on each axis of the radar chart represent the distance between the probability distribution of the data in this dimension of the cluster and the probability distribution of the entire training data, calculated using EMD distance (i.e., Wasserstein distance) [59]. The outer ring shows the label distribution, with yellow representing 1 and the blue representing 0, e.g., in the case of binary classification. When users click on a glyph in Figure 5(c1), VFLens displays its data features in Figure 5(c2) as a reference for the subsequent sampling operation in Figure 5(c3). It is worth noting that users can conduct an interactive sampling procedure for each cluster by filtering any number of samples from the center to the subcontinent (R4). In general, the principle of interactive sampling is to balance the data labels to diversify and enrich the characteristics of the samples. Therefore, we evaluate the overall data after sampling in Figure 5(c4), involving sample count, homogeneity, diversity.

4.3.6 VFL/Local Modeling and Model Verification. After identifying the features and samples to be modeled, we come to the VFL modeling stage. Multiple rounds of communication are required between host and guest to exchange encrypted gradients, and both parties transmit the following information to each other: 1) encrypted gradient information, i.e., the derivation of weights derived from sample features and labels; 2) de-identifiable sample ID, which are used to align the gradient information and transmit the convergence direction of the model. This process is a typical VFL, and the specific information interaction and communication process can be referred to the LR-based VFL configuration. Users can train the model by pressing the buttons in Figure 5(d1). After VFL modeling, a VFL model is built at this point. In addition, since the host contains certain features and full sample labels, we can train a local model at the host side.

After building the initial VFL model, both parties verify the accuracy of the VFL model using de-identifiable user ID, features, and labels of the verification user set. For ease of understanding, we describe the verification process as follows: 1) The host transmits the de-identified ID of the verification user set to the guest to verify whether the guest holds the data of the validation user set. If not, the verification process ends. 2) If the guest holds the verification user set, both parties start invoking their own "semi-models" to compute the corresponding prediction score. Specifically, the guest obtains the features of the verification user set from its local data and inputs them into the guest's semi-model to calculate the prediction score. Similarly, since the host has certain sample features, it can also calculate its prediction scores by feeding the sample features into the host's semi-model. Admittedly, in some cases, the host may have only sample labels without any feature dimensions, so there is no need to compute its prediction score. 3) After computing the prediction scores for both parties, the guest transmits the prediction score to the host, who aggregates the prediction score of both parties to obtain the final prediction result. 4) Finally, the host compares the prediction results with the ground truth labels and decides whether to fine-tune the model.

4.3.7 Model Performance View. In the model performance view, we display the results of the metrics used to evaluate the performance of the local and VFL models for each configuration (**R3**). We can iterate over the local model multiple times, since all communication and modeling processes occur locally. However, for the VFL model, we should iterate carefully if good enough performance is achieved.. Notably, as shown in Figure 5(d2), this view shows the traditional metrics used to evaluate binary or multiclass classifications, such as *accuracy*, *loss*, *KS*, and *AUC*. Each iteration of the local and VFL models is recorded to facilitate the selection of the best model for the final model prediction.

4.4 Inference Stage

In the inference stage, the host can initiate a prediction based on its real-world requirement and invoke the "semi-model" of both parties for inference. Generally, the "semi-model" of both parties will give a prediction score, and the guest will send its prediction score to the host. The host will aggregate the prediction scores of both parties to get the final prediction result. Specifically, the vertical federated learning inference process is as follows: 1) the host de-identifies the target user ID to be predicted and transmits the de-identified IDs to the guest. After receiving the de-identified IDs, the guest queries whether it holds these corresponding features of these IDs. If the guest does not have these features, the prediction ends; 2) If the guest contains the features of these IDs, both parties invoke their respective "semi-models" to calculate the prediction scores. For example, the guest reaches the target user data locally, obtains the features of the target IDs, and then inputs the guest's "semi-model" to obtain the prediction scores. Similarly, if the host holds some user features, the host computes its prediction scores for the target user IDs; otherwise, if the host does not have user features, it does not need to compute the prediction scores; 3) The guest transmits the prediction scores of its "semi-model" to the host, and the host aggregates the two parts to obtain the final prediction result. At this point, the VFL inference stage is over and all samples that need to be predicted will get the prediction result by VFL inference.

