Differentially Private Vertical Federated Clustering

Zitao Li
Purdue University
liz2490@purdue.edu

Tianhao Wang
University of Virginia
tianhao@virginia.edu

Ninghui Li
Purdue University
ninghui@purdue.edu

ABSTRACT
In many applications, multiple parties have private data regarding the same set of users but on disjoint sets of attributes, and a server wants to leverage the data to train a model. To enable model learning while protecting the privacy of the data subjects, we need vertical federated learning (VFL) techniques, where the data parties share only information for training the model, instead of the private data. However, it is challenging to ensure that the shared information maintains privacy while learning accurate models. To the best of our knowledge, the algorithm proposed in this paper is the first practical solution for differentially private vertical federated \(k\)-means clustering, where the server can obtain a set of global centers with a provable differential privacy guarantee. Our algorithm assumes an untrusted central server that aggregates differentially private local centers and membership encodings from local data parties. It builds a weighted grid as the synopsis of the global dataset based on the received information. Final centers are generated by running any \(k\)-means algorithm on the weighted grid. Our approach for grid weight estimation uses a novel, light-weight, and differentially private set intersection cardinality estimation algorithm based on the Flajolet-Martin sketch. To improve the estimation accuracy in the setting with more than two data parties, we further propose a refined version of the weights estimation algorithm and a parameter tuning strategy to reduce the final \(k\)-means loss to be close to that in the central private setting. We provide theoretical utility analysis and experimental evaluation results for the cluster centers computed by our algorithm and show that our approach performs better both theoretically and empirically than the two baselines based on existing techniques.

PVLDB Reference Format:
Zitao Li, Tianhao Wang, and Ninghui Li. Differentially Private Vertical Federated Clustering. PVLDB, 16(6): 1277 - 1290, 2023.
doi:10.14778/3583140.3583146

PVLDB Artifact Availability:
The source code, data, and/or other artifacts have been made available at https://anonymous.4open.science/r/public_vflclustering-63CD.

1 INTRODUCTION
Data privacy laws and regulations such as GDPR [2] and California Consumer Privacy Act [1] bring more restrictions and compliance requirements for the data collectors, including the companies and some government agencies. However, the demand for larger and more comprehensive datasets is increasing as political and business decisions become more and more reliant on different machine learning models. In many applications, data about entities are partitioned among multiple data parties and they cannot bring the data together, due to privacy restrictions. Federated learning (FL) with the cross-silo setting [41] is a computation concept that can enable these data parties to use their data to train useful models collaboratively without sharing the data. But FL by itself cannot provide any provable privacy guarantee in the sense that adversaries can still infer whether one user’s data is in the training set (i.e., membership attack [21, 45, 61]) or even recover the training data (i.e., reconstruction attack [9, 21, 79]) by examining the shared information from local data parties. As a result, FL needs to be deployed with other privacy techniques, such as those for satisfying differential privacy (DP) [19], to provide provable privacy guarantees.

This paper focuses on an important federated learning setting, vertical federated learning (VFL). Its difference from horizontal federated learning (HFL) is that all parties have data from the same set of users, but their data attributes are different from each other, while HFL assumes that all the data parties have data from different sets of users but all local datasets have the same attributes [51, 52, 74, 75]. VFL has been an interesting topic in the research area since the early 2000s [30, 67, 68, 76, 78]. The papers are usually motivated by medical or financial use cases, where the users’ private data are not allowed to be shared between data parties. More recently, VFL has been adapted by some fintech companies for more real-world services. For example, WeBank demonstrates how they do risk-control for car insurance cooperating with car rental companies with VFL techniques [73]. Compared with HFL, VFL tasks usually consider fewer data parties.

How to perform VFL while not leaking private information has been an interesting topic in the security and privacy community [20, 67, 68]. Many existing VFL approaches are based on secure multiparty computation (SMC), including learning classification tree models [48, 68, 76], regression models [29] and clustering models [67]. However, the SMC-based methods’ final results cannot provide provable resistance to membership or reconstruction attacks, and they usually have high computation and communication overheads. Other literature employs DP as the security notion to provide resistance to those attacks. More recently, researchers have developed VFL algorithms with DP guarantee for matrix factorization [46], regression [69], and boosting model [12]. We employ DP as the privacy notion for VFL clustering problem in this paper.

Many problems are more challenging in the VFL setting than in the central setting and the HFL setting. One example is the \(k\)-means clustering problem, in which desired solutions minimize the distances between user data points and their closest cluster centers. The \(k\)-means algorithms developed for the central DP setting [26, 66] require access to all dimensions of data points to compute distances for updating cluster centers. In the HFL setting,
We let each data party encode its memberships of the local clusters while we need intersection operations, we design an algorithm (Algorithm 4) with inclusion-exclusion rules for the server to estimate the number of data parties in VFL. Because FM sketches support only union operations, they are insufficient for our problem. To address this, we propose a heuristic estimation algorithm (Algorithm 5). It estimates the union cardinality of membership attributes. We also prove a heuristic method to choose the local clustering parameter for smaller final losses.

Our main contributions are summarized as follows:

1. We propose the first (according to our knowledge) differentially private VFL k-means algorithm with an untrusted central server. We let each data party encode its memberships of the local clusters into Flajolet-Martin (FM) sketches and share the partial views with the server. The server then runs a central k-means algorithm on its partial view. The server then runs a central k-means algorithm on its partial view. The server then runs a central k-means algorithm on its partial view. The server then runs a central k-means algorithm on its partial view. The server then runs a central k-means algorithm on its partial view. The server then runs a central k-means algorithm on its partial view.

2. The messages not only contain synopses of local information, but also can be “composed” by the central server to reconstruct correlations of inter-party features. The synopses of central DP k-means algorithms [26, 66] do not have such a “composability” property and the correlations between the inter-party attributes are lost from the central server’s view. We show that correlation retaining is essential for the VFL k-means problem, and the accuracy of the estimated correlations largely affects the final cost of k-means problem in our experiments.

Our contributions. This paper proposes a solution for the differentially private vertical federated k-means with multiple data parties and an untrusted central server. All the information shared by data parties in the process, as well as the final result, satisfy DP. The key idea is to have each data party generate a differentially private “data synopsis”, including the partial centers and encoded membership information that describes local k-means results based on its partial view. The server then runs a central k-means on the Cartesian product of all partial centers considering their weights, where the weight for each joint center is an estimate of the cardinality of intersection among users belonging to each partial center. Our main contributions are summarized as follows:

- We propose the first (according to our knowledge) differentially private VFL k-means algorithm with an untrusted central server.

- The cardinality estimation errors can grow very fast when the number of data parties in VFL increases. To improve the estimation accuracy when more than two data parties are involved, we propose a heuristic estimation algorithm (Algorithm 5). It estimates the local central clustering parameter for smaller final losses.

- Our experiments show that our proposed methods can outperform the other baseline methods and even approach the non-private VFL k-means algorithm when sufficient users are in the dataset. We also conduct ablation studies to empirically demonstrate the impact and effectiveness of each component of our algorithm.

Roadmap. We revisit the necessary background information in Section 2, we give an overview of the VFL clustering problem and our solution in Section 3, and provide more details of the key components in Section 4 and 5; experimental results are shown in Section 6; Section 7 discusses the related work from different perspectives, followed by a conclusion in Section 8. Because of the space limitation, the proofs and additional experimental results are provided in the appendix of the full version [47].

