Hounding Data Diversity: Towards Participant Selection in Vertical Federated Learning

Xiaokai Zhou[†] Xiao Yan[#] Fangcheng Fu[‡] Xinyan Li[†] Hao Huang[†]
Quanqing Xu[§,✉] Chuanhui Yang[§] Bo Du[†] Tieyun Qian[†] Jiawei Jiang[†,✉]
[†] School of Computer Science, Wuhan University [‡] School of Computer Science, Peking University
[§] OceanBase, Ant Group [#] Centre for Perceptual and Interactive Intelligence
{xiaokaizhou,xinyan_li,haohuang,dubo,qty,jiawei.jiang}@whu.edu.cn ccchengff@pku.edu.cn
yanxiaosunny@gmail.com {xuquanqing.xqq,rizhao.ych}@oceanbase.com

2025 IEEE 41st International Conference on Data Engineering (ICDE) | 979-8-3315-3603-9/25/$31.00 ©2025 IEEE | DOI: 10.1109/ICDE65448.2025.00211

***Abstract*—Due to the rising concerns on privacy protection, how to build machine learning models from distributed databases with privacy guarantees has gained more popularity. Vertical federated learning (VFL) trains machine learning models in a privacy-preserving way when the data features are scattered over distributed databases. We study the *participant selection problem* (PSP) for VFL, which chooses a given number of participants to conduct training while maximizing model accuracy. Compared to training with all participants, PSP can filter out hitch-riders that contribute marginally to model quality and reduce training time by involving fewer participants. To achieve good model accuracy, we formulate PSP as choosing a set of participants that maximizes the likelihood of the data samples. Then, utilizing the k-nearest neighbors (KNN) classifier as the proxy model, we express the likelihood as a function of the selected participants and prove that the function is submodular. The submodular property is favorable as it can account for the feature diversity among the participants and allows to greedily select the participant with the maximum gain in each step. However, the selection process requires finding the top-k neighbors of a data sample as the basic operation, which is expensive in VFL setting as it involves encrypted communication. As such, we adapt the Fagin's algorithm, a famous top-k query algorithm, to reduce the amount of encrypted communication. We deploy our solution VFPS-SM across five distributed nodes and conduct experiments with 10 datasets and 3 models to evaluate its performance. The results show that VFPS-SM can reduce the end-to-end running time by up to 35×, selection time 365× and improve model accuracy by 6.0% compared with state-of-the-art baselines.**

I. Introduction

Creating powerful machine learning (ML) models requires collecting large-scale and high-quality data. However, data is often soiled across various organizations, and data sharing is typically restricted by privacy regulations such as GDPR [1]. As such, many studies have raised intensive attention to distributed data management and analysis [2]–[4]. Researchers and data scientists are interested in building ML models from distributed databases in a privacy-preserving way [5]–[10]. Motivated by this problem, *federated learning* (FL) is a distributed machine learning (ML) scheme, which enables multiple participants to train models collaboratively while ensuring data privacy. That is achieved by sharing encrypted data or intermediate computation results instead of the raw data [4], [11], [12]. According to the data distribution, FL can be classified into two main scenarios, i.e., *horizontal federated learning* (HFL) and *vertical federated learning* (VFL). In HFL, each participant holds all features of some data samples, whereas VFL involves participants holding different features of the same set of data samples. VFL has attracted much interest from the database community due to its rich applications for distributed data management regarding issues such as tree model [5], [13], [14], data privacy [7], [15]–[18] and communication efficiency [19]–[22]. In this paper, we focus on VFL. Fig. 1 shows an example of VFL, where a bank wants to collaborate with an e-commerce company and a credit company to train a model to predict the label, i.e., whether a customer is involved in financial fraud.

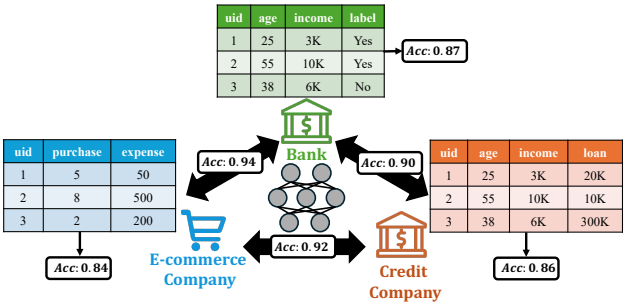

Fig. 1. An example of vertical federated learning.

Participant Selection for VFL. To safeguard privacy, typical VFL systems have to resort to expensive methods such as homomorphic encryption [14], [15], [23] and secret sharing scheme [7], [13], [24]. As a result, the scalability and efficiency of VFL systems are significantly constrained by the number of participants. Moreover, the performance of the global model in VFL largely depends on the quality of local data. Training with low-quality data hinders the model from achieving optimal performance. Therefore, identifying participants with high-quality data is crucial for efficient and effective model training. We define this challenge as the *participant selection problem* (PSP). Formally, PSP involves selecting p participants from a total of P to participate to conduct model training. PSP provides three key benefits. ❶ It can identify participants who significantly enhance model quality,

✉ Corresponding author

2375-026X/25/$31.00 ©2025 IEEE
DOI 10.1109/ICDE65448.2025.00211

protecting against hitch-riders and irrelevant participants that make little or negative contributions (e.g., because of poor data quality). ❷ By evaluating participant contributions, PSP supports a reward system that encourages essential participants to engage in VFL. ❸ PSP can also reduce VFL costs by involving fewer participants since VFL costs increase nearly linearly with more participants exchanging intermediate results and gradients during training. In Table I, we select 2 out of the 4 participants to train a logistic regression model on the SUSY dataset. The results show that the training time is accelerated by over 3× with only a slight degradation in model accuracy.

Existing Solutions and Their Problems. Typical solutions for a selection problem consist of two steps, i.e., *valuation* and *selection* [25]–[27]. For PSP, the valuation step quantifies the contributions of the participants to model accuracy, and the selection step chooses the p participants with the largest contribution scores. A natural solution is to use the Shapley value [28] for valuation due to its nice properties like additivity and fairness. Specifically, consider a set $\mathcal{P}$ with P participants, and use $U(\mathcal{S})$ to denote the utility (e.g., accuracy) of the model trained over a subset of participants $\mathcal{S} \subseteq \mathcal{P}$, the Shapley value $SV(p)$ quantifies the average marginal contribution of participant p to all possible subsets of $\mathcal{P}$. That is,

$$SV(p) = \frac{1}{P} \sum_{\mathcal{S} \subseteq \mathcal{P} \setminus p} \binom{P-1}{|\mathcal{S}|}^{-1} \left[U(\mathcal{S} \cup \{p\}) - U(\mathcal{S}) \right].$$

Using the Shapley value to solve PSP poses two challenges: ❶ It requires computing all $(2^P - 1)$ combinations for P participants, each needing individual model training, which results in a long selection time as evidenced in Table I. ❷ Selecting participants with the highest Shapley values may not optimize model accuracy because they may lack *feature diversity*. This is because participants with high-score but similar data may have redundancy. In Fig. 1, the bank and the credit company have Shapley values of 0.4 and 0.38, respectively, while the e-commerce company has a value of 0.3. Despite the bank and the credit company contributing more individually, pairing them adds limited value to the model since both primarily provide overlapping personal financial information. In contrast, pairing either the bank or the credit company with the e-commerce company, which offers diverse shopping data, enhances data diversity and improves model accuracy. Beside the toy example, Table I also shows that the model accuracy of Shapley is noticeably lower than our solution VFPS-SM. Besides, VF-MINE [27] groups participants, scores each based on mutual information, and averages these scores to assess importance. Although mutual information is cheaper to compute than the Shapley value, it still cannot consider the feature diversity among the participants, which is evidenced by its low model accuracy in Table I. The problems of existing solutions prompt our research question:

> *Can we design a solution to* PSP *that selects diverse participants for high model accuracy and conducts the selection process efficiently?*

TABLE I
TRAINING TIME AND MODEL ACCURACY FOR THE LOGISTIC REGRESSION (LR) MODEL ON THE SUSY DATASET. *All* TRAINS WITH ALL THE PARTICIPANTS, *Shapley* AND *VF-MINE* SELECT PARTICIPANTS USING THE SHAPLEY VALUE AND MUTUAL INFORMATION, RECEPTIVELY, WHILE *VFPS-SM* IS OUR PROPOSED METHOD.

	Party Count	Selection Time (s)	Training Time (s)	Total Time (s)	Test Accuracy
ALL	4	0	13503	13503	78.76%
SHAPLEY	2	136103	3881	139984	76.20%
VF-MINE	2	848	3881	4729	76.20%
VFPS-SM	2	372	3866	4238	78.19%

Our Solution VFPS-SM. To solve the research question, we propose VFPS-SM, which improves model accuracy by selecting a subset of participants that maximizes data sample likelihood. We use the KNN classifier as the proxy model because it is a classic yet simple method that aligns with many ML models in exploiting the geometric distribution of the data samples. As such, the likelihood function of the KNN classifier can be treated as a surrogate of classification accuracy. PSP is formulated as choosing the participants that maximize the KNN likelihood function. With extensive derivation, we establish that this likelihood function is *submodular* regarding the participants, a property critical for capturing participant feature diversity. Submodular functions, as demonstrated in previous studies [29]–[33], account for the diversity of a subset of items because its marginal gain from adding an item v to a group $\mathcal{G}$ diminishes as the subset size increases.

> *Let the utility of a set $\mathcal{G}$ be $f(\mathcal{G})$. The contribution of item v is $f(v|\mathcal{G}) = f(\mathcal{G} \cup v) - f(\mathcal{G})$. The utility function f is submodular if for all $\mathcal{A} \subseteq \mathcal{B} \subseteq \mathcal{G}$ and $v \notin \mathcal{B}$, it satisfies $f(v|A) \geq f(v|B)$.*

To optimize the submodular function, we use a greedy method for participant selection, which chooses the participant with the highest marginal gain until reaching a predefined size.

To safeguard data privacy, we employ homomorphic encryption when implementing the KNN classifier in the VFL setting. However, this approach introduces substantial computation and communication challenges due to many encryption operations. To improve efficiency, we incorporate the Fagin's algorithm, which efficiently merge sub-rankings from different participants, effectively reducing the required data transmissions. This strategic use of top-k query algorithms substantially reduces the encrypted computation and communication overhead, thereby streamlining the participant selection process and enhancing overall efficiency. Furthermore, we analyze the security requirements from three perspectives(i.e., feature security, label security and identity security) and discuss the privacy protection capabilities of our VFPS-SM.

We experiment with three popular ML models (i.e., KNN, LR, and Multi-Layer Perceptron (MLP)) and compare our VFPS-SM with two baselines (i.e., SHAPLEY, VFMINE). The results demonstrate that VFPS-SM significantly enhances efficiency, reducing selection time by up to 365× and cutting

end-to-end training time by as much as 33×. Additionally, VFPS-SM improves model accuracy by up to 6.0% by selecting more diverse participants. To further study the effect of participant diversity, we incrementally add participants with replicated data. VFPS-SM effectively identifies these redundancies, improving model quality, while the baselines fail to detect duplicates, leading to lower accuracy.

To summarize, we make the following contributions:

- We inspect existing vertical federated participant selection frameworks and identify significant issues related to high costs and neglect of diversity.
- We analyze the KNN classifier's likelihood function and its submodular properties, proposing a framework VFPS-SM that addresses PSP with submodular maximization.
- We optimize the KNN implementation in the VFL setting using top-k query algorithms and employ homomorphic encryption techniques to safeguard communicated data.
- We extensively evaluate VFPS-SM on various datasets. The experimental results demonstrate that VFPS-SM can efficiently and effectively select a diverse subset of participants.