4.4.1 Instances for Inference and Sample Selection. In reality, the number of samples to be predicted can be very large. In the inference phase of VFL, "semi-models" from both parties are required to cooperate online to complete the prediction, which poses a great challenge in terms of communication quality, modeling resources and economic costs. According to the expert feedback, VFL can improve the confidence of prediction results by expanding the feature space, but "this is not suitable for all samples to be predicted." It is worth noting that with limited budget cost, E1 and E5 want a less costly method to accomplish prediction for all samples to be predicted, while guaranteeing the quality of the prediction results to some extent. Therefore, we need to classify the samples to be predicted and send only those samples with high uncertainty to the VFL model for inference. And for the samples with high label confidence, we can use the local model of the host for prediction (R4).

For this purpose, we design the following sample selection schemes in the inference stage: 1) The sample ps_i to be predicted is input into the local model of the host for inference, and the prediction result is obtained. Note that the local model here is different from the "semi-model" of the host. The local model only considers the data features residing on the host side and the data labels. 2) In the sample space, we compute the similarity between ps_i and all training samples in terms of the host feature space, e.g., using the Euclidean distance to obtain the training sample ps_i that is closest to ts_i . 3) For ts_i , determine its ground truth label, the local model's prediction result and the prediction results of the VFL model are consistent. We consider ps_i a plausible sample if and only if the results of the three parts agree and are the same as the prediction results of the host local model for ps_i , where the prediction results of the host local model can be directly used as the final prediction results; otherwise, ps_i is considered to have high uncertainty and needs to be predicted by the VFL model. At this time, all the samples to be predicted will be automatically divided into two categories. One category is samples with high credibility that do not require federated prediction, and the other category is samples that require further inference using the VFL model.

4.4.2 Inference View. As mentioned before, we obtained the two classes of samples to be predicted. To visualize the results and present the current classification of our strategy (R4 - R5), we compute a many-to-many mapping based on the shortest distance from the predicted data to the training samples. In particular, as

shown in Figure 5(e), we use the *t-SNE* and *KMeans++* clustering methods to abstract the large number of samples to be predicted into the Sankey diagram. The flow from left to right in the Sankey diagram is the number of samples to be predicted. The leftmost column shows the eight label combinations of the training samples, corresponding to three cases (i.e., ground truth label, local model prediction result, and VFL model prediction results). The middle two columns correspond to the training data and the samples to be predicted, respectively. The last column lists the number of samples with high reliability and the number of samples that need to be predicted by the VFL model.

4.5 Interaction Among the Views

Rich interactions are integrated into *VFLens* to build a low-cost VFL model. *1) Sorting*. After the user clicks the average button in the feature selection view, the system will sort the average importance of all local features from highest to lowest. In the VFL feature selection view, we encode the importance of VFL feature as the color depth of the color block, and the color block color will change after the specified feature is selected. *2) Hover on*. In the sample selection view, when the user selects a data cluster, a zoomed-in view of the cluster content is presented at the edge of the view, which helps the user to identify the specific feature patterns of different clusters. *3) Parameter editing*. In the sample view, we adjust the scale of the samples in different regions by mediating the height of the bars with the mouse.

5 EVALUATION

5.1 Usage Scenario

The modeling data from the real estate industry mentioned in the observational study has confidentially rules and cannot be used as experimental data. As an alternative, we take a publicly available credit card dataset [54], where financial institutions can use their own customer data to predict various customer performance. Although most customer data are limited in the finance context, if other information can be used for joint modeling, financial institutions can obtain more accurate prediction models to further provide better services and reduce potential banking risks. To model this scenario, we use the dataset of customer defaults in Taiwan [54]. Notably, we use the first 14 dimensions as internal features and the last 9 dimensions as the federated external features. We also derive several alternative features. Finally, we obtain 20 local features and 33 external features for VFL modeling. In VFL, the host knows everything about the local features. Nevertheless, due to the privacy-preserving mechanism, we only use IDs to represent the federated external features. We use 15,000 rows of total data in the VFL modeling stage and 15,000 rows in the VFL inference stage.