2 BACKGROUND

2.1 Differential Privacy

Definition 1 (Differential privacy [19]). A randomized algorithm $A$ is $(\varepsilon, \delta)$-differentially private if for any pair of datasets $X, X'$ that differ in one record and for all possible subset $O$ of possible outputs of algorithm $A$, $Pr[A(X) \in O] \leq e^{\varepsilon} Pr[A(X') \in O] + \delta$.

Three properties of DP are frequently used to build complicated algorithms. Assume that there are two subroutines $A_1(\cdot)$ and $A_2(\cdot)$ that can provides $(\varepsilon_1, \delta_1, (\varepsilon_2, \delta_2))$-DP protection. Sequential composition states that $A_1(X, A_2(X))$ satisfies $(\varepsilon_1 + \varepsilon_2, \delta_1 + \delta_2)$-DP. On the other hand, parallel composition states that combining two subroutines each only accessing a non-overlapping sub-dataset $X_1$ or $X_2$ satisfies $(\max(\varepsilon_1, \varepsilon_2), \max(\delta_1, \delta_2))$-DP. A third property, post-processing property states that, any data independent operation on an $(\varepsilon, \delta)$-DP algorithm’s result still satisfies $(\varepsilon, \delta)$-DP.

Laplace mechanism. One of the most classic DP mechanisms, Laplace mechanism, adds Laplace noise to the return of a function $f$ to ensure the result is differentially private. The variance of the noise depends on $GS_f$, the global sensitivity or the $L_1$ sensitivity of $f$, defined with a pair of neighboring datasets as, $GS_f = \max_{X, X'} ||f(X) - f(X')||$. The Laplace mechanism mechanism $A$ is formalized as $A_f(X) = f(X) + \text{Lap}\left(\frac{GS_f}{\varepsilon}\right)$, where Lap$(b)$ denotes a random variable sampled from the zero-mean Laplace distribution with scale $b$. When $f$ outputs a vector, $A$ adds independent samples of Lap$\left(\frac{GS_f}{\varepsilon}\right)$ to each element of the vector.

Tighter DP sequential composition. The notion of Rényi Differential Privacy (RDP) [53] provides a succinct way to track the privacy loss from a composition of multiple mechanisms by representing privacy guarantees through moments of privacy loss.

Definition 2 (Rényi Differential Privacy [53]). A mechanism $M : X \rightarrow Y$ is said to satisfy $(\nu, \tau)$-RDP if the following holds for any two neighboring datasets $X, X'$

$$\frac{1}{\nu - 1} \log E_{o \sim M}(X) \left( \frac{Pr[A(X) = o]}{Pr[A(X') = o]} \right)^\nu \leq \tau.$$

Fact 2.1 (RDP Sequential Composition [53]). If $A_1$ and $A_2$ are $(\nu, \tau_1)$-RDP and $(\nu, \tau_2)$-RDP respectively then the mechanism resulting from the two $g(A_1(X), A_2(X))$ is $(\nu, \tau_1 + \tau_2)$-RDP.

Fact 2.2 (RDP to $\epsilon, \delta$-DP [53]). If a mechanism is $(\nu, \tau)$-RDP, then it also satisfies $(\tau + \frac{\log 1/\delta}{\nu - 1}, \delta)$-DP.
With the sequential composition of RDP and the conversion to \((\epsilon, \delta)\)-DP, the privacy loss of \(M\) sequential mechanism can be improved from \(O(M\epsilon)\) to the order of \(O(\sqrt{M\epsilon})\).

### 2.2 k-means Clustering

The \(k\)-means problem \([49]\) is one of the most well-known clustering problems. With a parameter \(k\) and a dataset \(X \in \mathbb{R}^{nm}\), the goal of the problem is to output a set of \(k\) centers \(C\) that can minimize the distance of data points to the nearest centers. The cost (or loss function) is formalized as 
\[
\text{cost}_X(C) = \sum_{x \in X} \min_{c \in C} \|x - c\|^2.
\]

The cost function can be extended to weighted data sets, where each data point \(x\) has a weight \(w(x)\) associated with it. It is equivalent to the scenario having \(w(x)\) copies of the same data point \(x\) in \(X\). The cost becomes 
\[
\text{cost}_X(C) = \sum_{x \in X} w(x) \cdot \left(\min_{c \in C} \|x - c\|^2\right).
\]

Theoretically, there is always a set of optimal \(k\) centers and the cost is denoted as \(\text{OPT}_X = \min_{\|C\|=k} \text{cost}_X(C)\). However, finding the optimal set of centers is NP-hard \([6]\). Research interests usually fall on approximate algorithms with polynomial running time. For example, the most well-known algorithm, Lloyd’s algorithm \([32]\) has time complexity \(O(nnk)\). A notation, \((\beta, \lambda)\)-approximate, is used to describe the utility guarantee of an approximate algorithm, such that \(\text{cost}_X(C) \leq \beta \cdot \text{OPT}_X + \lambda\) with any \(X\) and \(k\), where \(\beta\) is called approximation ratio. The best known non-private algorithm has \(\lambda = 0\) and \(\beta = 1 + \eta\) for any fixed \(\eta > 0\) when \(k\) is a constant \([50]\); but it is unavoidable for \(DP\) \(k\)-means to have \(\lambda > 0\) \([26]\).

### 2.3 Cardinality Estimation Sketches

Sketches usually refer to a family of succinct data structures that can store some basic information about a large amount of data with very low space and time complexity. One of the most well-known sketches is the Flajolet-Martin (FM) sketch \([24]\), which is designed to estimate the cardinality (i.e., the number of distinct elements) of a (multi)set \(M\). In FM sketch, all the elements in \(M\) are hashed with \(H_x\) (\(\cdot\)), an ideal geometric-value hash function. The estimate of the cardinality is \((1+\gamma)^\alpha\), where \(\alpha = \max(H_x(x)|x \in M)\) and \(\gamma\) is the parameter of hash function. Typically, multiple (e.g., \(1000\)) hash functions \((H\) with different hash keys \(\zeta)\) are used, and we take the harmonic/geometric average of those maximums as the final \(\alpha\). One appealing advantage of FM sketch is that it is mergeable. With the same hash key, sketches from different (multi)sets can be merged by taking the maximum, and we can derive the estimate of the cardinality of the union of those (multi)sets. With this property, we can estimate the cardinalities of the union/intersection of the set in the federated setting without leaking private information.

A recent series of research results show that if the cardinality is large enough, a family of hash-based, order-invariant sketches, including FM sketch, can satisfy DP without adding any additional noise \([16, 35, 63]\). We will introduce more details in Section 4.2.

### 3 OVERVIEW OF PROBLEM AND APPROACH

In this section, we define the problem of differentially private \(k\)-means under vertical federated learning (VFL), provide an overview of our four-phase approach, and discuss the first phase solution. The problem of VFL \(k\)-means (without DP) has been studied before by Ding et al. \([18]\). We thus describe the approach in \([18]\), the new challenges when we need to satisfy DP, and our framework.

### 3.1 Problem Formulation

We formalize the VFL \(k\)-means clustering as the following.

**Vertical federated learning (VFL).** Federated learning \([41]\) focuses on learning tasks among multiple data parties without directly sharing their local data. VFL assumes that each data party’s data are with different features of the same set of users. Consider a global view of dataset \(X\), where each row corresponds to a user, and each column corresponds to a feature. The setting of VFL is that \(X\) is vertically split into \(X = [X^{(1)}, \ldots, X^{(S)}]\), so that each data party \(\ell \in [S]\) has a local dataset \(X^{(\ell)}\) with \(m^{(\ell)}\) features. We assume each user is labeled with a unique id (e.g., MAC address) and is consistent across all the data parties.