II. PRELIMINARIES

In this section, we introduce the basic of vertical federated learning and submodular maximization.

A. Vertical Federated Learning

Data Layout. Consider a set of P participants: $\mathcal{P} = \{1, 2, ..., P\}$, and there is a dataset of N data samples: $\mathcal{D} = \{X, Y\}$. $X \in \mathbb{R}^{N \times F}$ represents the features of all samples and F is the dimension of the joint feature space. Each participant $p \in \mathcal{P}$ holds a subset of features, denoted by $X^p = \{x_i^p\}_{i=1}^N \in \mathbb{R}^{N \times F^p}$ where F^p is the feature dimension on participant p. In other words, if we collect the feature vectors from all participants and concatenate them, X will be reconstructed, i.e., $X = [X^1, \ldots, X^P]$. Here, $[\cdot, \ldots, \cdot]$ denotes the concatenation operation. Only one participant called the leader participant holds the labels Y where $Y = \{y_i\}_{i=1}^N$.

Privacy Protection Techniques. Data privacy is a fundamental aspect of federated learning. Existing FL systems often employ privacy protection techniques to safeguard the transmitted data. Below we briefly summarize key techniques used in FL: ① *Differential privacy* (DP) generates random noises, such as Gaussian noise [34], and Binomial noise [35], to perturb the communicated data. Nevertheless, adding noises inevitably affects the model accuracy. ② *Secure multiparty computation* (SMC) allows multiple parties to compute a function over their inputs without sending local data. SMC requires a careful design for each desired operator and the setup of connections between any two parties, which is costly in real applications. ③ *Homomorphic encryption* (HE) protects the data by encryption and allows for various mathematical operations over ciphertexts, such as addition and multiplication [36]–[38]. HE can maintain the correctness of data aggregation and provide a strong privacy guarantee, albeit at the additional cost of encryption and decryption.

TABLE II
THE COMMONLY USED NOTATIONS

Symbol	Description
$\mathcal{P}$	Participant set with P participants
$\mathcal{D}$	Dataset of N samples
X	Feature matrix, $X \in \mathbb{R}^{N \times F}$
Y	Labels, $Y = \{y_i\}_{i=1}^N$
X^p	Local features held by participant p
$\mathcal{S}$	Subset of participants
$\ell(\mathcal{S})$	Log-likelihood for subset $\mathcal{S}$
d^p	Partial distances computed by participant p
d	Complete distances by summing d^p over $\mathcal{P}$
$\mathcal{T}$	k-nearest neighbors set
$w(p, s)$	Similarity between p and s

Therefore, we choose HE to encrypt sensitive data for privacy protection in this work. We denote the encryption function as $HE.Enc(*)$, the decryption function as $HE.Dec(*)$, and the sum operations over encrypted items as $HE.Sum(*)$.

B. Submodular Maximization

Submodularity is a property that naturally models diversity with a diminishing return, making it effective for subset selection across various applications [31]–[33], [39], [40]. Functions with this property are termed submodular functions.

Definition 1 (Submodular Function)**.** Let $\mathcal{V}$ be a finite ground set consisting of n distinct elements. For any subsets $\mathcal{A} \subseteq \mathcal{B} \subseteq \mathcal{V}$, a utility function $f : 2^{\mathcal{V}} \to \mathbb{R}$, assigning a real number to each subset of $\mathcal{V}$, is called *submodular* if for all elements $v \in \mathcal{V} \setminus \mathcal{B}$, the following inequality holds:

$$f(\{v\} \cup \mathcal{A}) - f(\mathcal{A}) \geq f(\{v\} \cup \mathcal{B}) - f(\mathcal{B}). \tag{1}$$

Submodular function has the diminishing returns property, where adding an element v to a smaller subset $\mathcal{A}$ yields at least as much benefit as adding it to a larger subset $\mathcal{B}$. This property naturally encourages diversity by valuing the addition of diverse elements to a subset while penalizing redundancy. For optimization, if a submodular function is monotone (i.e.,$f(\mathcal{A}) \leq f(\mathcal{B})$, for $\mathcal{A} \subseteq \mathcal{B}$) and normalized ($f(\emptyset) = 0$), a simple greedy algorithm can achieve a $1 - \frac{1}{e}$ approximation guarantee to the optimal solution [41], [42]. This greedy algorithm adds the most valuable element at each step, progressively building an increasingly effective subset.

III. THE VFPS-SM FRAMEWORK

In this section, we introduce our proposed framework VFPS-SM for PSP, grounded in the maximum likelihood estimation. In particular, we derive the likelihood function of data sample using KNN as the proxy model and show that PSP can be modeled as a submodular maximization problem.

A. KNN-Driven Likelihood Maximization

Problem Formulation. We study the participant selection problem in VFL. Given a consortium $\mathcal{P}$ with P participants, our objective is to find a participant sub-consortium $\mathcal{S}$ from the entire consortium $\mathcal{P}$ where $\mathcal{S} \subseteq \mathcal{P}, |\mathcal{S}| = S$ and achieve a high model accuracy on the chosen subset $\mathcal{S}$. In training machine learning models, estimating maximum likelihood parameters is

crucial. This process involves identifying parameter values that maximize the likelihood of observed data for a specific model, which is typically correlated with maximizing the model's accuracy [43]–[45]. Similarly, in addressing PSP, we adopt maximum likelihood estimation, a fundamental approach in classical ML models, to frame our solution. Specifically, our goal is to maximize the likelihood of the data within the subset $\mathcal{S}$ given the ML model, formalized as:

$$\max_{\mathcal{S}\subseteq\mathcal{P}} \sum_{i\in\mathcal{D}} \log p(x_i, y_i; \theta(\mathcal{S})) \tag{2}$$

where θ is the maximum likelihood estimate of the parameters in the ML model. And θ can be seen as a mapping function for the KNN classifier since it's a non-parametric model.

Vertical KNN. For a query sample $q = (x_q, y_q)$, the goal of KNN in the VFL scenario is to identify the k-nearest samples in the joint feature space X. In this work, we define the following two types of distances:

- *Partial distance.* Each participant $p \in \mathcal{P}$ calculates the distances between its local features of x_q and data samples in $\mathcal{D}$, denoted as $d^p = [(x_q^p - x_i^p)^2, \text{ for } i \in [N]]$.
- *Complete distance.* The complete distance $d = \sum_{p\in\mathcal{P}} d^p$ is the sum of partial distances over all participants.

Evaluating Subset: A Log-Likelihood Perspective. Given the model parameter $\theta(\mathcal{S})$ when training using subset $\mathcal{S}$, we consider a log-likelihood set function $\ell : 2^{\mathcal{P}} \to \mathbb{R}$ that maps subset $\mathcal{S} \subseteq \mathcal{P}$ to a log-likelihood score on the whole set $\mathcal{P}$:

$$\begin{aligned}\ell(S) &= \sum_{i\in\mathcal{D}} \log p(x_i, y_i; \theta(S)) \\ &= \sum_{i\in\mathcal{D}} \log p(x_i|y_i; \theta(S)) + \sum_{i\in\mathcal{D}} \log p(y_i|\theta(S)),\end{aligned} \tag{3}$$

where $p(x_i|y_i; \theta(S))$ and $p(y_i|\theta(S))$ are the generative likelihood and the prior likelihood of the sample $i \in \mathcal{D}$.

Objective and Prior Likelihood. Our goal is to select a participant subset $\mathcal{S}$ within the larger consortium $\mathcal{P}$ that maximizes the score ℓ. Consider the dataset $\mathcal{D}$ comprised of C distinct label classes. Let N_c denote the number of samples in $\mathcal{D}$ with matching labels. The prior likelihood, expressed as $p(y_i|\theta(\mathcal{S}))$, is thus formulated as $\frac{N_c}{N}$, where N represents the total number of samples within the dataset. Notably, this prior likelihood is a constant value and remains independent of the subset $\mathcal{S}$. Hence, the crucial question concerning the log-likelihood function ℓ is *how to appropriately design the generative likelihood function?*

Simplifying the Generative Likelihood. The function $p(x_i^p|y_i; \theta(\mathcal{S}))$ represents the probability of observing the feature values x_i^p for a given sample i at participant p conditioned on the class label y_i and the model parameters $\theta(\mathcal{S})$. This probability depends on the subset $\mathcal{S}$ used to train the model. To simplify this function for the KNN classifier, we introduce the following assumptions:

Assumption 1. For a given sample i, $p(x_i^p|y_i; \theta(\mathcal{S}))$ can be reflected by the participant s in $\mathcal{S}$ that is closest to the participant $p \in \mathcal{P}$, i.e., $s \triangleq \arg_{u\in\mathcal{S}} \max w_i(p, u)$ where $w_i(p, u)$ is the similarity between p and u.

Assumption 2. Given Assumption 1, the generative likelihood can be expressed as $p(x_i^p|y_i; \theta(S)) = c'e^{w_i(p,s)} = c' \exp(\max_{s\in S} w_i(p, s))$.

Assumption 1 indicates that the impact of features x_i^p on model training can be approximated by the most relevant participant s. It holds not for general machine learning models but only when using the KNN classifier for likelihood estimation. However, VFPS-SM remains broadly applicable, as KNN serves as an effective non-parametric proxy that captures geometric data patterns. Prior works have also leveraged KNN as a general proxy [46]–[48]. Assumption 2 formulates the conditional probability using an exponential function of the similarities. Together, these assumptions emphasize that only the most relevant participant's contribution is considered, minimizing redundancy and enhancing diversity.

Refining Similarity Measurement. Above, we express the generative likelihood based on participant similarities. Next, we study *how to measure this similarity using data (i.e., partial distance) during KNN execution?* For each $q \in \mathcal{D}$, let $\mathcal{T}$ denote its k-nearest neighbors within $\mathcal{P}$, and $d_{\mathcal{T}}^p = \sum_{t=1}^{|\mathcal{T}|} d_t^p$ represent the sum of partial distances for participant $p \in \mathcal{P}$, where $d_t^p = (x_q^p - x_t^p)^2, t \in \mathcal{T}$. The sum of complete distances for all participants is $d_{\mathcal{T}} = \sum_{p\in\mathcal{P}} d_{\mathcal{T}}^p$. We define the similarity between two participants, p_1 and p_2, for one query sample q as $w_q(p_1, p_2) = \frac{d_{\mathcal{T}} - \|d_{\mathcal{T}}^{p_1} - d_{\mathcal{T}}^{p_2}\|}{d_{\mathcal{T}}} \geq 0$, and the overall similarity across the dataset as $w(p_1, p_2) = \frac{1}{|\mathcal{D}|} \sum_{q\in\mathcal{D}} w_q(p_1, p_2)$. This similarity measure quantifies the relative difference in aggregated distances from common nearest neighbors, effectively capturing the divergence in their feature spaces based on spatial relationships and data distribution within the consortium.