Feature Space Interaction. After loading the local data into *VFLens*, we select the top 13 local features based on the recommended results of the standard automatic feature selection methods, as shown in Figure 8(1). Regarding external features, to simulate reality, we randomly select 15 remote external features for VFL modeling.

Sample Space Interaction. After the initial determination of the internal and external features, the host must select the appropriate number of samples for VFL model training. We turn to the

sample selection view and first see a sharp feature in the upper left corner of the radar chart of cluster No.4. Based on the radar chart visualization, we find a similar distribution of features in clusters No.1, No.8, No. 15, and No.11 next to cluster 4. We then click on cluster No.4 and further observe the specific indicator of the radar chart enlarged on the right side of the view Figure 8(3A) and its feature distribution in Figure 8(4). We find that the feature distribution of this cluster has firm consistency. Therefore, we decide to reduce the overall sampling rate of these clusters to reduce the number of samples within these clusters for training, and to reduce the training cost while maintaining the loss of data features. More specifically, we first select cluster No. 4, and in Figure 8(4), we determine that the salient feature of this cluster is *limit bal log*. The other features are similar to those of the overall samples. In Figure 8(5), we reduce the number of duplicate training samples by reducing the sampling rate of the central region samples. Then we click the "Start Sampling" button of cluster No.4. The relevant information after sampling is displayed in Figure 8(6). We can observe that Homogeneity is 1.58, which reflects that the balance of data labels is moderate. The metric of Content Diversity is 0.10, which indicates that the current sample features are relatively concentrated with a low degree of diversity. Following a similar procedure, we sample cluster No.1, cluster No.8, cluster No.15, and cluster No.11 at a lower sampling rate. The metric of Homogeneity at this point becomes 1.58, while Content Diversity becomes 0.12. We can see that the balance of the current samples is not significantly improved, since most of the target labels of these clusters are 0. Inspired by this, we select clusters with more target labels of 1, such as cluster No.6, cluster No.10, cluster No.14, and cluster No.16 in Figure 8(3B). We first sample cluster No.6 and No.10 in appropriate proportions and observe that Homogeneity changes to 1.77 with significant improvement, while Content Diversity (i.e., 0.21), which also shows a huge improvement. However, we are still not satisfied with the value of Content Diversity. We continue to operate on the cluster No.13 and No.5 in Figure 8(3C) because they have more target labels. Finally, the Homogeneity and Content Diversity become 1.77 and 0.64, satisfying our modeling requirements.

Modeling Training and Result Analysis. Since the features and samples for training are ready, we come to the training phase, set the learning rate to 0.1, and the number of iterations to 2, 000 in Figure 8(7), and then perform VFL modeling. As shown in Figure 8(8), we can see that the results of the training model are updated in real time for each iteration. Finally, the ACC of the local model is 0.68, and the AUC is 0.60. The vertical federated model has an ACC of 0.75 and an AUC of 0.69. The results show that our trained VFL model achieves better performance than the host's local model. However, the current AUC is still below 0.7. We follow the above procedure to further improve the model performance, such as adding more external features required by the host and sampling other clusters for model training. Finally, the ACC of the local model reaches 0.78 and its AUC reaches 0.64, while the ACC of the VFL model is 0.79 and its AUC is 0.75.

Model Inference. After the model training, we then come to the model inference stage by clicking the "start predict" button. We observe that 6798 records require further VFL inference, and the remaining 8202 samples do not require calling the VFL model Figure 8(9) (i.e., the rejection rate is 54.6%). To evaluate the efficacy



Figure 8: Usage Scenario: (1) Select the top-ranked local features in the local feature selection view. (2) External features with high iv values are selected in the VFL feature selection view. (3) In the sample selection view, we select different clusters for sampling based on the proportional of label in the cluster and the specific feature distribution information encoded in the clusters. (4) Observe the specific feature distribution information of the data in the cluster. (5) Set the sampling weight for different regions in each cluster. (6) Observe the two statistical indicators of the overall data that has been sampled and we will stop sampling when the amount of data and the two statistical indicators reach the basic expectation value. (7) Adjust the learning parameters and start model training. (8) Run the local inference model, check the proportion of samples to be predicted, and verify the validity of the model.