**Security model.** We assume that an untrusted central server orchestrates the process and derives the results. Different from the local setting of DP, we assume that the data parties have common interests in protecting their data privacy, and none of them collude with the server. We want to ensure that all the information shared by data parties is differentially private to the central server. Thus, the server cannot learn any private information, even if it is malicious. However, to ensure the usefulness of the final output, we need to assume that the server does not deviate from the algorithm.

**Goal.** All the \(S\) parties want to cooperatively generate differentially private \(k\) centers \(C\) in the full domain (with all attributes) that can approximately minimize the \(k\)-means loss \(\text{cost}_X(C)\). The challenge is that each party only has its local view (a few attributes), where the final centers are computed with a full view (all attributes).

### 3.2 A Non-Private Baseline

Ding et al. \([18]\) consider VFL \(k\)-means, and aim to avoid the data communication cost of sending all data to a server. A natural approach is thus first to construct a global approximation of the data points and then perform \(k\)-means clustering on the approximation. In the approach taken in \([18]\), each data party first finds local cluster centers, then reports to the server these local cluster centers together with which local cluster each data point belongs to. The central server can assemble the local centers and local clustering memberships to create a set of weighted pseudo data points of the full dataset. We provide more details below.

Each party \(\ell\) performs clustering to find \(k^{(\ell)}\) local cluster centers \(C^{(\ell)} = \{c^{(\ell)}_1, \ldots, c^{(\ell)}_{k^{(\ell)}}\}\) and then sends the \(k^{(\ell)}\) centers together with membership information \(I^{(\ell)} = \{M^{(\ell)}_1, \ldots, M^{(\ell)}_{k^{(\ell)}}\}\) to the central server, where \(M^{(\ell)}_a = \{1d \mid a = \arg \min \|x^{(\ell)}_d - c^{(\ell)}_a\|^2\}\). The server constructs \((k')^S\) pseudo data points as a grid from the Cartesian product of the local centers received from \(S\) parties, i.e.,

\[
G = \{(c^{(1)}_a, \ldots, c^{(S)}_a) \mid \forall (a_1, \ldots, a_S) \in [k')^S]\}
\]

and assigns the cardinality of the intersection of the corresponding clusters as weights to them such that \(w(G(a_1, \ldots, a_S)) = \sum_{a_1 \leq a_S} M^{(1)}_{a_1} \cap \ldots \cap M^{(S)}_{a_S}\).

This algorithm with only one round of communication can perform well because the grid built by the central server actually maintains most of the necessary information about the local datasets: each local center \(c^{(\ell)}_a\) is the exact average of the user data in \(M^{(\ell)}_a\).
As long as the weighted grid nodes are representative enough for a case of multiple parties. Algorithm 1 consists of four phases: 

Phase 1: Each party clusters local data and generates differentially private local centers (sub-procedure LocCluster).

Phase 2: Each party encodes the differentially private “membership information” of each local cluster with the private centers and user data points (sub-procedure MemEnc).

Phase 3: The central server first randomly queries a party for an estimate of the total number of users with the Laplace mechanism and a small privacy budget \(e_0\). Then the central server receives the private local clustering centers and local membership information of the local clusters. It builds a weighted grid, where grid nodes are the Cartesian product of different parties’ local centers, and have the estimate of intersection cardinality of the corresponding clusters as their weights (Line 3(c) and sub-procedure WeightEstimate).

Phase 4: The central server runs a known central \(k\)-means algorithm on the weighted grid to generate the final \(k\) centers.

\[\text{Algorithm 1 Private Vertical Federated Clustering}\]

Input: Local datasets \(\{X^f_i \in \mathbb{R}^{n \times m^f} \mid f \in [S]\}\), total privacy budget \((\epsilon, \delta)\) is divided as \(e_0 = (1 - b)\epsilon\), \(e_1 = \frac{b_1}{b_2}\), \(e_2 = \frac{b_2}{b_3}\), \(\delta_2 = \frac{\delta}{\delta_3}\) for each data party, \(k\) and \(k'\) for clustering, and auxiliary membership encoding parameters aux.

Output: A set of \(k\) centers \(\{c_1, \ldots, c_k\} \in \mathbb{R}^{k \times m}\)
1. Each data party \(f \in [S]\):
   (a): \(\{c_1^f, \ldots, c_k^f\} \leftarrow \text{LocCluster}(X^f, e_1, k')\)
2. Each data party \(f \in [S]\):
   (a): \(C^f \leftarrow \text{MemEnc}(X^f, \{c_1^f, \ldots, c_k^f\}, e_2, \delta_2, \text{aux})\)
   (b): sends \(C^f = \{c_1^f, \ldots, c_k^f\}\) and \(I^f\) to server
3. Central server:
   (a): uses \(e_0\) to estimate the total number of user \(\hat{n}\)
   (b): receives \((\{C^f, I^f\}\mid f \in [S]\)) from all parties
   (c): computes grid by Cartesian product \(G \leftarrow C^1 \times \ldots \times C^S\)
   (d): computes \(W(G) \leftarrow \text{WeightEst}(\hat{n}, e_2, \delta_2, [I^f\mid f \in [S]])\)
4. Central server:
   (a): computes and outputs \(\{c_1, \ldots, c_k\} \leftarrow k\text{-means}(W(G), G, k)\)

In what follows, Section 3.5 describes our approach for Phase 1, and Section 4 describes our approach for Phase 2 and 3.

3.5 Private Local Clustering

We review some existing solutions to generate private centers in the central setting and then explain our adaptation to our VFL setting.

DPLloyd. A straight-forward differentially private central \(k\)-means is the \(DPLloyd\) [8, 66]. In each iteration, the assignment step is the same as the non-private Lloyd algorithm, where each data point is assigned to the closest center produced from the previous iteration. The updating step ensures DP by 1) using the Laplace mechanism with sensitivity 1 to get the noisy count of data points assigned to the center, 2) using the Laplace mechanism with sensitivity \(r\) (it requires that all attributes are bounded in \([−r, r]\)) and a split privacy budget for each dimension to get the noisy sums of the data points assigned to the same center. The centers are updated as the averages of all data points in the same cluster with the noisy count and noisy sum. Every iteration consumes privacy budget for computing noisy sums and noisy counts.

DPLSF. There are two recently proposed algorithms for differentially private \(k\)-means with theoretical performance guarantees, one for the central setting [26] and one for the local setting [10]. Both algorithms are built on a theoretical concept called efficiently decodable net. But how to implement the efficiently decodable net in practice is still unclear. Therefore, the authors also propose a DP \(k\)-means algorithm based on locality sensitive hashing (LSH) forest [11] to approximate the effect of the efficiently decodable net. The central DP implementation is open-sourced [27]. The high-level idea is to partition the data points based on their LSH outputs, generate differentially private means and counts for these partitions, and finally run a (non-private) \(k\)-means algorithm on the means with counts as weights. We call this method DPLSF. We choose
DPLSF as the instantiation of LocCluster in this paper because it is shown to outperform other existing methods in experiments [27].

Adapting DPLSF to VFL setting. The implementation in [27] requires a known \( L_2 \) norm upper bound for the data points because of the usage of the Gaussian mechanism. However, assuming the \( L_2 \) norm upper bound for each user’s data may be unreasonable in the VFL setting because a user’s data are spread in different data parties’ datasets. Thus, we normalize each attribute to some ranges to restrict the sensitivity of the data points averaging operations of DPLSF, and apply the Laplace mechanism to provide DP guarantees. Note that normalizing different attributes to different ranges is essentially adjusting the weights of different attributes when computing the distances. To simplify the discussion and experiment settings, we let each data party normalize its attributes from one party are independent of those from all other parties. Following this idea, the private MemEnc only needs to generate a histogram of \( \left| M_{a_1}^{(1)} \right| \ldots \left| M_{a_S}^{(S)} \right| \) with Laplace mechanism and privacy budget \( \varepsilon_2 \). Denote the randomized histogram vector as \( \hat{f}^{(f)} \).