Modeling Likelihood With KNN. Building on our foundational assumptions regarding spatial locality and influence, we utilize the KNN classifier to model the likelihood function in VFL. These assumptions guide the representation of the generative log-likelihood, which can be expressed as $\log p(x_i^p|y_i; \theta(\mathcal{S})) = w_i(p, s)$. The log-likelihood score for a subset $\mathcal{S}$ is thus formulated as:

$$\begin{aligned}\ell(\mathcal{S}) &= \sum_{p\in\mathcal{P}} \sum_{i\in\mathcal{D}} \max_{s\in\mathcal{S}} w_i(p, s) + \sum_{i\in\mathcal{D}} \log \frac{N_c}{N} + C \\ &= \sum_{p\in\mathcal{P}} \max_{s\in\mathcal{S}} w(p, s) + \sum_{i\in\mathcal{D}} \log \frac{N_c}{N} + C.\end{aligned} \tag{4}$$

In this model, the likelihood function is influenced by the closest participant s in subset $\mathcal{S}$ to each data feature in participant p, which is determined by maximizing the similarity measure $w(p, s)$. This approach allows us to capture the most significant interactions within the data, focusing on maximizing these values to enhance the overall effectiveness of the learning process. The constant terms, including the logarithm of the class occurrences $\log \frac{N_c}{N}$ and constant C,

remain invariant regardless of the subset $\mathcal{S}$, simplifying the optimization task to:

$$\max_{\mathcal{S}\subseteq\mathcal{P}} \ell(\mathcal{S}) \iff \sum_{p\in\mathcal{P}} \max_{s\in\mathcal{S}} w(p,s) \tag{5}$$

Consequently, the essence of optimizing $\ell(\mathcal{S})$ narrows down to leveraging the most significant similarities between participants, aiming to maximize the collective impact of these relationships on the overall model's performance.

B. Design Rationale

We leverage the nearest neighbors to approximate the data contribution of participants, a task conceptually similar to kernel density estimation (KDE) [49]–[51]. KDE is a widely used technique for estimating the probability density of a data sample by accounting for the contributions of nearby data points. KDE is straightforward: the denser the concentration of data points around a location, the higher the likelihood of observing a data sample there. This is achieved by using weighted distances from all observations across a linearly spaced set of points. Specifically, let $X_1, \ldots, X_n \in \mathbb{R}^d$ be a random sample from an unknown distribution P with density function p. Formally, KDE can be expressed as

$$p(x) = \frac{1}{nh}\sum_{i=1}^{n} K(\frac{x - X_i}{h}) \tag{6}$$

where $K : \mathbb{R}^d \to \mathbb{R}$ is a non-negative function called kernel function and $h > 0$ is a smoothing bandwidth that controls the amount of smoothing. The kernel function, $K(x)$, specifies how to compute the probability density given the distance,$x - X_i$. A commonly used kernel is the Gaussian kernel, which is expressed as $K(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}x^2}$. Let q denote a query data sample and $\mathcal{T} \subset \{1, 2, \cdots, N\}$ represent the indexes of q's k-nearest neighbors in the dataset $\mathcal{D}$. The likelihood of q using a Gaussian kernel over consortium $\mathcal{P}$ can be expressed as

$$\begin{aligned} p(x_q) &= c\exp\big(\sum_{i\in\mathcal{T}} -\|x_q - x_i\|^2\big) \\ &= c\exp\big(\sum_{i\in\mathcal{T}} -\|x_q - \sum_{p\in\mathcal{P}} x_i^p\|^2\big) \\ &= c\exp\big(\sum_{p\in\mathcal{P}}\big(\sum_{i\in\mathcal{T}} -\|x_q^p - x_i^p\|^2\big)\big), \end{aligned} \tag{7}$$

where c is a constant, x_i^p is the feature of sample x_i on participant p (by filling in 0 for the missing features), and similarly for x_q^p. Taking the logarithm on both sides and ignoring the constants, we have

$$\begin{aligned} \log p(x_q) &= \sum_{p\in\mathcal{P}}\big(\sum_{i\in\mathcal{T}} -\|x_q^p - x_i^p\|^2\big) \\ &= \sum_{p\in\mathcal{P}} -d_{\mathcal{T}}^p, \quad \text{where } d_{\mathcal{T}}^p = \sum_{i\in\mathcal{T}} \|x_q^p - x_i^p\|^2. \end{aligned} \tag{8}$$

Connection between KDE and Our Design. Our goal is to select a subset $\mathcal{S} \subseteq \mathcal{P}$ that maximizes the log-likelihood score $\ell(\mathcal{S})$. In essence, the selected subset $\mathcal{S}$ should aggregate likelihoods in a way that approximates the full aggregation across the entire consortium $\mathcal{P}$. We define a mapping $\sigma : \mathcal{P} \to \mathcal{S}$ such that the partial log-likelihood information from participant p is approximated by a selected participant $\sigma(p) = s \in \mathcal{S}$. For $s \in S$, let $\mathcal{U}_s \triangleq \{p \in \mathcal{P} | \sigma(p) = s\}$ be the set of participants approximated by participant s and $\gamma_s \triangleq |U_s|$. The full aggregated log-likelihood can be written as

$$\sum_{p\in\mathcal{P}} d_{\mathcal{T}}^p = \sum_{p\in\mathcal{P}} (d_{\mathcal{T}}^p - d_{\mathcal{T}}^{\sigma(p)} + d_{\mathcal{T}}^{\sigma(p)}) \tag{9}$$

$$= \sum_{p\in\mathcal{P}} [d_{\mathcal{T}}^p - d_{\mathcal{T}}^{\sigma(p)}] + \sum_{s\in\mathcal{S}} \gamma_s d_{\mathcal{T}}^s. \tag{10}$$

By subtracting the second term from both sides, taking the norms, and applying triangular inequality, we can get an upper bound on the error of estimating the full log-likelihood by S

$$\|\sum_{p\in\mathcal{P}} d_{\mathcal{T}}^p - \sum_{s\in\mathcal{S}} \gamma_s d_{\mathcal{T}}^s\| \le \sum_{p\in\mathcal{P}} \|d_{\mathcal{T}}^p - d_{\mathcal{T}}^{\sigma(p)}\|. \tag{11}$$

The right-hand side is the error on approximating the full log-likelihood using the selected subset of participants $\mathcal{S}$. The above inequality holds for any feasible mapping σ since the left-hand side does not depend on σ. We take the minimum of the right-hand side w.r.t. $\sigma(p), \forall p \in P$,

$$\|\sum_{p\in\mathcal{P}} d_{\mathcal{T}}^p - \sum_{s\in\mathcal{S}} \gamma_s d_{\mathcal{T}}^s\| \le \sum_{p\in\mathcal{P}} \min_{s\in\mathcal{S}} \|d_{\mathcal{T}}^p - d_{\mathcal{T}}^s\|.$$

The objective for minimizing the approximation error

$$\sum_{p\in\mathcal{P}} \min_{s\in\mathcal{S}} \|d_{\mathcal{T}}^p - d_{\mathcal{T}}^s\| \tag{12}$$

can be equivalently expressed as:

$$\sum_{p\in\mathcal{P}} \max_{s\in\mathcal{S}} d_{\mathcal{T}} - \|d_{\mathcal{T}}^p - d_{\mathcal{T}}^s\| \triangleq w(p,s). \tag{13}$$

For a discarded participant p due to participant selection, Assumption 1 says we approximate $d_{\mathcal{T}}^p$ using the contribution of another participant s (i.e., $d_{\mathcal{T}}^s$), where s is selected as $s \triangleq \arg_{u\in\mathcal{S}} \max w(p,u)$. That is, we select the participant s that yields the smallest approximation error for discarded participant p. This is also the best estimation we can get using the available information after participant selection.

C. K-Nearest Neighbors Submodular Function

Let $f(S) = \sum_{p\in P} \max_{s\in S} w(p,s)$. Our analysis transforms PSP into maximizing $f(S)$. Below, we will demonstrate that the KNN likelihood function $f(S)$ is submodular and discuss how to perform submodular maximization.

Theorem 1. The function f is a normalized, monotone submodular function.

Proof. Let $\mathcal{P}$ be a finite ground set with elements p, and let $\mathcal{S}$ be a subset of $\mathcal{P}$. The weight function $w : \mathcal{P} \times \mathcal{P} \to \mathbb{R}$ assigns a real number weight to each pair of elements (p, s), where both p and s belong to $\mathcal{P}$. We consider the set function $f : 2^{\mathcal{P}} \to \mathbb{R}$, defined for any subset $\mathcal{S} \subseteq \mathcal{P}$ as $f(\mathcal{S}) = \sum_{p\in\mathcal{P}} \max_{s\in\mathcal{S}} w(p,s)$.

Monotone. Given subsets $\mathcal{A}$ and $\mathcal{B}$ such that $\mathcal{A} \subseteq \mathcal{B}$, for each $p \in \mathcal{P}$, the maximization operation within f ensures that $\max_{s\in\mathcal{A}} w(p,s) \leq \max_{s\in\mathcal{B}} w(p,s)$ due to the broader or equal choice set in $\mathcal{B}$ compared to $\mathcal{A}$. Summing these maximum values over all p in $\mathcal{P}$ yields $f(\mathcal{A}) \leq f(\mathcal{B})$, demonstrating that f is monotone.

Normalized. For $\mathcal{S} = \emptyset$, there are no elements s to consider, implying for all $p \in \mathcal{P}$, $\max_{s\in\emptyset} w(p,s)$ contributes no value. Thus, $f(\emptyset) = \sum_{p\in\mathcal{P}} 0 = 0$.

Submodular. For subsets $\mathcal{A}, \mathcal{B}$ where $\mathcal{A} \subseteq \mathcal{B}$, and an element $x \notin \mathcal{B}$, we examine the marginal gains $f(\mathcal{A}\cup\{x\}) - f(\mathcal{A})$ and $f(\mathcal{B}\cup\{x\}) - f(\mathcal{B})$ through two cases for each $p \in \mathcal{P}$:

Case 1: If $w(p,x)$ is less than or equal to the maximum weight in both $\mathcal{A}$ and $\mathcal{B}$ for some p, then adding x does not change the maximum weight for p. Thus, x contributes no additional value, and the marginal gains for both $\mathcal{A}$ and $\mathcal{B}$ are zero for this p.

Case 2: If $w(p,x)$ exceeds the current maximum weight for p in $\mathcal{A}$ or $\mathcal{B}$, then we consider:

- If $w(p,x) > \max_{s\in\mathcal{A}} w(p,s)$, the marginal gain for $\mathcal{A}$ is $w(p,x) - \max_{s\in\mathcal{A}} w(p,s)$.
- Similarly, if $w(p,x) > \max_{s\in\mathcal{B}} w(p,s)$, the marginal gain for $\mathcal{B}$ is $w(p,x) - \max_{s\in\mathcal{B}} w(p,s)$.

Given that $\mathcal{A} \subseteq \mathcal{B}$, it follows $\max_{s\in\mathcal{A}} w(p,s) \leq \max_{s\in\mathcal{B}} w(p,s)$, implying the marginal gain for adding x to $\mathcal{A}$ is at least as large as the gain for $\mathcal{B}$. Combining both cases, we conclude that for every $p \in P$, the inequality $f(\mathcal{A}\cup\{x\}) - f(\mathcal{A}) \geq f(\mathcal{B}\cup\{x\}) - f(\mathcal{B})$ is satisfied, proving f's submodularity. □

The submodularity of function f is crucial as it ensures that adding participants to a smaller subset S yields a greater or equal utility gain than to a larger subset, efficiently addressing diminishing returns. Since f is submodular, we effectively *transform the vertical federated learning participant selection into a problem of maximizing the KNN submodular function*.