of our interactive sampling strategy, we run the VFL model on the samples that do not need to invoke the VFL model to obtain their labels generated by the VFL model. We find that the results of federated inference for 7782 samples are the same as the those of the local model (i.e., a hit rate of 94%), indicating that our interactive feature and sample selection strategy helps improve the efficacy of VFL modeling. As a comparison, we also use the VFL model to predict samples that require further VFL inference and find that the VFL model results for the 4707 samples are the same as those of the local model. Compared to 94% hit rate for the unnecessary samples, 69.2% hit rate suggests that a larger proportion of samples is not confident in their labels. For these samples, we need to fine-tune more good features and train a better prediction model.

5.2 Quantitative Experiment

We also conduct quantitative experiments to compare the effect of *VFLens* with other cases that do not use interactive feature selection and sample selection. As shown in Table 1, we can analyze that using only local data for modeling (Exp.1), the training will be very fast because no cryptographic operations are needed. However, the accuracy of the model will be very low due to the poor features. In Exp.2, all external features provided by the data provider are selected for federated modeling to obtain a better model. However, in real business scenarios, the cost spent on federated modeling is proportional to the sample size involved in the modeling process. So there is no need to select the entire aligned data for federated

modeling. From the perspective of data reduction and reducing unnecessary costs, the corresponding (Exp.3 – 4) is about *VFLens*'s ability to use local feature selection and federated feature selection to eliminate some redundant features and also to reduce the tie of fine-tuning the model by domain experts. In Exp.5, we further use *VFLens* for sample selection based on Exp.4, and we can see that the accuracy of the model is not significantly reduced, demonstrating the practical value of *VFLens* in business scenarios.

6 DISCUSSION AND LIMITATION

We conduct semi-structured interviews with all experts (E1 – E5) to assess the efficacy of *VFLens* and to determine whether our approach could help them improve the efficiency of VFL modeling.

System Performance. All experts agreed that *VFLens* demonstrates a straightforward data science process with an intuitive visual design and practical implications. "*The system is very useful for hosts to assess the quality of their data and to further evaluate how much resources are needed for VFL modeling,*" said E1. Our system has the potential to help a variety of commercial enterprises and sectors build economically usable VFL models. Considering the balance between resource input and model accuracy, the results seem to be quite good. *VFLens* is the first system that attempts to address realistic pain points across business sectors, which will go a long way to "*enrich and complement existing federated learning frameworks*". In discussing with our collaborating experts what inspired them most about our system, E2 said that they were most

Experiment id	Local feature	FL feature	Sample number	AUC	ACC	Time
1	All(20/20)	None(0/30)	10000	0.68	0.70	< 1 min
2	All(20/20)	All(30/30)	10000	0.78	0.82	> 1 day
3	Part(13/20)	All(30/30)	10000	0.78	0.80	> 12 hours
4	Part(13/20)	Part(21/30)	10000	0.77	0.80	> 6 hours
5	Part(13/20)	Part(21/30)	2100	0.75	0.79	2 hours

Table 1: Training time(s) of local model and federated learning model on Default Credits Prediction data in different situations.

impressed with the fine-tuning and interactive sampling of local data features based on dimensionality reduction results, which "*fits our sense and it was very valuable.*"

Visual Design and Learnability. We draw from observational studies of experts working routinely in real-world scenario to inform the system design and classicial interfaces used in commercial federated learning products (e.g., *FateBoard* in *FATE* [12]). For example, the matrix design visually presents the feature importance provided by different automatic feature selection algorithms. The model performance represented by various metrics and the sample distribution represented by the histogram design are familiar visual metaphors in traditional data science pipelines and analysis. Experts commented, "we can quickly get used to visual encoding because the interactions are the same as in traditional modeling." Specifically, after a briefly introduction to each view and its capabilities, the experts developed customized exploration paths for interactive VFL modeling and inference.