The central server’s sub-procedure WeightEst is \( w(G(a_1, \ldots, a_S)) = \hat{a} \prod_{f \in [S]} \frac{\hat{f}^{(f)}}{\hat{n}} \). We call this baseline as IND-LAP because it makes the independence assumption and uses the Laplace mechanism.

However, when the assumption of inter-party attributes independence fails, this cardinality estimation can be far from the ground truth and make the final centers far from optimal, because the correlation information between the inter-party attributes is completely lost. Thus, maintaining the inter-party attribute correlations is the main focus of improving the utility in general scenarios.

Baseline 2: Estimate weights based on local differential privacy protocols. Another choice for aggregating the cardinality information is to let the parties report each user’s membership information separately, instead of aggregating the membership information first and then reporting. When such reporting satisfies local differential privacy (LDP) for each user, it also satisfies DP for the whole local dataset. In this paper, we apply either the optimized local hashing (OLH) or the general random response (GRR) protocol in [70] (which is used depends on the privacy parameter \( \varepsilon_2 \) and the domain size), and name this approach as LDP-AGG. We set \( \epsilon_0 = 0 \) because LDP-AGG does not need to estimate the number of users.

Local memberships of a user in \( S \) different data parties can be seen as an \( S \)-dimension record. Thus, it is equivalent to randomizing each “dimension” of a user record independently with LDP protocols. After receiving all the local memberships of a user, the server first computes the probability vector of this user in all possible intersections \( M_{a_1}^{(1)} \cap \ldots \cap M_{a_S}^{(S)} \). Then the server sums the probability vectors of all users to get the desired weights. The correlation of each user’s attributes is preserved because the server first aggregates all local memberships of each user. We defer more details of LDP-AGG in the appendix of our full version [47].

However, the reported memberships are very noisy. Based on the known LDP protocol error analysis [71, Proposition 10], the variance of an estimated cardinality with LDP-AGG is in the order \( O\left(\frac{n}{e_2^2}\right) \) for each intersection. The noise can easily overwhelm the true counts when \( e_2 \) is small or \( S \) is large.

4.2 Prerequisite: DP FM Sketch

As is shown, neither baseline is satisfactory. An effective approach for privacy-preserved MemEnc and WeightEst should accurately maintain most of the inter-party correlation information. We propose a new approach based on the Flajolet-Martin (FM) sketch...
because it can satisfy DP with a little additional overhead and support the set union operation.

**FM sketch achieves DP.** As mentioned in Section 2, FM sketches are used to estimate cardinality. Recently, some research results show that a family of sketches, including FM sketch, satisfy DP as long as the cardinality is large enough and the hash keys are unknown to the adversary [13, 16, 36, 63]. The DP version of the FM sketch algorithm is described as Algorithm 2 following the approach in Smith et al. [63], where the FM sketch is implemented using an ideal geometric-value hash function $H : \mathcal{X} \times \mathbb{Z} \rightarrow \mathbb{N}$. The above theorem is based on the following lemma about the geometric distribution parameter $\gamma$, an ideal random hash function $H_\gamma(x)$ when $\zeta \sim \text{Uniform}(\mathbb{Z})$.

Algorithm 2 DP FM Sketch Generation DPFM [63]

**Input:** a (multi)set $M$, privacy parameter $\epsilon'$, and geometric distribution parameter $\gamma$, an ideal random hash function $H_\gamma(x)$

**Output:** Sketch $\alpha$ for cardinality of $M$

1. $n_\gamma = \left\lfloor \frac{1}{e^{\gamma - 1}} \right\rfloor$, $a_{\min} = \log_{1 + e^{-\gamma}} \frac{1}{\gamma}$
2. $\alpha_p = \max\{Y_1, \ldots, Y_n\}$ where $Y_i \sim \text{Geometric}(\frac{\gamma}{1 + e^{\gamma}})$
3. $\alpha_{real} = \max\{H_\gamma(id) | id \in M\}$
4. Return $\alpha = \max\{\alpha_p, \alpha_{real}, a_{\min}\}$

Algorithm 2 generates a DP FM sketch. The intuition of the non-private FM sketch is that when elements in $M$ are encoded as a set of geometric random variables and $\alpha_{real} = \max\{H_\gamma(id) | id \in M\}$, we can expect $(1 + \gamma)^{\alpha_{real}} \in \left[\frac{|M|}{1 + e^{\gamma}}, (1 + \gamma) \cdot |M|\right]$ with reasonable probability. Compared with the non-private version, the DP FM sketch needs two additional steps to ensure privacy: adding phantom elements, and lower-bounding the output by $a_{\min}$. The phantom elements are used to ensure that the cardinality estimated by the final output is at least $n_\gamma$; the $a_{\min}$, which is at the $e^{-\epsilon'}$ quantile of the Geometric distribution, is used to ensure a probability that none of the items affects the output. It has been shown that the harmonic/geometric mean of $M$ runs of Algorithm 2 satisfies DP:

**Lemma 1 (Privacy Guarantee of DPFM [63]).** Given an ideal geometric-value hash function $H$, Algorithm 2 is $\epsilon'$-DP. Besides, repeating it $M$ times with different hash keys and $\epsilon' = \frac{\epsilon}{4M \log(1/\delta)}$ satisfies $(\epsilon, \delta)$-DP as long as $\epsilon \leq 2 \log(1/\delta)$.

### 4.3 Sketch-based MemEnc and WeightEst

Unlike the DP FM sketch [63] which aims to estimate the cardinality of a single set, our task is to encode the partitions, namely multiple local clusters consisting of user ids assigned to different local centers. Thus, we extend the FM sketch to encode the partition memberships, called Differentially Private Flajolet-Martin Partition Sketch (DPFMPS). We call an FM sketch vector for $\left\{\mathcal{M}_1^{(t)}, \ldots, \mathcal{M}_k^{(t)}\right\}$ as a set of FM sketches. The sketch generation function, DPFMPS-Gen (Algorithm 3), is an instantiation of MemEnc. The data parties need to share a set of auxiliary parameters, $\alpha = \{y, M, \zeta = \{\zeta_1, \ldots, \zeta_M\}\}$, where $y$ is the geometric distribution parameter, $M$ is the number of sets of sketches and each $\zeta_i \sim \text{Uniform}(\mathbb{Z})$ is the hash key to an ideal geometric-value hash function for the $i$-th set of sketch. Notice that all data parties need to share the same set of hash keys. Since the hash keys must be unknown to the central server to achieve DP with the FM sketches, each data party can generate a random number and share it with all other parties via some secure peer-to-peer channels (e.g., key-exchange protocol [17]); then, each data party can use the sum/XOR/concatenation of those $S$ random numbers as a hash key. Algorithm 3 generates $M$ sets of FM partition sketches and has the following privacy guarantee.

**Theorem 2.** Given an ideal geometric-value hash function $H$, DPFMPS-Gen (Algorithm 3) generating $M$ sets of partition sketches satisfies $(\epsilon_2, \delta_2)$-differential privacy.

The above theorem is based on the following lemma about the privacy guarantee for each set of sketch, namely $A_1$ in Algorithm 3.