A Greedy Optimization Algorithm. We aim at finding a subset $\mathcal{S} \subseteq \mathcal{P}$ that maximizes $f(\mathcal{S})$ subject to $|\mathcal{S}| \leq S$. Obviously, finding the optimal solution to this problem is NP-hard. To achieve practical performance, we utilize the greedy algorithm to optimize the submodular function in polynomial time, with a $1 - \frac{1}{e}$ approximation guarantee to the optimal solution [41], [42]. This greedy algorithm for maximizing the k-nearest neighbors submodular function f starts from $\mathcal{S} = \emptyset$ and adds one participant $r \in \mathcal{P}\backslash\mathcal{S}$ with the greatest marginal gain to S in each step, where the marginal gain of r is $f(\mathcal{S}\cup\{r\}) - f(\mathcal{S})$. We provide the pseudocode of this greedy algorithm in Algorithm 1.

IV. Implementation

After formalizing VFPS-SM, we next study *how to efficiently run the KNN oracle in our framework.* We present two implementations: a baseline, which helps identify system bottlenecks, and an optimized version that employs top-k query algorithms to improve both computation and communication efficiency in vertical federated KNN.

Algorithm 1 Greedy Maximization

Input: Submodular function $f : 2^{\mathcal{P}} \to \mathbb{R}$
Output: Subset $\mathcal{S} \subseteq \mathcal{P}$ satisfying $|\mathcal{S}| = S$
$\mathcal{S} = \emptyset$ ▷ Initialize an empty set $\mathcal{S}$
while $|\mathcal{S}| < S$ **do** ▷ Repeat until size of $\mathcal{S}$ reaches S
  $r \in \mathcal{P}\backslash\mathcal{S}$ ▷ Select elements r not in S
  $r^* \in \arg\max[f(\mathcal{S}\cup\{r\}) - f(\mathcal{S})]$
    ▷ Choose r^* with maximum marginal gain
  $S = S \cup \{r^*\}$ ▷ Add the selected element r^* to S
end while

A. The Baseline Implementation

Below, we outline the straightforward implementation of vertical federated KNN using HE, focusing on KNN-based submodular maximization involving multiple query data pairs. For a given query set $\mathcal{Q} \subseteq \mathcal{D}$, our goal is to calculate the similarity $w(p,s)$ for each participant pair p and s in $\mathcal{P}$ based on $\mathcal{Q}$. These similarities then guide participant selection via the greedy algorithm in Algorithm 1.

System Architecture. There are three roles in the system–*key server*, *aggregation server*, and *participants*.

- *Key Server*. The key server generates a HE key pair consisting of a public key pk and a private key sk. It distributes pk to all participants and the aggregation server, while sk is securely sent to the leader participant.
- *Aggregation Server*. The aggregation server runs independently and provides the mathematical operators (e.g., *Sum*) that securely aggregate the encrypted data from participants.
- *Participants*. Each participant holds a feature subset of all instances and communicates with the aggregation server, while a leader participant holds the instance labels.

Execution Workflow. For each query sample $q = (x_q, y_q)$ in $\mathcal{Q}$, each participant $p \in \mathcal{P}$ calculates the partial distances d^p, which are then encrypted using a public key: $[d^p] = HE.Enc(d^p, pk)$ and sent to the aggregation server. The server then aggregates these encrypted partial distances into the complete distances $[d] = HE.Sum([d^p], pk)$ and forwards them to the leader participant. The leader decrypts these distances, identifies the k-nearest neighbors $\mathcal{T}$, and shares $\mathcal{T}$ with all participants. Each participant p computes the sum $d^p_{\mathcal{T}} = \sum_{t\in\mathcal{T}} d^p_t$, and sends $d^p_{\mathcal{T}}$ back to the leader to determine the similarity $w_q = \frac{d_{\mathcal{T}} - \|d^p_{\mathcal{T}} - d^s_{\mathcal{T}}\|}{d_{\mathcal{T}}}$. After processing all queries, the leader computes the overall similarity $w(p,s) = \frac{1}{|\mathcal{Q}|}\sum_{q\in\mathcal{Q}} w_q(p,s)$ for each participant pair p and s. These similarities are then used in a greedy algorithm to select the participant sub-consortium $\mathcal{S}$ from $\mathcal{P}$.

Cost Analysis. We assume the cost of calculating a partial distance is denoted as β, the cost of encrypting/decrypting a partial distance as ϕ_e/ϕ_d, the cost of transmitting a data item as η, and the cost of adding two encrypted distances as γ, the cost of adding two plaintext distances as δ. On each participant, the computation cost is $O(k\delta + N(\beta + \phi_e))$, and the communication cost is $O(k(\eta + P\delta) + N(\phi_d + \eta + P\gamma))$.

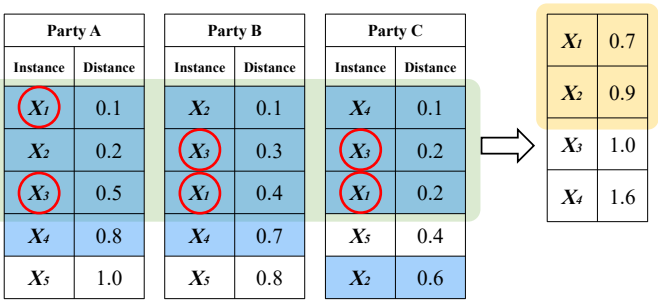

Fig. 2. An illustration of the Fagin algorithm.

B. Efficiency Optimization

The baseline implementation encrypts and transmits all instances' partial distances for a query sample, which becomes costly with large-scale datasets due to the high expense of HE operations. The computation and communication cost scale with the number of samples N. This leads us to consider:

> *Can we avoid encrypting all the instances' partial distances and ensure correctness meanwhile?*

We observe that the vertical federated KNN can be framed as a problem of multi-party top-k query where each party holds a ranked list of scores for the same data samples. To identify the top-k samples with the highest or lowest scores across multiple parties, it's necessary to merge these local ranked lists into a global ranking. Top-k query algorithms like Fagin and Threshold [52]–[55] can effectively identify the k-nearest neighbors across multiple parties, addressing the vertical KNN challenge. Thus, we propose to use a top-k query algorithm to find the k-nearest neighbors efficiently.

Top-k Query Algorithms. Assume that there are P parties and N instances, and each party $p \in [P]$ holds a data series $[(x_i, s_p(x_i))]$ where $i \in [N]$, and $s_p(x_i)$ represents the score of instance i on party p. The instances at each party are sorted by their scores to form a ranked score list. Globally, an overall score is assigned to each instance by aggregating the scores from all parties using an aggregate function. The partial distances between the participants' local features and the query can be seen as the scores of the instances. Each participant sorts the partial distances locally and generates a ranked list of identities (IDs). The aggregation server employs the top-k query algorithm to identify the top-k (minimal-k) instances across all participants. Note that the partial distances are sorted in ascending order.

Choice of Top-k Query Algorithm. In this work, we utilize the Fagin algorithm, widely recognized in the literature [52], [53]. Note that our VFPS-SM also supports other top-k query algorithms. Fagin assumes a monotone aggregate function: $F(x_1, \ldots, x_P) \leq F(x'_1, \ldots, x'_P)$ whenever $x_i \leq x'_i$ for all i. The Fagin algorithm involves three main steps: 1) sequentially access all the sorted lists in parallel until k instances appear in all lists; 2) perform random accesses to obtain the scores of all seen instances (including those instances that have not occurred in all lists); 3) compute the global scores with the aggregate function for all candidates found in the previous step, and return the top-k instances. Fig. 2 illustrates an example of identifying the minimal-2 instances from ascending ranked lists of 3 participants. First, Fagin's algorithm sequentially scans the lists until finding two instances that appear in all the 3 lists (i.e., X_1 and X_3). Next, the algorithm retrieves and aggregates the scores of all the encountered instances (marked with blue shade in Fig. 2) across the 3 lists (X_1, X_2, X_3, X_4). Finally, it sorts these instances based on their aggregated scores and selects the minimal-2 instances (i.e., X_1 and X_2).

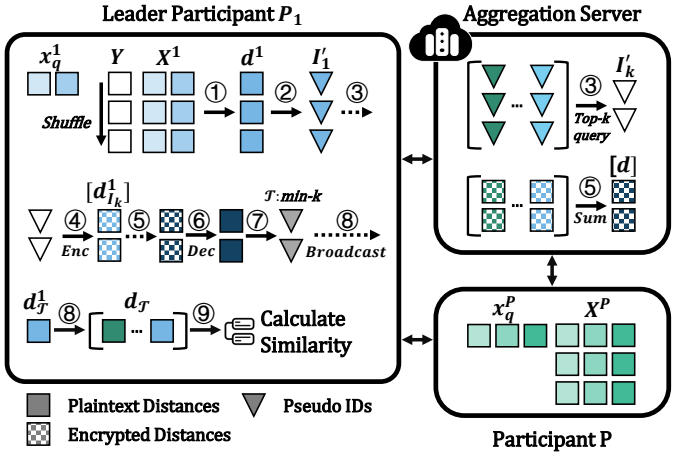

Fig. 3. The optimized workflow of VFPS-SM.

Assume an instance A with the scores $(s_1(A), \ldots, s_P(A))$ was not seen, and the instance B with the scores $(s_1(B), \ldots, s_P(B))$ is one of the candidates returned by the Fagin algorithm. Since $s_p(A) \leq s_p(B)$ for all $p \in [P]$, we can assure that $F(s_1(A), \ldots, s_P(A)) \leq F(s_1(B), \ldots, s_P(B))$. This establishes the correctness of the Fagin algorithm.

Execution Workflow. We employ the top-k query algorithms to boost efficiency by narrowing down the set of candidate samples for encryption and analysis. Taking Fig. 3, initially, participants shuffle all samples using a consistent seed and generate pseudo IDs I' mapped back to the original IDs I. For each query $q \in \mathcal{Q}$, each participant $p \in \mathcal{P}$ computes the partial distances d^p between q and local features X^p, and sorts d^p in ascending order to create sub-rankings of pseudo IDs I'_p. Then participant p sends I'_p to the aggregation server in a mini-batch, that is, each participant iteratively sends b pseudo IDs to the server until Fagin terminates (Step ①-②). The server employs Fagin's algorithm to identify the top-k candidate pseudo IDs I'_k from all sub-rankings (Step③). Then participants remap the top-k pseudo IDs I'_k back to the original IDs I_k and encrypt the corresponding partial distances $[d^p_{I_k}]$ (Step④). The following steps are similar to the baseline method (Step ⑤-⑧). As we will show, the candidate set I'_k is smaller than the training set, optimizing the encryption and computation load.

Cost Analysis. Assume the top-k query algorithm needs to sequentially scan n rows until s pseudo IDs have been seen in all participants. The computation and communication costs of the Fagin phase are both $O(n)$. There are at most $C = \min(Pn - Ps, N)$ unique candidates seen so far. Therefore, the computation cost for the following steps is

at most $O(k\delta + C(\beta + \phi_e))$, and the communication cost is at most $O(k(\eta + P\delta) + C(\beta_d + \eta + P\gamma))$, based on our definition in Section IV-A. HE operations are known to be time-consuming. As we will empirically demonstrate later, the costs of the baseline implementation can be inefficient in practice, particularly in many large-scale scenarios. In contrast, our optimized method via top-k query algorithms can effectively mitigate this performance bottleneck.

C. Security Analysis

Following existing VFL systems [7], [13], [14], [17], [27], we analyze the security of VFPS-SM from three perspectives: *feature security*, *label security*, and *identity security*. We consider the semi-honest model, a commonly used threat model in FL [15], [22], [56]. That is, each party follows the protocol but it tries to speculate on others based on the received data. Note that there is no collusion between the server and participants; otherwise, the aggregation server can decrypt anything it receives from the participants.