Generalizability and Scalability. In this work, we only show our pipeline using LR-based VFL. Tree-based models such as *XG*-*Boost* [10] and *Random Forest* [19] are more commonly used in various commercial domains. Incorporating such models into *VFLens* is not a difficult task. Nevertheless, the visualization design of the front-end may require unavoidable alternations, as the underlying models may introduce the necessary specific information to be checked. Notably, if other types of models or even deep learning models are used, the deployment and implementation of VFL versions of the models and a detailed discussion of model performance comparisons are major concerns for future work.

Contributions Over Previous Work. Compared to previous works, our system has inspired experts in many ways. They commented that when they utilize VFL in real-world scenarios, they often experience poor data quality or lack of sufficient data attributes when a large enterprise group consists of different affiliated business units that are unable to share data. Some business units even lack data labels. All of these real-world problems pose a huge challenge for VFL modeling. Experts said that if they want to leverage external data sources from the Guest using VFL, VFLens can greatly help them sort out the current data quality before modeling. With an understanding of overall data and feature quality, experts can better understand whether they need to call on external data sources for VFL modeling. "VFLens could help business units in companies assess data quality, such as data fill rate, label fill rate," said E1, adding that this needs to be done before FL modeling. Also, VFLens can help experts understand data samples and features before modeling, which has high application value in real-world scenarios.

Limitation. Our work has several limitations. First, interactive samplings of predicted clusters relies heavily on user's experience

and domain knowledge, introducing uncertainty in sample selection. We should provide more intelligent guideline to help users perform interactive sampling more efficiently and confidently. Second, we only utilize representative metrics such as *accuracy* and *AUC* to compare different versions of the model. We did not consider using the internal information of the models themselves, such as gradient distribution and weight information, which may be more critical when employing tree-based or deep learning models.

7 CONCLUSION AND FUTURE WORK

In this study, we present *VFLens*, a visual analytics system for interactive VFL modeling and inference, which improves VFL modeling efficiency by supporting domain experts to co-design internal features and interactively manipulate sample spaces. A usage scenario, a quantitative experiment, and expert feedback confirm the efficacy of *VFLens*. In the future, we will combine more models with the FL version to enable experts to achieve better prediction results. Also, we will introduce more sampling methods to reduce the impact of anomalous samples on training and further discuss the design space when involving other types of models, such as tree-based or deep learning-based models.

ACKNOWLEDGMENTS

We are grateful for the valuable feedback and comments provided by the anonymous reviewers. This work is partially supported by the Research Start-up Fund of ShanghaiTech University and HKUST-WeBank Joint Laboratory Project Grant No.: WEB19EG01-d.

REFERENCES

- André Altmann, Laura Toloşi, Oliver Sander, and Thomas Lengauer. 2010. Permutation importance: a corrected feature importance measure. *Bioinformatics* 26, 10 (2010), 1340–1347.
- [2] Yindalon Aphinyanaphongs, Lawrence D Fu, Zhiguo Li, Eric R Peskin, Efstratios Efstathiadis, Constantin F Aliferis, and Alexander Statnikov. 2014. A comprehensive empirical comparison of modern supervised classification and feature selection methods for text categorization. Journal of the Association for Information Science and Technology 65, 10 (2014), 1964–1987.
- [3] David Arthur and Sergei Vassilvitskii. 2006. k-means++: The advantages of careful seeding. Technical Report. Stanford.
- [4] Avrim L Blum and Pat Langley. 1997. Selection of relevant features and examples in machine learning. Artificial intelligence 97, 1-2 (1997), 245–271.
- [5] H. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. 2017. Communication-efficient learning of deep networks from decentralized data. In Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS 2017. arXiv:1602.05629. https://arxiv.org/pdf/ 1602.05629.pdf
- [6] Girish Chandrashekar and Ferat Sahin. 2014. A survey on feature selection methods. Computers & Electrical Engineering 40, 1 (2014), 16–28.
- [7] Angelos Chatzimparmpas, Rafael M Martins, Kostiantyn Kucher, and Andreas Kerren. 2021. FeatureEnVi: Visual Analytics for Feature Engineering Using Stepwise Selection and Semi-Automatic Extraction Approaches. arXiv preprint arXiv:2103.14539 (2021).