**Lemma 3.** Given an ideal geometric-value hash function $H$, each set of the partition sketch (i.e., a row $A_1$) is $\epsilon'$-DP.

With Lemma 3, the privacy guarantee claimed in Theorem 2 can be derived with the sequential composition of RDP [53] and converted back to $(\epsilon, \delta)$-DP following the same proof as in [63].

Algorithm 3 DPFMPS-Gen

**Input:** A set of $k'$ centers $C^{(t)}$, dataset $X^{(t)}$, privacy parameter $\epsilon_2$ and $\delta_2$, Geometric distribution parameter $y$, number of sketches $M$ and the corresponding hash keys $\zeta = \{\zeta_1, \ldots, \zeta_M\}$

**Output:** $M$ sets of sketches $A$

1. $\epsilon' = \frac{\epsilon_2}{4M \log(1/\delta)}$
2. Generate $\left\{\mathcal{M}_1^{(t)}, \ldots, \mathcal{M}_k^{(t)}\right\}$ where $id \in \mathcal{M}_j^{(t)}$ based on $C^{(t)}$
3. $A \leftarrow 0^{M \times k'}$
4. for $i \in [M], a \in [k']$ do
5. $A_{i,a} = \text{DPFM}((\mathcal{M}_a^{(t)}, \epsilon', y, \zeta_i)$
6. end for
7. Return $A$

**Estimate intersection cardinality.** One key observation of the local partition is that an id can be clustered to one and only one $\mathcal{M}_a^{(t)}$ by each data party $t$. Thus, the intersection problem can be transformed into the union problem by an extension of the inclusion-exclusion principle:

$$\bigcap_{t=1}^{S} \mathcal{M}_{\alpha t}^{(t)} = \bigcup_{t=1}^{S} \mathcal{M}_{\alpha t}^{(t)} = \bigcup_{t=1}^{S} \bigcup_{a \neq \alpha t} \mathcal{M}_{a t}^{(t)}. \quad (1)$$

Algorithm 4 gives the detailed procedure. There are $(k')^S$ possible intersections that need to be estimated, and there are $M$ sets of FM sketches. In Line 1, $U$ is initialized to store the FM sketches for the cardinalities of the complementary set of intersections; $w_\alpha$ is an intermediate vector representing the cardinalities of union of complements, i.e., $\bigcup_{t=1}^{S} \mathcal{M}_{\alpha t}^{(t)}$.

To estimate the cardinalities of the intersection with Equation (1), we first need to calculate the sketch of their complementary set, i.e.,
Algorithm 4 DPFMPS-BasicEst

Input: Estimate of the total number of users \( \hat{n} \), Privacy parameter \( \epsilon_2 \) and \( \delta_2 \), the FM partition sketches \( \{A^{(1)}, \ldots, A^{(s)}\} \)

Output: Estimate of the weights \( w(G) \)

1: \( U \leftarrow 0^{\epsilon \times \ell(n/\delta_2)}, w' \leftarrow 0^{\epsilon \times \ell(n/\delta_2)} \)
2: \( \epsilon' = \frac{\epsilon_2}{\sqrt{\log(1/\delta_2)}} \)
3: for \( i \in [M], (a_1, \ldots, a_s) \in [k']^s \) do
4: \( U_i(a_1, \ldots, a_s) = \max \left\{ A_{i,a}^{(s)} \mid \ell \in [s], a \neq a' \right\} \)
5: end for
6: for \( (a_1, \ldots, a_s) \in [k']^s \) do
7: \( \alpha_{(a_1, \ldots, a_s)} = \text{HarmonicMean} \left( U_i(a_1, \ldots, a_s) \right) \)
8: \( w'_{(a_1, \ldots, a_s)} = (1 + y)\alpha_{(a_1, \ldots, a_s)} - s(k' - 1)\eta \)
9: \( w(G(a_1, \ldots, a_s)) = \hat{n} - w'(a_1, \ldots, a_s) \)
10: end for
11: Ensure \( \sum_{(a_1, \ldots, a_s)} w(G(a_1, \ldots, a_s)) = \hat{n} \) and \( w(G(a_1, \ldots, a_s) \geq 0 \)
12: Return \( w(G) \)

The corresponding FM sketch can be obtained by taking the max of all other elements in the same sketch set, \( \max \left\{ A_{i,a}^{(s)} \mid a \neq a' \right\} \), according to the union property of FM sketch. Next, we need to derive the sketch for the union of the complementary partition of all data parties, i.e. \( \bigcup_{i=1}^{M} A_{i, a}^{(s)} \). This union’s FM sketches can be obtained by max \( \max \left\{ A_{i,a}^{(s)} \mid a \neq a' \right\} \). Merging the two max operations gives us the operation in Line 4.

We use the Harmonic mean over the M FM sketches to estimate the cardinality of \( \bigcup_{i=1}^{M} A_{i, a}^{(s)} \) in Line 7, because it is shown [63] that this sketch mean estimate is more stable and accurate. As the cardinality of \( M \) operation gives us the operation in Line 4.

4.4 Privacy, Utility and Communication Cost

The proofs of the theorems in this subsection are provided in the appendix of the full version [47] because of the space limitation.

Privacy guarantee. According to the privacy spitting strategy in Algorithm 1, Theorem 2 and the sequential composition of DP, the following privacy statement can directly derived to describe the privacy loss from all data parties to the central server, and equivalently, the final output of the central server.

Theorem 4. Algorithm 1 is \((\epsilon, \delta)\)-DP with DPFMPS-Gen and DPFMPS-BasicEst as the implementation of MemEnc and WeightEst.

Error analysis of the weights. The cardinality estimate with Algorithm 2 approximates the real cardinality within a factor of \( 1 \pm \gamma \) and an additive error of \( O\left(\sqrt{\ln(1/\delta)}/\epsilon\right) \) [63]. The additive error is because of the phantom elements, and \( \sigma_{min} \) also slightly increases the expectation of the sketch. We provide a refined result to show that the utility of the private FM sketches generated from Algorithm 2 is similar to the non-private FM sketch when the real cardinality is large enough, in order to use the more advanced results from non-private FM sketch research.

Lemma 5. Let \( \hat{\alpha} = \max\{\alpha_{true}, \alpha\} \) and \( \alpha \) be the return of Algorithm 2. If \( H \) is an ideal geometric-value hash function and \( \epsilon' \) is fixed, we have \( \mathbb{E}\left[\alpha\right]/\mathbb{E}\left[\hat{\alpha}\right] \rightarrow 1 \) and \( \text{Var}\left[\alpha\right]/\text{Var}\left[\hat{\alpha}\right] \rightarrow 1 \) when \( |M| \) is sufficiently large.

With Lemma 5, we can claim that the lower bound \( \sigma_{min} \) does not affect the mean and variance of the sketch \( \hat{\alpha} \) too much. So we can treat the \( \alpha \) returned by Algorithm 2 approximately the same as the vanilla non-private FM sketch, and we can use the standard deviation results of the non-private FM sketch to analyze the error of DPFM in a more fine-grained way.

The standard deviation of a non-private FM sketch estimation can be represented as \( \rho N/\sqrt{M} \), where \( \rho \) is a constant, \( N \) is the cardinality and \( M \) is the repetition [23, 24, 44]. Our algorithm has \( s(k' - 1) \) union operations to derive an element of \( U \) and each set has \( \eta \) phantom elements. So the cardinality estimated by \( U_i(a_1, \ldots, a_s) \) becomes \( N(a_1, \ldots, a_s) = \bigcup_{i=1}^{M} \bigcup_{j} A_{i,a}^{(s)} \bigcup_{s(k'-1)\eta} = \hat{n} - w^*(G(a_1, \ldots, a_s) \) + \( s(k'-1)\eta \), where \( w^*(G(a_1, \ldots, a_s) \) represent the true intersection cardinality of \( |A_{a_1} \cap \cdots \cap A_{a_s}| \). Based on the property of the non-private FM sketch and the value of \( M \) and \( \eta \), we can state the following lemma.