- *Feature Security* is protected against any malicious party. In our framework, since the local features are not directly shared, each participant only transmits the encrypted partial distances to other parties. Existing studies on attacking VFL [57]–[59] assume the server may steal the participant model and data by generating synthetic data samples to query the participant model for data reconstruction. This kind of attack does not work for our proposed VFPS-SM because (i) we do not allow any party to query the participants with new data samples, and (ii) even if some parties get the partial distance, using it to reconstruct the participant data features is an under-constrained problem (i.e., using 1 dimension distance to guess high dimension features), which can not provide meaningful solutions.
- *Label Security* is maintained if the leader does not collude with other participants. The leader maintains the labels and never shares them with others. If the leader does not collude with others, the curious or colluding participant cannot speculate the labels. However, if the leader colludes with others, the colluding participants can directly get the labels.
- *Identity Security* is protected against any malicious party. Since the original IDs of instances are shuffled and replaced by "pseudo" IDs, identity security is guaranteed against the server. Moreover, participants use the same pseudo IDs, so even if they collude, they cannot access or decipher the original identities, preserving identity security.

D. Limitation of VFPS-SM

In this work, we propose VFPS-SM, which is the first framework focused on the diverse participant selection in VFL. Although our VFPS-SM works well on many workloads, one limitation of VFPS-SM is that its contribution scores cannot be used to reward the participants and encourage them to join model training. This is because VFPS-SM evaluates participant contributions based on submodularity, and submodular functions assign diminishing returns for the participants that are selected later. This causes fairness problems as the contributions (and thus rewards) are biased towards the participants selected earlier. We leave solving this problem for future work.

TABLE III
EVALUATED DATASETS

Datasets	# Instances	# Features	Domain
Bank [27]	10,000	11	Finance
Credit [6]	30,000	23	
Phishing [19]	11,055	68	Internet
Web [15]	64,700	300	
Rice [61]	18,185	10	Science
Adult [4]	32,561	123	
IJCNN [40]	141,691	22	
SUSY [14]	5,000,000	18	
HDI [60]	253,661	21	Healthcare
SD [60]	991,346	23	

V. Experimental Evaluation

We detail the experimental settings in Section V-A, compare VFPS-SM with state-of-the-art baselines in Section V-B, discuss additional results that evaluate VFPS-SM's design in Section V-C, and conduct an ablation study in Section V-D.

A. Experiment Settings

Datasets and Models. The evaluated datasets in our study are listed in Table III. These datasets are collected from online repositories [60], [61] and prior works [27], [62], [63]. Each dataset is randomly partitioned into a training set (80%), a validation set (10%), and a test set (10%). We randomly split each dataset into four vertical partitions based on the number of features, and put each partition on one physical machine.

To validate the effectiveness of VFPS-SM, we run several downstream classification tasks over the selected participants. We choose k-nearest neighbors(KNN), logistic regression (LR), and multi-layer perceptron (MLP) as the representative machine learning models.

Baselines and Metrics. We compare VFPS-SM with the following baselines: ① *RANDOM*: randomly selects l participants; ② *SHAPLEY*: selects l participants with the highest values calculated via a vertical federated KNN proxy model; ③ *VF-MINE* [27]: computes participant importance based on mutual information and selects the top-l; ④ *VFPS-SM-BASE*: a direct implementation of vertical federated KNN. We evaluate our VFPS-SM using two key metrics: model accuracy and end-to-end running time cost. The model accuracy is measured by the performance of the trained model over the test dataset. The end-to-end running time cost includes both participant selection time and model training time. The results are averaged over five runs for robustness and reliability.

Implementation. We use the Numpy and PyTorch libraries for data loading and tensor operations. For communication between parties, we implement RPC communication using proto3 and gRPC. We utilize the Cheon-Kim-Kim-Song (CKKS) scheme provided by TenSEAL [38] to implement homomorphic encryption We adopt the Adam optimizer [64] as the optimization algorithm for LR and MLP. We set the

TABLE IV
TEST ACCURACY ON DIFFERENT DOWNSTREAM TASKS. THE HIGHEST AND THE SECOND HIGHEST ACCURACY AMONG RANDOM, SHAPLEY, VFMINE, AND VFPS-SM IS HIGHLIGHTED IN BOLD AND UNDERLINED, RESPECTIVELY.

Task	Method	Bank	Phishing	Rice	Credit	Adult	Web	IJCNN	HDI	SD	SUSY
KNN	**ALL**	0.8300	0.9483	0.9911	0.8111	0.8167	0.9883	0.9833	0.9250	0.7111	0.7844
	RANDOM	0.7833	0.8738	0.9789	0.7856	0.7767	0.9800	0.9100	0.9083	0.6678	0.7422
	SHAPLEY	**0.8400**	0.9336	0.9900	**0.8267**	**0.8500**	0.9900	**0.9844**	0.9164	0.7089	0.7722
	VFMINE	0.8100	0.9335	0.9889	0.8100	0.7900	0.9900	0.9289	0.9083	0.7111	0.7544
	VFPS-SM	**0.8400**	**0.9369**	**0.9911**	0.8244	**0.8500**	**0.9917**	0.9811	**0.9167**	**0.7156**	**0.7756**
LR	**ALL**	0.8156	0.9360	0.9882	0.8115	0.8463	0.9866	0.9197	0.9075	0.7263	0.7876
	RANDOM	0.7920	0.8660	0.9820	0.7835	0.8168	0.9796	0.9021	0.9027	0.6760	0.7140
	SHAPLEY	0.8153	0.9127	0.9865	0.8102	**0.8388**	0.9813	0.9072	0.9062	0.7057	0.7620
	VFMINE	0.8006	0.9047	**0.9876**	0.7983	0.8306	0.9810	0.9048	0.9061	0.6952	0.7620
	VFPS-SM	**0.8156**	**0.9145**	0.9875	**0.8109**	**0.8388**	**0.9815**	**0.9075**	**0.9064**	**0.7067**	**0.7819**
MLP	**ALL**	0.8595	0.9418	0.9889	0.8062	0.8415	0.9883	0.9570	0.9082	0.8205	0.8011
	RANDOM	0.8006	0.8696	0.9786	0.7785	0.8188	0.9782	0.8878	0.9061	0.7893	0.7563
	SHAPLEY	**0.8367**	0.9196	0.9879	0.8188	**0.8365**	0.9822	0.9337	0.9063	0.7995	0.7908
	VFMINE	0.8256	0.9063	0.9883	0.7921	0.8273	**0.9830**	0.9160	0.9061	**0.8075**	0.7786
	VFPS-SM	**0.8367**	**0.9270**	**0.9887**	**0.8190**	**0.8365**	**0.9830**	**0.9461**	**0.9067**	0.8070	**0.7932**

TABLE V
END-TO-END RUNNING TIME ON DIFFERENT DOWNSTREAM TASKS (SECONDS). THE FASTEST AND THE SECOND FASTEST RUNNING TIME AMONG RANDOM, SHAPLEY, VFMINE, AND VFPS-SM ARE HIGHLIGHTED IN BOLD AND UNDERLINED, RESPECTIVELY.

Task	Method	Bank	Phishing	Rice	Credit	Adult	Web	IJCNN	HDI	SD	SUSY
KNN	**ALL**	205	296	522	1460	1800	6254	15018	21849	69612	306055
	RANDOM	**108**	**149**	**337**	**770**	**958**	**954**	**6509**	**10984**	**48531**	**200140**
	SHAPLEY	517	669	978	1648	1963	2721	9692	17291	75476	336244
	VFMINE	218	252	440	881	1077	1104	6660	11448	49221	200989
	VFPS-SM	135	174	361	798	989	997	6551	11113	48738	200513
LR	**ALL**	181	1981	1222	473	527	2077	2523	1790	6916	13503
	RANDOM	**116**	1622	789	**203**	**328**	836	**497**	**1056**	**3917**	**3881**
	SHAPLEY	525	2353	1502	1076	1220	2746	3677	7400	30862	139984
	VFMINE	225	1830	929	311	559	1058	646	1539	4607	4729
	VFPS-SM	143	**1436**	**742**	235	471	**736**	541	1149	4124	4238
MLP	**ALL**	1058	1977	1132	2350	4646	5600	16297	12801	169791	846668
	RANDOM	**430**	1081	729	1274	1758	4459	8140	**6671**	**41741**	**460684**
	SHAPLEY	846	2055	1465	1901	2735	6414	16425	13054	68663	591072
	VFMINE	545	942	788	1807	1901	4469	5880	6944	42848	462359
	VFPS-SM	465	**876**	**722**	**1132**	**1810**	**4444**	**5489**	6930	42169	463324

batch size to 100 and terminate the training of models after 200 epochs or when the validation loss does not decrease within 5 consecutive epochs. To tune the optimal learning rate, we conduct a grid search within the range $\{0.001, 0.01, 0.1\}$. All experiments are conducted on Amazon AWS, with each party deployed on separate g4dn.xlarge EC2 GPU instances.

Hyper-parameter Settings. We implement downstream ML models using a split learning framework: ① *MLP*: The model has 3 layers and is partitioned into two parts: a 1-layer bottom model on the participants and a 2-layer top model on the server. The dimensions of the hidden layers are the same as the input feature, and the activation function is ReLU. ② *LR*: Each participant maintains a single linear layer, and the server aggregates the outputs of the participant by summing them. ③ *KNN*: Each participant computes the partial distances, and the server aggregates them into the complete distances to identify the top-k neighbors. To protect data privacy, we employ homomorphic encryption to secure the transmitted data, which include participant-side model outputs for LR and MLP and partial distances for KNN.

B. Main Results

Table IV and Table V report the test accuracy and end-to-end running time for three downstream models respectively.

Comparison to Participant Selection Baselines. We empirically study the first question: *can* VFPS-SM *outperform the other participant selection baselines in both effectiveness and efficiency?* Selecting 50% participants (2 out of 4) for downstream classification, RANDOM is the fastest but sacrifices accuracy. SHAPLEY is the slowest and computationally demanding due to requiring extensive retraining across all participant subsets which may hinder its real-world applicability. VF-MINE is the second slowest since it needs to compute the mutual information for different participant groups. In contrast, VFPS-SM is marginally slower than RANDOM but significantly outpaces both SHAPLEY and VF-MINE. For example, on the SUSY dataset with the LR model, VFPS-

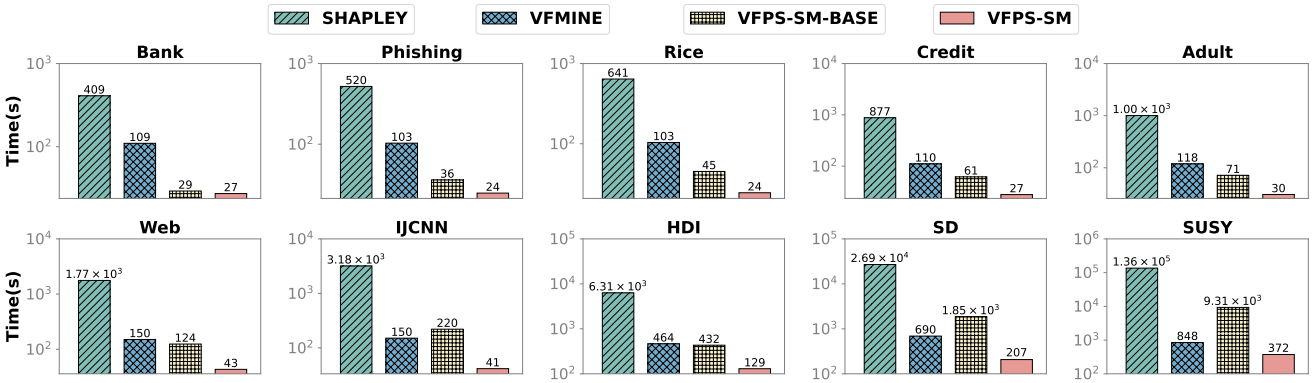

Fig. 4. VFPS-SM vs. the baselines in terms of selection time. Note that, the selection time of RANDOM and ALL TRAIN is both 0 since they instantly choose participants. The y-axis is the selection time in seconds.