Chinese CHI 2022, October 22-23, 2022, Guangzhou, China and Online, China

Yun Tian, He Wang, Laixin Xie, Xiaojuan Ma, and Quan Li

- [8] Hao Chen, Zhicong Huang, Kim Laine, and Peter Rindal. 2018. Labeled PSI from fully homomorphic encryption with malicious security. In Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security. 1223–1237.
- [9] Hao Chen, Kim Laine, and Peter Rindal. 2017. Fast private set intersection from homomorphic encryption. In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security. 1243–1255.
- [10] Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining. 785–794.
- [11] Kewei Cheng, Tao Fan, Yilun Jin, Yang Liu, Tianjian Chen, Dimitrios Papadopoulos, and Qiang Yang. 2021. Secureboost: A lossless federated learning framework. *IEEE Intelligent Systems* (2021).
- [12] Tao Fan. 2018. FATE-Board_FATE's Visualization Toolkit., 11 pages. https: //github.com/FederatedAI/FATE-Board
- Fedai. 2019. Computer vision Platform powered by Federated Learning. https://www.fedai.org/cases/computer-vision-platform-powered-byfederated-learning/
- [14] George Forman et al. 2003. An extensive empirical study of feature selection metrics for text classification. J. Mach. Learn. Res. 3, Mar (2003), 1289–1305.
- [15] Isabelle Guyon and André Elisseeff. 2003. An introduction to variable and feature selection. *Journal of machine learning research* 3, Mar (2003), 1157–1182.
 [16] Alon Halevy, Peter Norvig, and Fernando Pereira. 2009. The unreasonable
- effectiveness of data. *IEEE Intelligent Systems* 24, 2 (2009), 8–12.
- [17] Andrew Hard, Kanishka Rao, Rajiv Mathews, Swaroop Ramaswamy, Françoise Beaufays, Sean Augenstein, Hubert Eichner, Chloé Kiddon, and Daniel Ramage. 2018. Federated Learning for Mobile Keyboard Prediction. (2018). arXiv:1811.03604 http://arxiv.org/abs/1811.03604
- [18] Stephen Hardy, Wilko Henecka, Hamish Ivey-Law, Richard Nock, Giorgio Patrini, Guillaume Smith, and Brian Thorne. 2017. Private federated learning on vertically partitioned data via entity resolution and additively homomorphic encryption. arXiv preprint arXiv:1711.10677 (2017).
- [19] Tin Kam Ho. 1995. Random decision forests. In Proceedings of 3rd international conference on document analysis and recognition, Vol. 1. IEEE, 278–282.
- [20] Fred Hohman, Kanit Wongsuphasawat, Mary Beth Kery, and Kayur Patel. 2020. Understanding and visualizing data iteration in machine learning. In *Proceedings* of the 2020 CHI Conference on Human Factors in Computing Systems. 1–13.
- [21] Yan Huang, David Evans, and Jonathan Katz. 2012. Private set intersection: Are garbled circuits better than custom protocols?. In NDSS.
- [22] Qinghe Jing, Weiyan Wang, Junxue Zhang, Han Tian, and Kai Chen. 2019. Quantifying the performance of federated transfer learning. arXiv preprint arXiv:1912.12795 (2019).
- [23] Alan Jović, Karla Brkić, and Nikola Bogunović. 2015. A review of feature selection methods with applications. In 2015 38th international convention on information and communication technology, electronics and microelectronics (MIPRO). Ieee, 1200–1205.
- [24] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. 2019. Advances and open problems in federated learning. arXiv preprint arXiv:1912.04977 (2019).
- [25] Ron Kohavi and George H John. 1997. Wrappers for feature subset selection. Artificial intelligence 97, 1-2 (1997), 273–324.
- [26] Anran Li, Lan Zhang, Juntao Tan, Yaxuan Qin, Junhao Wang, and Xiang-Yang Li. 2021. Sample-level Data Selection for Federated Learning. In IEEE INFOCOM 2021-IEEE Conference on Computer Communications. IEEE, 1–10.
- [27] Anran Li, Lan Zhang, Juntao Tan, Yaxuan Qin, Junhao Wang, and Xiang-Yang Li. 2021. Sample-level Data Selection for Federated Learning. In *IEEE INFOCOM* 2021 - IEEE Conference on Computer Communications. 1–10. https://doi.org/10. 1109/INFOCOM42981.2021.9488723
- [28] Qinbin Li, Bingsheng He, and Dawn Song. 2021. Model-Contrastive Federated Learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 10713–10722.
- [29] Quan Li, Kristanto Sean Njotoprawiro, Hammad Haleem, Qiaoan Chen, Chris Yi, and Xiaojuan Ma. 2018. Embeddingvis: A visual analytics approach to comparative network embedding inspection. In 2018 IEEE Conference on Visual Analytics Science and Technology (VAST). IEEE, 48–59.
- [30] Quan Li, Xiguang Wei, Huanbin Lin, Yang Liu, Tianjian Chen, and Xiaojuan Ma. 2021. Inspecting the Running Process of Horizontal Federated Learning via Visual Analytics. *IEEE Transactions on Visualization and Computer Graphics* (2021), 1-1. https://doi.org/10.1109/TVCG.2021.3074010
- [31] Qinbin Li, Zeyi Wen, and Bingsheng He. 2020. Practical federated gradient boosting decision trees. In Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 34. 4642–4649.
- [32] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. 2018. Federated optimization in heterogeneous networks. arXiv preprint arXiv:1812.06127 (2018).
- [33] Zhang Li and Guo Jun. 2006. A method for the selection of training samples based on boundary samples. *Journal of Beijing University of Posts and Telecommunications* 29, 4 (2006), 77.