Theorem 6. Given a constant \( \rho \) such that the non-private FM sketch’s standard deviation is \( \rho N/\sqrt{M} \) where \( N \) is the cardinality, and \( M \) is number of repetitions. With \( \eta \) and \( \epsilon' \) as set in Algorithm 4, each intersection cardinality estimate generated has standard deviation

\[ \sigma(a_1, \ldots, a_s) = \rho(n - w^*(G(a_1, \ldots, a_s))) + 4ps(k'-1) \sqrt{\log(1/\delta)} \frac{\sqrt{M}}{\epsilon_2} \]

The result is directly derived after plugging in the value of \( N(a_1, \ldots, a_s) \) and \( \eta \), and use the approximation \( e^y \approx y + 1 \) when \( y \) is a small positive number.

Final utility cost. When we set \( k' = k \) as indicated in the non-private setting [18], we can show that Algorithm 1 is a \((\beta, \lambda)\)-approximation algorithm with assumption of accessibility to some guaranteed central (private) \( k \)-means algorithm.

Theorem 7. Assume the data parties have access to a differentially private \((\beta_{priv}, \lambda_{priv})\)-approximate \( k \)-means algorithm, and the central server has access to a \((\beta_0, \lambda_0)\)-approximate \( k \)-means algorithm. Algorithm 1 with DPFMPS-Gen and DPFMPS-BasicEst is a \((\beta, \lambda)\)-approximation algorithm with probability \( 1 - \epsilon \), where \( \beta = 2\beta_{priv} + 4\beta_0 + 4\beta_0\beta_{priv} \), \( \lambda = 2(\beta_0 + 1)S\lambda_{priv} + \left( \beta_0 m^2 k^{1.55} \frac{n}{\sqrt{\delta}} + 2s(k-1)\sqrt{\log(1/\delta)} \right) \frac{\sqrt{M}}{\epsilon_2} \).
We introduce two heuristic methods in this section to improve the accuracy of the weight estimation when the number of data parties increases. As the number of data parties increases, the accuracy of the weight estimation will decrease dramatically. From the error analysis of Theorem 7, we propose to deal with this challenge by 1) computing only all pair-wise intersection cardinaities, and 2) using these two-party intersection cardinalities with Equation (2) as constraints, and iteratively update the \( w(G) \) to fulfill all these constraints.

The improved estimation is described in Algorithm 5 DPFMPS-2P Est. The central server first estimates the single-party cardinality of all local clusters with only \( A^i \) (Line 3). Namely, \( w^i \) is an estimate for \( |A^i_{a^i} \vert \). Then the server initializes the full grid weights \( w(G) \) with only the single-party cardinality information assuming no correlation between the inter-party attributes (Line 5). Next, the server estimates all the two-party weights with DPFMPS-BasicEst as a sub-procedure (Line 7). To be more detailed, the grid weights \( w(G^{(i,l)}) \) returned by DPFMPS-BasicEst with \( A^i \) and \( A^l \) are the estimates for \( \{ |A^i_{a^i} \cap A^l_{a^l} | \mid a^l_1,a^l_2 \in [k']^S \} \). With all pairs of \( w(G^{(i,l)}) \), the server iteratively updates the grid weights \( w(G) \) to make them consistent with all the two-party intersection cardinalities \( w(G^{(i,l)}) \). In each iteration, the server first randomly selects a pair of data parties (Line 10), groups the current full grid weights \( w(G) \) by the cluster indices of the chosen two parties, and sums the weights in the same group to generate a two-party grid weights \( \tilde{w}(G^{(i,l)}) \) according to Equation (2):

\[
\tilde{w}(G^{(i,l)}_{a^i_1,a^l_2}) = \sum_{(a^l_1,a^l_2) \in [k']^S} w(G_{a^l_1,a^l_2})
\]

We use the differences between \( w(G^{(i,l)}) \) and \( \tilde{w}(G^{(i,l)}) \) to update the weight evenly with a step size \( \eta \) (Line 15). After sufficient update iterations, the full-party grid weight \( w(G) \) is expected to approximately satisfy the constraints (Equation (2)) with all two-party weights \( w(G^{(i,l)}) \). Because both DPFMPS-2P Est and DPFMPS-BasicEst are post-processing components in the DP definition, the privacy guarantee in Theorem 4 still holds for DPFMPS-2P Est.

5.2 Auto-adjusted \( k' \)

The utility result of Theorem 7 is given by assuming the number of local and central clusters is the same, i.e., \( k' = k \). However, we can see a trade-off on the value of \( k' \). In the non-private setting, the larger the \( k' \) is, the more local dataset information is preserved for the final central clustering. On the other hand, the larger the \( k' \) is, the more fine-grained the grid becomes, and the fewer records (or smaller \( w^i(G_{a^l_1,a^l_2}) \)) are expected to be assigned to the grid node. According to Theorem 6, a larger \( k' \) can make the error of \( w(G_{a^l_1,a^l_2}) \) larger and increase the final cost.

Giving a closed form solution for the local \( k' \) to minimize the final loss is difficult. However, as we can see from the error bound of Theorem 7, the grid quality reflected by the second term of \( \lambda \) plays an important role. We propose an empirical rule to set \( k' = \max(k_0,k^{1/5}) \) for DPFMPS-2P Est to prevent the true cardinalities from being overwhelmed by the noise, where \( k_0 \) is the smallest integer that satisfies \( 2\sigma(a_1,...,a_s) \geq \frac{\beta}{k_0} \). Because \( w^i(G_{a^l_1,a^l_2}) \) is the sum of the weights in the same group to generate a two-party grid weights \( \tilde{w}(G^{(i,l)}) \) according to Equation (2), we set \( \rho = 0.649 \) according to [44].

We experimentally demonstrate that such a choice of \( k' \) can be good choices for the datasets in our experiments in Section 6.2.
6 EXPERIMENTS

Datasets. We use four different datasets in our experiments, pre-process the data so that all attributes are normalized to $[-1, 1]$.

Synthetic mixed Gaussian dataset with $k$ centers. To echo the implicit assumption of $k$-means problem, we first synthesize a mixed Gaussian dataset with $m=8$ for evaluation. We generated this data in a similar way as [10] but enforced each dimension’s range to be $[-1,1]$ instead of in a $L_2$ ball. We first randomly sample $k=5$ centers in the domain, then randomly sample $n=20,000$ data points from the Gaussian distribution with the means at those $k$ centers.

New York Taxi dataset [4]. This taxi dataset contains 8 attributes and 100,000 records of taxi trips information, including pick-up/drop-off times and locations. We preprocess the pick-up/drop-off times to make them a number indicating the time in a week, ranging from 0 to 60 $\times$ 60 $\times$ 24 $\times$ 7 before normalizing them.

Loan dataset [3]. The original Loan dataset has 120 attributes. We extract the first 60k records and 16 numerical attributes of the applicant’s credit information and apartment information. We clip these attributes’ values and make them upper-bounded by their original 95% quantile to eliminate the outliers.

Letter dataset [62]. This dataset consists of information about black-and-white rectangular pixels displayed as the letters in the English alphabet. There are 16 attributes and 20,000 records in the datasets.