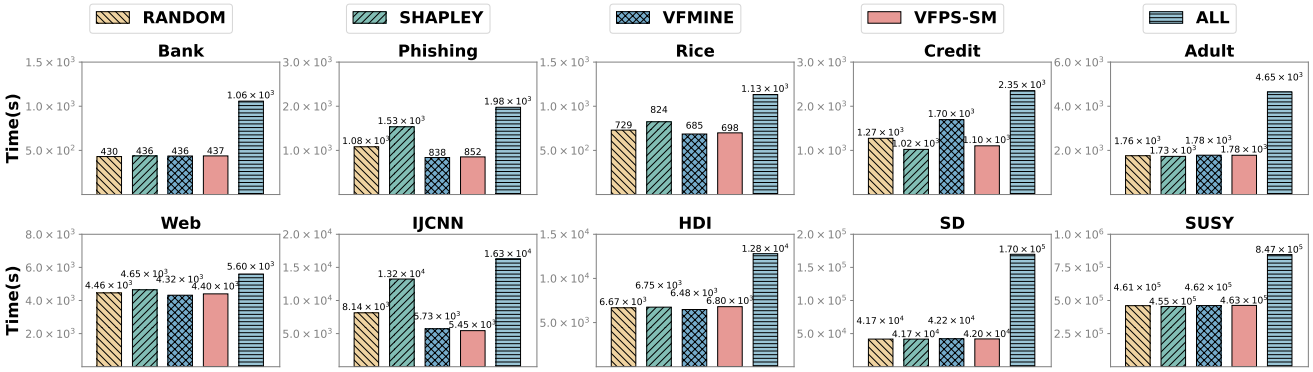

Fig. 5. VFPS-SM vs. the baselines in terms of training time on the model MLP. The y-axis is the training time in seconds.

SM is 33.0× faster than SHAPLEY, 1.1× faster than VF-MINE, and only 1.1× slower than RANDOM in running time. Additionally, VFPS-SM often achieves higher accuracy, like on the IJCNN dataset with the MLP model, where it surpasses SHAPLEY by 1.2%, VF-MINE by 3.0%, and RANDOM by 5.8%, thanks to selecting more diverse participant subsets which enhance the performance of downstream models.

Comparison to All-Participant Training. Then we turn to another question: *does participant selection yield advantages compared to training with all participants?* The results of end-to-end running time using all participants ("ALL") are presented in Table IV and Table V. Overall, the end-to-end running time of VFPS-SM is much faster than using all participants, which does not have a selection phase. Meanwhile, training a model on a selected subset of participants can obtain similar accuracies compared to ALL. For example, on the Rice dataset with the LR model, VFPS-SM demonstrates a 1.65× speedup in running time, with only a marginal reduction (0.07%) in the model accuracy. Interestingly, on some datasets we even observe a higher accuracy after participant selection compared to the full-fledged training. For example, on the SD dataset with the KNN model, VFPS-SM achieves a 0.45% increase in model accuracy to ALL while exhibiting a 1.4× improvement in running time. Similarly, on the Credit dataset with the MLP model, using the MLP model, VFPS-SM realizes a 1.3% increase in model accuracy compared to ALL. This suggests that certain participants may negatively impact model quality, highlighting the importance of selecting high-value and diverse participants. These experimental results underscore the effectiveness and efficiency of strategic participant selection in enhancing model performance.

Time Breakdown. We further decouple the running time into selection phase and training phase. The selection time is depicted in Fig. 4 and the training time of MLP is shown in Fig. 5. ① *Selection Time*: Our proposed VFPS-SM substantially reduces participant selection costs across all datasets compared to VF-MINE and SHAPLEY. VFPS-SM utilizes less time for HE operations by handling fewer instances, leading to significant speed improvements. For example, on the SUSY dataset, VFPS-SM achieves a speedup of 365.2 × relative to SHAPLEY and 2.3 × to VF-MINE. The performance gains come from the top-k algorithm's efficient ranked-list merging and batched candidate processing. ② *Training Time*: Training with a sub-consortium of participants significantly reduces the time required versus full consortium ("ALL") across all datasets, mainly due to reduced communication costs from fewer participants in VFL. For example, on the IJCNN dataset, training with a participant sub-consortium selected

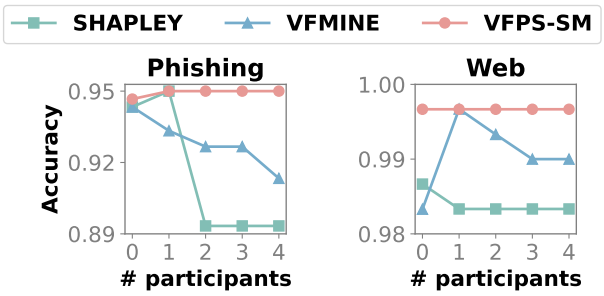

Fig. 6. Study of participant diversity. The x-axis represents the number of manually injected duplicate participants, while the y-axis shows the accuracy of the KNN model. The initial consortium size is four, and two participants are selected from the consortium for evaluation.

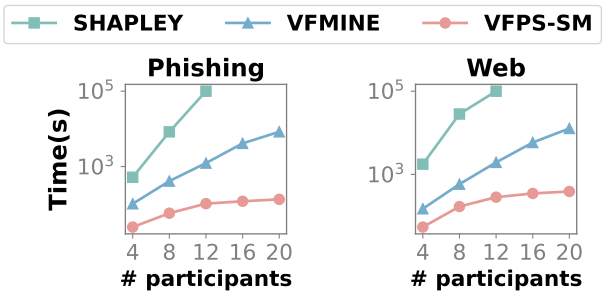

Fig. 7. Scalability evaluation. The x-axis is the number of participants, and the y-axis is the algorithm's running time.

by VFPS-SM achieves a 3.0× speedup over training with all participants. Meanwhile, VFPS-SM 's selection time is markedly shorter than the full training duration. For example, on the SUSY dataset, VFPS-SM 's selection time is 372 s compared to the full training time of 8.47×10^5 s.

C. More Experimental Results

Study of Diversity. To assess the impact of participant diversity, we experiment with the Phishing, and Web datasets, initially split into four partitions and then augmented with duplicate partitions. As shown in Fig. 6, using KNN as the downstream model, SHAPLEY and VF-MINE show decreased accuracy with additional duplicates, while VFPS-SM maintains almost the same model accuracy. For example, with up to four duplicate participants on the Phishing dataset, VF-MINE and SHAPLEY's accuracies drop by 3.03% and 5.01% respectively, while VFPS-SM improves by 0.34%. VFPS-SM leverages submodular maximization to prioritize diversity, effectively identifying duplicates and boosting model quality.

Scalability Evaluation. To study the scalability performance of our framework and the baselines, we partition Phishing, and Web datasets into varying numbers of partitions (4/8/12/16/20). Fig. 7 reports the running time of SHAPLEY, VF-MINE, and VFPS-SM. As expected, SHAPLEY's running time increases nearly exponentially with the number of participants, as each additional participant doubles the required coalition evaluations. Similarly, VF-MINE's running time exhibits a slight exponential increase, due to its pairwise mutual information computations across growing participant sets. In contrast, VFPS-SM consistently outperforms the baselines across all datasets and partition scenarios by evaluating only one group, the entire consortium. This demonstrates the superior scalability of our VFPS-SM when increasing the number of participants, compared to the existing methods.

Impact of k for the KNN classifier. To further study the impact of k, we conduct experiments on the Phishing and Web datasets with varying k. As shown in Fig. 8, after a certain threshold ($k \geq 10$), increasing k further has a minimal impact on performance. This is because the likelihood estimation becomes stable due to the aggregation of enough data samples.

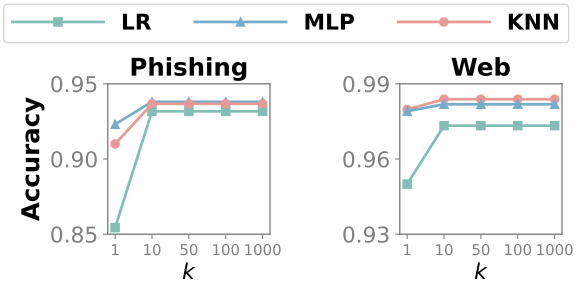

Fig. 8. Impact of k. The x-axis is the number of nearest samples identified in VFPS-SM, and the y-axis is the model test accuracy.

D. Ablation Study

Below, we conduct ablation studies to compare the performance of VFPS-SM-BASE and VFPS-SM.

Number of Candidates. In Fig. 9, we report the average number of data instances that are encrypted and communicated for each query instance in our experiments. VFPS-SM-BASE must encrypt partial distances for all training instances per query, incurring substantial computational and communication costs, especially for large-scale datasets. In contrast, VFPS-SM employs top-k query to select candidate subsets, significantly reducing these overheads. As we can observe in Fig. 9, VFPS-SM with the optimization of top-k query can greatly reduce the number of candidate instances involved in the processing procedure on all datasets. For example, compared to VFPS-SM-BASE, VFPS-SM decreases the average number by 46.0× on the SUSY dataset and 24.5× on the Rice dataset.

Selection Cost. We also compare the selection time of the two methods. As illustrated in Fig. 4, VFPS-SM consistently outperforms VFPS-SM-BASE. In resonance with the comparison of candidate instances, VFPS-SM takes a much shorter time in terms of encryption, decryption, and communication. The reason is that, in VFPS-SM, each participant handles much fewer instances and transfers fewer partial distances. On the large-scale datasets, our VFPS-SM achieves a remarkable reduction of cost through system efficiency optimization. For instance, in terms of the selection time, our VFPS-SM surpasses VFPS-SM-BASE, achieving a speed improvement of 8.9× on the SD dataset, and 25.0× on the SUSY dataset.