- [34] Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. 2021. A survey on bias and fairness in machine learning. ACM Computing Surveys (CSUR) 54, 6 (2021), 1–35.
- [35] Mike. 2018. Federated learning: distributed machine learning with data locality and privacy. https://blog.fastforwardlabs.com/2018/11/14/federated-learning. html
- [36] Mehryar Mohri, Gary Sivek, and Ananda Theertha Suresh. 2019. Agnostic federated learning. In International Conference on Machine Learning. PMLR, 4615– 4625.
- [37] Ronald Rivest and S Dusse. 1992. The MD5 message-digest algorithm.
- [38] Marcin Rojek. 2018. Devices learning from each other ? See it live this September at AI Summit in San Francisco !, 7–11 pages.
- [39] David Roschewitz, Mary-Anne Hartley, Luca Corinzia, and Martin Jaggi. 2021. IFedAvg: Interpretable Data-Interoperability for Federated Learning. arXiv preprint arXiv:2107.06580 (2021).
- [40] Nithya Sambasivan, Shivani Kapania, Hannah Highfill, Diana Akrong, Praveen Paritosh, and Lora M Aroyo. 2021. "Everyone wants to do the model work, not the data work": Data Cascades in High-Stakes AI. In proceedings of the 2021 CHI Conference on Human Factors in Computing Systems. 1–15.
- [41] Erwan Scornet. 2020. Trees, forests, and impurity-based variable importance. arXiv preprint arXiv:2001.04295 (2020).
- [42] Jinwook Seo and Ben Shneiderman. 2005. A rank-by-feature framework for interactive exploration of multidimensional data. *Information visualization* 4, 2 (2005), 96–113.
- [43] MA Syakur, BK Khotimah, EMS Rochman, and Budi Dwi Satoto. 2018. Integration k-means clustering method and elbow method for identification of the best customer profile cluster. In *IOP conference series: materials science and engineering*, Vol. 336. IOP Publishing, 012017.
- [44] Praneeth Vepakomma, Otkrist Gupta, Tristan Swedish, and Ramesh Raskar. 2018. Split learning for health: Distributed deep learning without sharing raw patient data. arXiv:1812.00564 [cs.LG]
- [45] Guan Wang. 2019. Interpret federated learning with shapley values. arXiv preprint arXiv:1905.04519 (2019).
- [46] Hongyi Wang, Mikhail Yurochkin, Yuekai Sun, Dimitris Papailiopoulos, and Yasaman Khazaeni. 2020. Federated learning with matched averaging. arXiv preprint arXiv:2002.06440 (2020).
- [47] Xiguang Wei, Quan Li, Yang Liu, Han Yu, Tianjian Chen, and Qiang Yang. 2019. Multi-Agent Visualization for Explaining Federated Learning. In Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, [IJCAI-19]. International Joint Conferences on Artificial Intelligence Organization, 6572– 6574. https://doi.org/10.24963/ijcai.2019/960