We randomly partition the attributes of the mixed Gaussian dataset and the Loan datasets into $S$ parties; we split the attributes of the Taxi and Letter dataset with high correlation into different parties to simulate the worst case of information loss in the VFL.

Parameter settings. There are different methods to decide the best $k$ value for the real-world datasets. The Silhouette method [60] is one of the most commonly used. Silhouette measures how closely data points are matched to data within its cluster and how loosely they are matched to data of the neighboring cluster. The datasets we use in this paper all have relatively high Silhouette scores when $k$ is smaller. Thus, we fix $k = 5$ for experiments in this paper to eliminate the effect of $k$ unless we explicitly mention it.

According to the experimental results in [63], a larger number of repeated FM sketches (hyper-parameter $M$ in our paper) leads to a smaller relative error of cardinality estimation. Thus, we set $M = 4096$ and $\delta = 1/n$ as default in different experiments, making the communication and privacy cost reasonable in the cross-silo FL setting and achieving stable accuracy.

Evaluation Metric. We evaluate the quality of the final output with two metrics. The first is the normalized central $k$-means loss with all the data points, i.e., $\frac{1}{n} \sum_{x \in X}^{min_{a \in C} \|x-c\|^2}$, which measures how representative the final centers are. The second is the V-score [59] in the scikit-learn package [58], which measures how consistent the clustering results of VFL algorithms are when compared to the ground truth classes (for the synthetic dataset) or central non-private $k$-means results (for the real word datasets). The V-score is the harmonic mean of two conditional entropy scores measuring the homogeneity and completeness. It is a score between 0 and 1, and the closer to 1 the better. Because of the space limitation, we refer readers to [59] for detailed formulas.

Compared methods. The adapted DPLSF is used as the LocCluster in all private VFL experiments. We name the end-to-end private VFL clustering method based on their MemEnc and WeightEst instantiations. Our experiments compare the following methods:

- our proposed method DPFMPS-BasicEst and DPFMPS-2PEst;
- two baseline methods IND-LAP and LDP-AGG-2PEst (LDP-AGG-2PEst is an improved version of LDP-AGG using the same technique in Section 4.1 because directly applying the LDP-AGG gives us a very large loss when $S > 2$);
- the central private $k$-means method (central) DPLSF from [27];
- the central non-private $k$-means from the scikit-learn package [58];
- a VFL non-private implementation following [18].

6.1 End-to-end Comparison

We first demonstrate the end-to-end performance of the algorithm and show the advantages of our proposed algorithms.

$k$-means loss results. Figure 3 shows the loss of the final centers produced by different algorithms. As we can see, the losses of non-private vertical federated $k$-means are higher than the that of the central version in most cases. Notice that we set $k' = k = 5$ for the non-private VFL algorithm, so there are 25 or 625 grid nodes when $S = 2$ or $S = 4$ accordingly. It means some information is lost when we building the grid, and a more fine-grained grid can improve the utility in the non-private setting if we compare the figures in the first rows with the ones in the second row. In some cases, the
central DPLSF outperforms the non-private VFL baseline because the noise with a large privacy budget has a smaller impact on the final results than the information loss of representing the local data with local centers and memberships.

Comparing our proposed method, DPFMPS-2PEst, with the two baseline methods and DPFMPS-BasicEst, we can see that our proposed method can produce final centers with consistently lower losses for most settings. When there are only two parties \((S = 2)\), DPFMPS-BasicEst has the same performance as DPFMPS-2PEst because they are the same. But DPFMPS-2PEst largely improves over DPFMPS-BasicEst when \(S = 4\). The exception is the outcomes of the IND-LAP approach on the Letter dataset (Figure 3(d) and 3(h)), where IND-LAP approach has slightly better performance when \(\epsilon\) is small. That is because the attributes of the Letter dataset are relatively independent and do not follow the underlying assumption of k-means algorithm. As for the LDP baseline, LDP-AGG-2PEst, we can see that its performance is close to our proposed method only with large privacy budget but is inferior when \(\epsilon\) is small, which echoes with the theoretical results in Section 4. When the privacy budget is large enough, the LDP protocol has little randomness to perturb the local membership of a user. In contrast, the FM sketch has its inherent randomness, even with large privacy budgets. However, one may also notice that the empirical losses are closer to the non-private ones than the theorem indicates. It is because the accuracy of FM sketches is usually better than its theoretical guarantee, especially after normalization (Line 11 of Algorithm 4) [44, 63].

Moreover, comparing the private central with the private VFL algorithms, we can see that the private VFL almost always has a higher cost than the central one. The gap exists because of two different reasons. 1) Only the local centers and the sketches are shared with the central server in VFL, so some information is lost. 2) When we use sketches to encode the cardinalities of the intersections, privacy budgets are split to both differential private local clustering and generating FM sketches.

V-scores. When generating the synthetic mixed Gaussian dataset, we assign the same label for the data points drawn from the same center. However, the real-world datasets have no label, or the number of labels is different from our experiment setting, so we apply the central non-private k-means algorithm to generate the labels.
We perform the following ablation studies to demonstrate the impact of enforcing privacy on different components.

### 6.2 Ablation Study of Components

#### Intersection cardinality accuracy comparison.

To break-down the error, the first interesting metric is the relative accuracy of the intersection cardinality estimation. The relative error is defined as

$$\frac{1}{n} \sum w(G(a_1,...,a_S)) - w^*(G(a_1,...,a_S)).$$

We fix $k = k' = 5$ for fair comparison of all methods and the results are shown as Figure 7. Because the evaluation is based on the intermediate results of the end-to-end private algorithm, only half of the privacy budgets are spend on the intersection estimation. We can see that our proposed method DPFMPS-2PEst outperforms other methods in most experiment settings. The DPFMPS-BasicEst component or the intersection estimation component. For "DP + non-priv kmeans" experiments, each data party uses non-private central $k$-means to generate local centers and spends $\epsilon_2$ on the intersection estimation algorithm. For "DPLSF + non-priv intersection" experiments, each data party spends $\epsilon_1$ privacy budget on DPLSF to generate local centers, and the intersection cardinalities are computed exactly without enforcing any privacy.

#### Distinguishing the impact of private LocCluster from private intersection cardinality estimation.

In Figure 8, we compare the impact of enforcing privacy on either the local clustering component or the intersection estimation component. For "non-priv kmeans + DPFMPS-2PEst" experiments, each data party uses non-private central $k$-means to generate local centers and spends $\epsilon_2$ on the intersection estimation algorithm. For "DPLSF + non-priv intersection" experiments, each data party spends $\epsilon_1$ privacy budget on DPLSF to generate local centers, and the intersection cardinalities are computed exactly without enforcing any privacy.

As shown Figure 8, enforcing the LocCluster with DP but using the non-private intersection cardinality estimation gives the cost closer to the end-to-end private ones when $S = 2$. However, when $S = 4$, making either component non-private while enforcing the other private has a similar effect on the final loss. These results show that when $S$ is small, the private LocCluster (related to the first term of additive error $\lambda$ of Theorem 7) is the component that introduces the majority error. However, they also support our analysis in Section 4.4 that the intersection component will introduce a larger error when the number of parties $S$ increases.
Impact of different local $k'$. As we discussed in Section 5.2, in order to trade-off between the error introduced in the membership intersection cardinality estimation with the information loss of local clustering, we can adjust the number of local clusters, namely $k'$. In Figure 9, we compare the impact of different numbers of local clusters on the final cost. The empirical optimal $k'$ for different privacy budgets are close to our heuristic choice. If $\epsilon$ is large, larger $k'$ can benefit the final result by keeping more local information; if $S$ is large, then it would be better to choose a smaller $k'$ to limit the exponentially grown number of grid nodes.