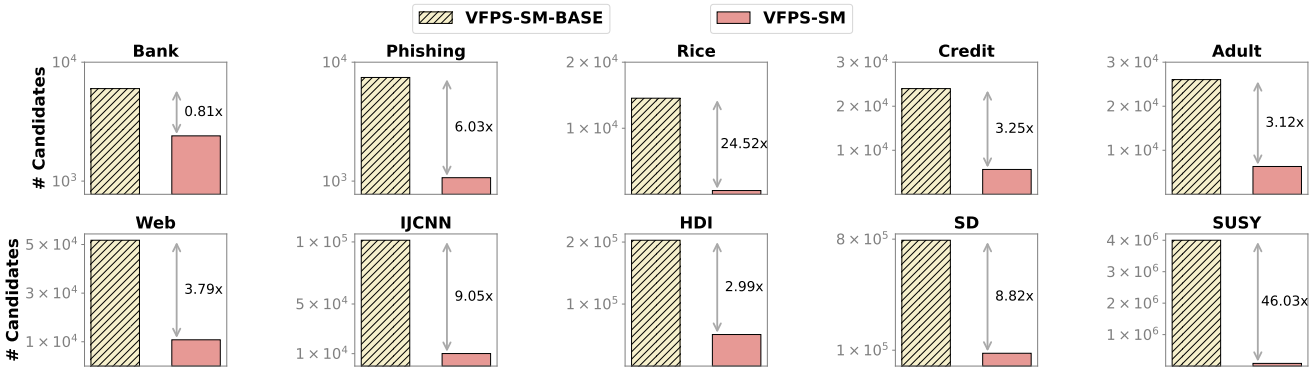

Fig. 9. Ablation study on the effect of the top-k query algorithm. The y-axis shows the average number of encrypted and communicated samples per que

VI. Related Work

Vertical Federated Learning. VFL stands out as a promising paradigm for privacy-preserving collaborative learning, wherein different parties share a common sample space while retaining distinct feature spaces. In particular, Hardy et al. [65] and Cheng et al. [66] propose solutions for logistic regression and gradient boosted decision trees on vertically partitioned data, incorporating homomorphic encryption for enhanced data privacy. Yang et al. [67] extend this framework by adopting the quasi-Newton method to reduce communication costs. Inspired by split learning, Vepakomma et al. [68] and Wu et al. [69] introduce model-splitting concepts to support more complex deep neural networks in VFL settings. Previous works also studied other settings in VFL, e.g., how to align the data among different parties [70], how to reduce the number of samples required for training in VFL [71], how to adopt asynchronous training [72], and how to defend against attacks in VFL. In this work, we aim to effectively and efficiently select representative participants to enhance VFL performance.

Participant Selection in Federated Learning. Participant selection is essential for fast and accurate FL. Many studies have focused on developing strategies for horizontal federated learning (HFL) [73]–[75] to enhance efficiency, fairness, and model performance. Some works exploit submodularity to select representative participants in HFL [31], [76], [77]. For example, Zhang et al. [78] define the contribution of a subset based on the expected generalization error and optimize the selection problem with a constant approximate ratio. More recently, research has started to focus on selection regarding the contribution of each participant to the trained model. The Shapley value is a fair metric for contribution evaluation, which has been widely used in machine learning [23], [46], [79]. For example, Jia et al. [80] propose KNN-Shapley, a data valuation framework that leverages KNN as a proxy model to compute the Shapley value of each data sample in centralized ML. The concept of Shapley value was introduced into HFL to measure participant contributions, incentivizing engagement [81]. Sun et al. [82] propose a client sampling strategy based on the Shapley value and aggregate model updates accordingly to enhance the model's robustness.

However, participant selection in VFL remains less explored compared to HFL. Unlike in HFL, where participants train locally, a single participant in VFL cannot access the full feature space. As a result, computing the loss and gradient for a single data requires collaboration and communication with other parties, adding complexity to the selection process. Wang and Dang et al. [25] apply Shapley values to assess contributions in vertical federated linear regression, although this method is resource-intensive due to repeated model training. Jiang et al. [27] use mutual information to identify key participants in VFL. Huang et al. [83] utilize a sampling strategy based on VFMINE to reduce the costs of computation of mutual information, although often at the expense of model precision. Unlike prior contribution-focused approaches, we prioritize selection diversity to boost model performance.

VII. Conclusion

We identify high costs and overlooked diversity in existing participant selection algorithms. To address both challenges, we introduce VFPS-SM, a framework that can efficiently and effectively select a participant subset in VFL. VFPS-SM approaches PSP by maximizing data sample likelihood within chosen subsets, using the KNN classifier as a proxy. Our analysis confirms the likelihood function's submodularity, enhancing participant diversity in the selection process and allowing for a greedy algorithm that maximizes each selection for optimal gain. Additionally, VFPS-SM employs top-k query algorithms, reducing encrypted communication and computation burdens. Experiment results show that VFPS-SM can reduce participant selection time and provide strong performance across various ML models and datasets.

Acknowledgment

This work was sponsored by Key R&D Program of Hubei Province (2023BAB077), and National Natural Science Foundation of China (62472327). This work was supported by Ant Group through CCF-Ant Research Fund (CCF-AFSG RF20240104). Jiawei Jiang and Quanqing Xu are the corresponding authors.

REFERENCES

[1] P. Voigt and A. v. d. Bussche, *The EU general data protection regulation (gdpr): a practical guide*. Springer Publishing Company, Incorporated, 2017.

[2] C. Niu, Z. Zheng, F. Wu, X. Gao, and G. Chen, "Trading data in good faith: integrating truthfulness and privacy preservation in data markets," in *2017 IEEE 33rd International Conference on Data Engineering (ICDE)*. IEEE, 2017, pp. 223–226.

[3] S. Shastri, V. Banakar, M. Wasserman, A. Kumar, and V. Chidambaram, "Understanding and benchmarking the impact of gdpr on database systems," *Proceedings of the VLDB Endowment (VLDB)*, p. 1064–1077, 2019.

[4] Q. Li, Y. Diao, Q. Chen, and B. He, "Federated learning on non-iid data silos: an experimental study," in *2022 IEEE 38th International Conference on Data Engineering (ICDE)*. IEEE, 2022, pp. 965–978.

[5] X. Li, Y. Hu, W. Liu, H. Feng, L. Peng, Y. Hong, K. Ren, and Z. Qin, "Opboost: a vertical federated tree boosting framework based on order-preserving desensitization," *Proceedings of the VLDB Endowment (VLDB)*, vol. 16, no. 2, p. 202–215, oct 2022.

[6] A. Li, Y. Cao, J. Guo, H. Peng, Q. Guo, and H. Yu, "Fedcss: joint client-and-sample selection for hard sample-aware noise-robust federated learning," in *Proceedings of the International Conference on Management of Data (SIGMOD)*, vol. 1, no. 3. ACM New York, NY, USA, 2023, pp. 1–24.

[7] Y. Wu, N. Xing, G. Chen, T. T. A. Dinh, Z. Luo, B. C. Ooi, X. Xiao, and M. Zhang, "Falcon: a privacy-preserving and interpretable vertical federated learning system," *Proceedings of the VLDB Endowment (VLDB)*, vol. 16, no. 10, pp. 2471–2484, 2023.

[8] Y. Cheng, L. Zhang, J. Wang, X. Chu, D. Huang, and L. Xu, "Fedmix: boosting with data mixture for vertical federated learning," in *2024 IEEE 40th International Conference on Data Engineering (ICDE)*. IEEE, 2024, pp. 3379–3392.

[9] Y. Wang, K. Li, Y. Luo, G. Li, Y. Guo, and Z. Wang, "Fast, robust and interpretable participant contribution estimation for federated learning," in *2024 IEEE 40th International Conference on Data Engineering (ICDE)*. IEEE, 2024, pp. 2298–2311.

[10] H. Lin, L. Shou, K. Chen, G. Chen, and S. Wu, "Fl-guard: A holistic framework for run-time detection and recovery of negative federated learning," *Data Science and Engineering*, vol. 9, no. 2, pp. 204–219, 2024.

[11] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: concept and applications," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 10, no. 2, pp. 1–19, 2019.

[12] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: challenges, methods, and future directions," *IEEE Signal Processing Magazine (SPM)*, vol. 37, no. 3, pp. 50–60, 2020.

[13] Y. Wu, S. Cai, X. Xiao, G. Chen, and B. C. Ooi, "Privacy preserving vertical federated learning for tree-based models," *Proceedings of the VLDB Endowment (VLDB)*, vol. 13, no. 12, p. 2090–2103, 2020.

[14] F. Fu, Y. Shao, L. Yu, J. Jiang, H. Xue, Y. Tao, and B. Cui, "Vf2boost: very fast vertical federated gradient boosting for cross-enterprise learning," in *Proceedings of the International Conference on Management of Data (SIGMOD)*. Association for Computing Machinery, 2021, p. 563–576.

[15] F. Fu, H. Xue, Y. Cheng, Y. Tao, and B. Cui, "Blindfl: vertical federated machine learning without peeking into your data," in *Proceedings of the International Conference on Management of Data (SIGMOD)*, 2022, pp. 1316–1330.

[16] Z. Li, T. Wang, and N. Li, "Differentially private vertical federated clustering," *Proceedings of the VLDB Endowment (VLDB)*, vol. 16, no. 6, p. 1277–1290, 2023.

[17] Z. Xiang, T. Wang, W. Lin, and D. Wang, "Practical differentially private and byzantine-resilient federated learning," in *Proceedings of the International Conference on Management of Data (SIGMOD)*, vol. 1, no. 2. ACM New York, NY, USA, 2023, pp. 1–26.

[18] Q. Zhang, X. Yan, Y. Ding, F. Fu, Q. Xu, L. Ziyi, C. Hu, and J. Jiang, "Hacore: efficient coreset construction with locality sensitive hashing for vertical federated learning," *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2025.

[19] T. Castiglia, Y. Zhou, S. Wang, S. Kadhe, N. Baracaldo, and S. Patterson, "Less-vfl: communication-efficient feature selection for vertical federated learning," in *International Conference on Machine Learning (ICML)*. PMLR, 2023, pp. 3757–3781.

[20] Z. Li, B. Ding, C. Zhang, N. Li, and J. Zhou, "Federated matrix factorization with privacy guarantee," *Proceedings of the VLDB Endowment (VLDB)*, vol. 15, no. 4, 2021.

[21] F. Fu, X. Miao, J. Jiang, H. Xue, and B. Cui, "Towards communication-efficient vertical federated learning training via cache-enabled local updates," *Proceedings of the VLDB Endowment (VLDB)*, vol. 15, no. 10, 2022.

[22] R. Fu, Y. Wu, Q. Xu, and M. Zhang, "Feast: a communication-efficient federated feature selection framework for relational data," in *Proceedings of the International Conference on Management of Data (SIGMOD)*, vol. 1, no. 1. ACM New York, NY, USA, 2023, pp. 1–28.

[23] S. Zheng, Y. Cao, and M. Yoshikawa, "Secure shapley value for cross-silo federated learning," *Proceedings of the VLDB Endowment (VLDB)*, vol. 16, no. 7, 2023.

[24] A. Li, H. Peng, L. Zhang, J. Huang, Q. Guo, H. Yu, and Y. Liu, "Fedsdg-fs: efficient and secure feature selection for vertical federated learning," *2023 IEEE Conference on Computer Communications (INFOCOM)*, pp. 1–10, 2023.

[25] G. Wang, C. X. Dang, and Z. Zhou, "Measure contribution of participants in federated learning," in *IEEE International Conference on Big Data (Big Data)*, 2019, pp. 2597–2604.

[26] X. Han, L. Wang, and J. Wu, "Data valuation for vertical federated learning: an information-theoretic approach," *arXiv preprint arXiv:2112.08364*, 2021.

[27] J. Jiang, L. Burkhalter, F. Fu, B. Ding, B. Du, A. Hithnawi, B. Li, and C. Zhang, "Vf-ps: how to select important participants in vertical federated learning, efficiently and securely?" *Advances in Neural Information Processing Systems (NIPS)*, vol. 35, pp. 2088–2101, 2022.

[28] A. E. Roth, *The Shapley value: essays in honor of Lloyd S. Shapley*. Cambridge University Press, 1988.

[29] A. Prasad, S. Jegelka, and D. Batra, "Submodular meets structured: finding diverse subsets in exponentially-large structured item sets," in *Advances in Neural Information Processing Systems (NIPS)*, vol. 27, 2014.

[30] J. Bilmes and W. Bai, "Deep submodular functions," *arXiv preprint arXiv:1701.08939*, 2017.