- [48] Ting Wu, Lei Chen, Pan Hui, Chen Jason Zhang, and Weikai Li. 2015. Hear the whole story: Towards the diversity of opinion in crowdsourcing markets. *Proceedings of the VLDB Endowment* 8, 5 (2015), 485–496.
- [49] Kai Yang, Tao Fan, Tianjian Chen, Yuanming Shi, and Qiang Yang. 2019. A quasi-newton method based vertical federated learning framework for logistic regression. arXiv preprint arXiv:1912.00513 (2019).
- [50] Liu Yang, Ben Tan, Vincent W Zheng, Kai Chen, and Qiang Yang. 2020. Federated recommendation systems. In *Federated Learning*. Springer, 225–239.
- [51] Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. 2019. Federated machine learning: Concept and applications. ACM Transactions on Intelligent Systems and Technology 10, 2 (2019), 1–19. https://doi.org/10.1145/3298981 arXiv:1902.04885
- [52] Qiang Yang, Yang Liu, Yong Cheng, Yan Kang, Tianjian Chen, and Han Yu. 2019. Federated learning. Synthesis Lectures on Artificial Intelligence and Machine Learning 13, 3 (2019), 1–207.
- [53] Ka-Ping Yee, Kirsten Swearingen, Kevin Li, and Marti Hearst. 2003. Faceted metadata for image search and browsing. In Proceedings of the SIGCHI conference on Human factors in computing systems. 401–408.
- [54] I-Cheng Yeh and Che-hui Lien. 2009. The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients. *Expert Systems with Applications* 36, 2 (2009), 2473–2480.
- [55] Ching-Hung Yuen and Kwok-Wo Wong. 2011. A chaos-based joint image compression and encryption scheme using DCT and SHA-1. Applied Soft Computing 11, 8 (2011), 5092–5098.
- [56] Mikhail Yurochkin, Mayank Agarwal, Soumya Ghosh, Kristjan Greenewald, Nghia Hoang, and Yasaman Khazaeni. 2019. Bayesian nonparametric federated learning of neural networks. In *International Conference on Machine Learning*. PMLR, 7252–7261.
- [57] Chengliang Zhang, Suyi Li, Junzhe Xia, Wei Wang, Feng Yan, and Yang Liu. 2020. {BatchCrypt}: Efficient Homomorphic Encryption for {Cross-Silo} Federated Learning. In 2020 USENIX Annual Technical Conference (USENIX ATC 20). 493–506.
- [58] Lingchen Zhao, Lihao Ni, Shengshan Hu, Yaniiao Chen, Pan Zhou, Fu Xiao, and Libing Wu. 2018. Inprivate digging: Enabling tree-based distributed data mining with differential privacy. In *IEEE INFOCOM 2018-IEEE Conference on Computer Communications*. IEEE, 2087–2095.
- [59] Yue Zhao, Meng Li, Liangzhen Lai, Naveen Suda, Damon Civin, and Vikas Chandra. 2018. Federated learning with non-iid data. arXiv preprint arXiv:1806.00582 (2018).

[60] Fanglan Zheng, Kun Li, Jiang Tian, Xiaojia Xiang, et al. 2020. A vertical federated learning method for interpretable scorecard and its application in credit scoring. Chinese CHI 2022, October 22-23, 2022, Guangzhou, China and Online, China

arXiv preprint arXiv:2009.06218 (2020).