Communication/computation cost. The communication cost of each data party in our method is determined by two parameters, the number of local clusters $k'$ and the repetition of FM sketches $M$. We record the communication cost of some combinations in Table 1 with the mixed Gaussian dataset. As we can see, IND-LAP has the smallest communication cost, at the expense of ignoring all inter-party correlation. Our method can have a smaller communication cost than the LDP-AGG-2PEst and the non-priv if $n > Mk'$. With larger $k'$ and $S$ (e.g., $k' = 8$ and $S = 4$), the DPFMPS-2PEst and LDP-AGG-2PEst require more iterations to achieve the convergence of the estimates. This iterative update dominates the computation time, but the computation can still be done efficiently.

7 RELATED WORK

We summarize the related work from the following three axes.

Vertical federated learning. To our knowledge, this paper is the first work that targeting at the clustering problem under vertical federated learning with DP guarantee. VFL has been studied in the recent years, sometimes under the name of vertical distributed learning. The most relevant paper is [18], which proposes a solution for the clustering problem but does not consider the privacy leakage issue. Existing work about other problems in the VFL setting includes learning tree models with secure multiparty computation techniques [48, 76] and training composed models [37]. Some recent papers [36, 77] apply the ADMF framework in the VFL setting. Another paper [12] discussing asynchronous supervised learning with VFL and DPFSGD [5] assumes the labels are public accessible.

Data sketches with DP and private set intersection cardinality estimation. It is originally claimed in [15] that cardinality estimators, including FM sketch as its variants, leak the membership of a user in a set. However, their result assumes that the adversary knows the hash key, which is not a common DP security setting. Because DP security is based on the assumption that the adversary does not know the random seed; otherwise the adversary can regenerate the randomness (i.e., Laplace noise) by itself and recover the true value. Recently, several papers [13, 16, 35, 63] reveal that hash-based, order-invariant sketches satisfy differential privacy as long as the cardinality set is large enough. An earlier paper [37] uses linear sketch and perturbs the sketch with random response, but it has larger relative error [63] and assumes no duplicate.

While the private set intersection cardinality (PSI-CA) problem is a traditional cryptographic problem and has been studied in series of literature [14, 25, 33, 34, 39, 42], there are few papers about how to solve it under DP. To solve the PSI-CA problem with DP guarantee, the existing solutions also rely on some kinds of data sketch. For example, [64] proposes a solution to support set union or intersection with an untrusted third party based on Bloom filters. However, their solution introduces large randomness and is unable to scale to the setting with more than 2 parties. Other similar work includes using cryptographic tools to encrypt the sketches as [43]. Some other cryptography oriented research [28, 40] use DP definition as a relaxation of the traditional security definition, and develop secure PSI protocol resisting malicious adversary.

Differential private $k$-means. In the central DP setting, some earlier papers, including [8, 22, 38, 54–56, 65, 72], contribute to the theoretical bound for the $k$-means cost. A practical implementation [7] with cost bounded is based on privately selecting a candidate center set and gradually swapping in better centers into the final $k$ centers. Another open-source implementation of the DP $k$-means is [27], which is based on locality sensitive hashing. Its adapted version is used in this paper as a building block.

8 CONCLUSION

In this paper, we propose novel differentially private solutions for the vertical federated clustering problem. We demonstrate that our solution can outperform other baselines while providing desired privacy protection on the local data. Some future directions include extending and customizing the DP intersection cardinality estimation sketch to other VFL problems, and providing solutions that can support more data parties and larger $k$ at once.

ACKNOWLEDGMENTS

This work is supported in part by the United States NSF under Grant No. 2220433, No. 2213700, No. 2217071.
F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blon-del, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

A. Rosenberg and J. Hirschberg. V-measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL)*, pages 410–420, 2007.

P. J. Rousseuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65, 1987.

R. Shokri, M. Stronati, C. Song, and V. Shmatikov. Membership inference attacks against machine learning models. In *2017 IEEE symposium on security and privacy (SP)*, pages 3–18. IEEE, 2017.

D. J. Slate. Letter recognition data set. https://archive.ics.uci.edu/ml/datasets/letter-recognition.

A. Smith, S. Song, and A. Thakurta. The flajolet-martin sketch itself preserves differential privacy: Private counting with minimal space. *Advances in Neural Information Processing Systems 33 pre-proceedings* (NeurIPS 2020), 2020.

R. Stanojevic, M. Naheel, and T. Yu. Distributed cardinality estimation of set operations with differential privacy. In *2017 IEEE Symposium on Privacy-Aware Computing (PAC),* pages 37–48. IEEE, 2017.

U. Stemmer and H. Kaplan. Differentially private k-means with constant multiplicative error. In *NeurIPS*, 2018.

D. Su, J. Cao, N. Li, E. Bertino, and H. Jin. Differentially private k-means clustering. In *Proceedings of the sixth ACM conference on data and application security and privacy*, pages 26–37, 2016.

J. Vaidya and C. Clifton. Privacy-preserving k-means clustering over vertically partitioned data. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 206–215, 2003.

J. Vaidya and C. Clifton. Privacy-preserving decision trees over vertically partitioned data. In *IFIP Annual Conference on Data and Applications Security and Privacy*, pages 139–152. Springer, 2005.

C. Wang, J. Liang, M. Huang, B. Bai, K. Bai, and H. Li. Hybrid differentially private federated learning on vertically partitioned data. *arXiv preprint arXiv:2009.02763*, 2020.

T. Wang, J. Blocki, N. Li, and S. Jha. Locally differentially private protocols for frequency estimation. In *26th USENIX Security Symposium*, USENIX Security 2017, Vancouver, BC, Canada, August 16–18, 2017, pages 729–745, 2017.

T. Wang, B. Ding, J. Zhou, C. Hong, Z. Huang, N. Li, and S. Jha. Answering multi-dimensional analytical queries under local differential privacy. In *Proceedings of the 2019 International Conference on Management of Data*, pages 159–176, 2019.

Y. Wang, Y.-X. Wang, and A. Singh. Differentially private subspace clustering. *Advances in Neural Information Processing Systems*, 28, 2015.

WeBank. Webank use case. https://www.fedai.org/cases/a-case-of-traffic-violations-insurance-using-federated-learning/, 2022.

K. Wei, J. Li, M. Ding, C. Ma, H. H. Yang, F. Farokhi, S. Jin, T. Q. Quek, and H. V. Poor. Federated learning with differential privacy: Algorithms and performance analysis. *IEEE Transactions on Information Forensics and Security*, 15:3454–3469, 2020.

N. Wu, F. Farokhi, D. Smith, and M. A. Kaafar. The value of collaboration in convex machine learning with differential privacy. In *2020 IEEE Symposium on Security and Privacy (SP)*, pages 304–317, New York, NY, USA, 2020. IEEE.

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*, 13(11):2096–2103, 2020.

C. Xie, P.-Y. Chen, C. Zhang, and B. Li. Improving privacy-preserving vertical federated learning by efficient communication with admm. *arXiv preprint arXiv:2207.10226*, 2022.

H. Yunhong, F. Liang, and H. Guoping. Privacy-preserving svm classification on vertically partitioned data without secure multi-party computation. In *2009 fifth international conference on natural computation*, volume 1, pages 545–546. IEEE, 2009.

Y. Zhang, R. Jia, H. Pei, W. Wang, B. Li, and D. Song. The secret revealer: Generative model-inversion attacks against deep neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 253–261, 2020.