[31] R. Balakrishnan, T. Li, T. Zhou, N. Himayat, V. Smith, and J. Bilmes, "Diverse client selection for federated learning via submodular maximization," in *International Conference on Learning Representations (ICLR)*, 2022.

[32] W. Liu, Y. Xi, J. Qin, X. Dai, R. Tang, S. Li, W. Zhang, and R. Zhang, "Personalized diversification for neural re-ranking in recommendation," in *2023 IEEE 39th International Conference on Data Engineering (ICDE)*. IEEE, 2023, pp. 802–815.

[33] C. J. Zhang, Y. Liu, P. Zeng, T. Wu, L. Chen, P. Hui, and F. Hao, "Similarity-driven and task-driven models for diversity of opinion in crowdsourcing markets," *The VLDB Journal (VLDBJ)*, pp. 1–22, 2024.

[34] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*, ser. CCS '16. Association for Computing Machinery, 2016, p. 308–318.

[35] N. Agarwal, A. T. Suresh, F. X. X. Yu, S. Kumar, and B. McMahan, "cpsgd: Communication-efficient and differentially-private distributed sgd," in *Advances in Neural Information Processing Systems*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds., vol. 31. Curran Associates, Inc., 2018.

[36] V. Nikolaenko, U. Weinsberg, S. Ioannidis, M. Joye, D. Boneh, and N. Taft, "Privacy-preserving ridge regression on hundreds of millions of records," in *IEEE Symposium on Security and Privacy (SP)*, 2013, pp. 334–348.

[37] R. L. Rivest, L. Adleman, M. L. Dertouzos *et al.*, "On data banks and privacy homomorphisms," *Foundations of secure computation*, vol. 4, no. 11, pp. 169–180, 1978.

[38] A. Benaissa, B. Retiat, B. Cebere, and A. E. Belfedhal, "Tenseal: a library for encrypted tensor operations using homomorphic encryption," *arXiv preprint arXiv:2104.03152*, 2021.

[39] T. Zhou and J. Bilmes, "Minimax curriculum learning: machine teaching with desirable difficulties and scheduled diversity," in *International Conference on Learning Representations (ICLR)*, 2018.

[40] B. Mirzasoleiman, J. Bilmes, and J. Leskovec, "Coresets for data-efficient training of machine learning models," in *International Conference on Machine Learning (ICML)*. PMLR, 2020, pp. 6950–6960.

[41] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher, "An analysis of approximations for maximizing submodular set functions," *Mathematical programming*, vol. 14, pp. 265–294, 1978.

[42] B. Mirzasoleiman, A. Badanidiyuru, A. Karbasi, J. Vondrak, and A. Krause, "Lazier than lazy greedy," *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, vol. 29, no. 1, 2015.

[43] B. Efron and D. V. Hinkley, "Assessing the accuracy of the maximum likelihood estimator: observed versus expected fisher information," *Biometrika*, vol. 65, no. 3, pp. 457–483, 1978.

[44] J.-X. Pan, K.-T. Fang, J.-X. Pan, and K.-T. Fang, "Maximum likelihood estimation," *Growth curve models and statistical diagnostics*, pp. 77–158, 2002.

[45] K. Wei, R. Iyer, and J. Bilmes, "Submodularity in data subset selection and active learning," in *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, vol. 37. PMLR, 07–09 Jul 2015, pp. 1954–1963.

[46] J. T. Wang, P. Mittal, and R. Jia, "Efficient data shapley for weighted nearest neighbor algorithms," in *Proceedings of the 27th International Conference on Artificial Intelligence and Statistics (AISTATS)*, vol. 238, 2024.

[47] Q. Feng, J. Zhang, W. Zhang, L. Qin, Y. Zhang, and X. Lin, "Efficient k nn search in public transportation networks," *Proceedings of the VLDB Endowment (VLDB)*, vol. 17, no. 11, pp. 3402–3414, 2024.

[48] D. Ouyang, D. Wen, L. Qin, L. Chang, Y. Zhang, and X. Lin, "Progressive top-k nearest neighbors search in large road networks," in *Proceedings of the International Conference on Management of Data (SIGMOD)*, 2020, p. 1781–1795.

[49] S. J. Sheather and M. C. Jones, "A reliable data-based bandwidth selection method for kernel density estimation," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 53, no. 3, pp. 683–690, 1991.

[50] G. R. Terrell and D. W. Scott, "Variable kernel density estimation," *The Annals of Statistics*, pp. 1236–1265, 1992.

[51] J. Kim and C. D. Scott, "Robust kernel density estimation," *The Journal of Machine Learning Research (JMLR)*, vol. 13, no. 1, pp. 2529–2565, 2012.

[52] R. Fagin, "Combining fuzzy information from multiple systems," in *Proceedings of the ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS)*, 1996, pp. 216–226.

[53] R. Fagin, A. Lotem, and M. Naor, "Optimal aggregation algorithms for middleware," in *Proceedings of the ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems (PODS)*, 2001, pp. 102–113.

[54] W. Jin and J. M. Patel, "Efficient and generic evaluation of ranked queries," in *Proceedings of the International Conference on Management of Data (SIGMOD)*, 2011, pp. 601–612.

[55] L. M. Haas, D. Kossmann, E. L. Wimmers, and J. Yang, "Optimizing queries across diverse data sources," *Proceedings of the VLDB Endowment (VLDB)*, vol. 97, pp. 25–29, 1997.

[56] S. Li, D. Yao, and J. Liu, "Fedvs: straggler-resilient and privacy-preserving vertical federated learning for split models," in *International Conference on Machine Learning (ICML)*. PMLR, 2023, pp. 20 296–20 311.

[57] X. Luo, Y. Wu, X. Xiao, and B. C. Ooi, "Feature inference attack on model predictions in vertical federated learning," in *2021 IEEE 37th International Conference on Data Engineering (ICDE)*. IEEE, 2021, pp. 181–192.

[58] X. Xu, W. Wang, Z. Chen, B. Wang, C. Li, L. Duan, Z. Han, and Y. Han, "Finding the piste: towards understanding privacy leaks in vertical federated learning systems," *IEEE Transactions on Dependable and Secure Computing (TDSC)*, 2024.

[59] F. Fu, X. Wang, J. Jiang, H. Xue, and B. Cui, "Projpert: projection-based perturbation for label protection in split learning based vertical federated learning," *IEEE Transactions on Knowledge and Data Engineering(TKDE)*, 2024.

[60] D. Dua and C. Graff, "Uci machine learning repository," UCI Machine Learning Repository, 2017, available online: https://archive.ics.uci.edu/ml.

[61] Kaggle, "Kaggle: your home for data science," Kaggle, 2024, available online: https://www.kaggle.com/datasets.

[62] C.-C. Chang and C.-J. Lin, "Libsvm: a library for support vector machines," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 2, no. 3, pp. 1–27, 2011.

[63] X. Zhou, X. Yan, X. Li, H. Huang, Q. Xu, Q. Zhang, Y. Jerome, Z. Cai, and J. Jiang, "Vfdv-im: an efficient and securely vertical federated data valuation," in *International Conference on Database Systems for Advanced Applications (DASFAA)*, 2024, pp. 409–424.

[64] D. P. Kingma and J. Ba, "Adam: a method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[65] S. Hardy, W. Henecka, H. Ivey-Law, R. Nock, G. Patrini, G. Smith, and B. Thorne, "Private federated learning on vertically partitioned data via entity resolution and additively homomorphic encryption," *arXiv preprint arXiv:1711.10677*, 2017.

[66] K. Cheng, T. Fan, Y. Jin, Y. Liu, T. Chen, D. Papadopoulos, and Q. Yang, "Secureboost: a lossless federated learning framework," *IEEE Intelligent Systems*, vol. 36, no. 6, pp. 87–98, 2021.

[67] K. Yang, T. Fan, T. Chen, Y. Shi, and Q. Yang, "A quasi-newton method based vertical federated learning framework for logistic regression," *arXiv preprint arXiv:1912.00513*, 2019.

[68] P. Vepakomma, O. Gupta, T. Swedish, and R. Raskar, "Split learning for health: distributed deep learning without sharing raw patient data," *arXiv preprint arXiv:1812.00564*, 2018.

[69] Z. Wu, Q. Li, and B. He, "A coupled design of exploiting record similarity for practical vertical federated learning," *Advances in Neural Information Processing Systems (NIPS)*, vol. 35, pp. 21 087–21 100, 2022.

[70] J. Sun, X. Yang, Y. Yao, A. Zhang, W. Gao, J. Xie, and C. Wang, "Vertical federated learning without revealing intersection membership," *arXiv preprint arXiv:2106.05508*, 2021.

[71] L. Huang, Z. Li, J. Sun, and H. Zhao, "Coresets for vertical federated learning: regularized linear regression and k-means clustering," *Advances in Neural Information Processing Systems (NIPS)*, vol. 35, pp. 29 566–29 581, 2022.

[72] T. Chen, X. Jin, Y. Sun, and W. Yin, "Vafl: a method of vertical asynchronous federated learning," *arXiv preprint arXiv:2007.06081*, 2020.

[73] F. Lai, X. Zhu, H. V. Madhyastha, and M. Chowdhury, "Oort: efficient federated learning via guided participant selection," in *USENIX Symposium on Operating Systems Design and Implementation (OSDI)*. USENIX Association, 2021, pp. 19–35.

[74] T. Nishio and R. Yonetani, "Client selection for federated learning with heterogeneous resources in mobile edge," in *IEEE International Conference on Communications (ICC)*, 2019, pp. 1–7.

[75] Y. Jee Cho, J. Wang, and G. Joshi, "Towards understanding biased client selection in federated learning," vol. 151, pp. 10 351–10 375, 2022.

[76] J. Zhang, J. Wang, Y. Li, F. Xin, F. Dong, J. Luo, and Z. Wu, "Addressing heterogeneity in federated learning with client selection via submodular optimization," *ACM Trans. Sen. Netw.*, vol. 20, no. 2, 2024.

[77] A. C. C. Jiménez, E. C. Kaya, L. Ye, and A. Hashemi, "Submodular maximization approaches for equitable client selection in federated learning," *arXiv preprint arXiv:2408.13683*, 2024.

[78] R. Zhang, Y. Wang, Z. Zhou, Z. Ren, Y. Tong, and K. Xu, "Data source selection in federated learning: a submodular optimization approach," in *International Conference on Database Systems for Advanced Applications*. Springer, 2022, pp. 606–614.

[79] P. Kolpaczki, V. Bengs, M. Muschalik, and E. Hüllermeier, "Approximating the shapley value without marginal contributions," *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2025.

[80] R. Jia, D. Dao, B. Wang, F. A. Hubis, N. M. Gurel, B. Li, C. Zhang, C. Spanos, and D. Song, "Efficient task-specific data valuation for nearest neighbor algorithms," *Proceedings of the VLDB Endowment (VLDB)*, vol. 12, no. 11, p. 1610–1623, 2019.

[81] T. Song, Y. Tong, and S. Wei, "Profit allocation for federated learning," in *IEEE International Conference on Big Data (Big Data)*, 2019, pp. 2577–2586.

[82] Q. Sun, X. Li, J. Zhang, L. Xiong, W. Liu, J. Liu, Z. Qin, and K. Ren, "Shapleyfl: robust federated learning based on shapley value," in *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2023, pp. 2096–2108.

[83] J. Huang, L. Zhang, A. Li, H. Cheng, J. Xu, and H. Song, "Adaptive and efficient participant selection in vertical federated learning," in *2023 19th International Conference on Mobility, Sensing and Networking (MSN)*, 2023, pp. 455–462.