Differentially Private Vertical Federated Clustering

Zitao Li
Purdue University
li2490@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,

Proceedings of the VLDB Endowment, Vol. 16, No. 6 ISSN 2150-8097.
doi:10.14778/3583140.3583146

each data party also has all dimensions of some data points, and can thus compute these distances. In the VFL setting, however, each data party has access to only a subset of features. As we assume there is only an untrusted server for information aggregation, each data party wants to protect its private user data, and aligning each user's record across different parties under the DP privacy constraint is hard. Thus, the challenge is to *design a differentially private algorithm in which data parties share messages to convey the necessary information for deriving the final global centers.* There are two expectations regarding the shared messages. 1) The messages satisfy DP and convey local information precisely, even with a small privacy budget. Note that a user's information is spread among different parties in VFL and the privacy budget needs to be split among all data parties. 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. We let each data party encode its memberships of the local clusters into Flajolet-Martin (FM) sketches and take advantage of the parallel composition property of DP to reduce the amount of noise (Algorithm 3). Because FM sketches support only union operations while we need intersection operations, we design an algorithm (Algorithm 4) with inclusion-exclusion rules for the server to estimate the intersection cardinalities of memberships. We also prove a theoretical utility guarantee for the final global k centers derived by the server with limited computation and communication overhead.
- 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 intersection cardinalities of memberships from all parties based on pair-wise intersection cardinalities and reduces estimation errors significantly. In addition, we propose a heuristic method to choose the local 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 $\mathcal{A}$ is (ϵ, δ)-differentially private if for any pair of datasets $\mathbf{X}$, $\mathbf{X}'$ that differ in one record and for all possible subset O of possible outputs of algorithm $\mathcal{A}$, $\Pr[\mathcal{A}(\mathbf{X}) \in O] \leq e^{\epsilon} \Pr[\mathcal{A}(\mathbf{X}') \in O] + \delta$.*

Three properties of DP are frequently used to build complicated algorithms. Assume that there are two subroutines $\mathcal{A}_1(\cdot)$ and $\mathcal{A}_2(\cdot)$ that can provides (ϵ_1, δ_1), (ϵ_2, δ_2)-DP protection. *Sequential composition* states that $\mathcal{A}_1(\mathbf{X}, \mathcal{A}_2(\mathbf{X}))$ satisfies $(\epsilon_1 + \epsilon_2, \delta_1 + \delta_2)$-DP. On the other hand, *parallel composition* states that combining two subroutines each only accessing a non-overlapping sub-dataset $\mathbf{X}_1$ or $\mathbf{X}_2$ satisfies $(\max\{\epsilon_1, \epsilon_2\}, \max\{\delta_1, \delta_2\})$-DP. A third property, *post-processing* property states that, any data independent operation on an (ϵ, δ)-DP algorithm's result still satisfies (ϵ, δ)-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, $\mathrm{GS}_f = \max_{\mathbf{X} \simeq \mathbf{X}'} ||f(\mathbf{X}) - f(\mathbf{X}')||_1$. The Laplace mechanism mechanism $\mathcal{A}$ is formalized as $\mathcal{A}_f(\mathbf{X}) = f(\mathbf{X}) + \mathrm{Lap}\left(\frac{\mathrm{GS}_f}{\epsilon}\right)$, where $\mathrm{Lap}(b)$ denotes a random variable sampled from the zero-mean Laplace distribution with scale b. When f outputs a vector, $\mathcal{A}$ adds independent samples of $\mathrm{Lap}\left(\frac{\mathrm{GS}_f}{\epsilon}\right)$ 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 $\mathcal{M} : \mathcal{X} \to \mathcal{Y}$ is said to satisfy (ν, τ)-RDP if the following holds for any two neighboring datasets $\mathbf{X}, \mathbf{X}'$*

$$\frac{1}{\nu - 1} \log \mathbb{E}_{o \sim \mathcal{A}(\mathbf{X})} \left[\left(\frac{\Pr[\mathcal{A}(\mathbf{X}) = o]}{\Pr[\mathcal{A}(\mathbf{X}') = o]} \right)^{\nu} \right] \leq \tau.$$

Fact 2.1 (RDP Sequential Composition [53]). *If $\mathcal{A}_1$ and $\mathcal{A}_2$ are (ν, τ_1)-RDP and (ν, τ_2)-RDP respectively then the mechanism combining the two $g(\mathcal{A}_1(\mathbf{X}), \mathcal{A}_2(\mathbf{X}))$ is $(\nu, \tau_1 + \tau_2)$-RDP.*

Fact 2.2 (RDP to (ϵ, δ)-DP [53]). *If a mechanism is (ν, τ)-RDP, then it also satisfies $(\tau + \frac{\log 1/\delta}{\nu - 1}, \delta)$-DP.*

With the sequential composition of RDP and the conversion to (ϵ, δ)-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 $\mathbf{X} \in \mathbb{R}^{n\times m}$, the goal of the problem is to output a set of k centers $\mathbf{C}$ that can minimize the distance of data points to the nearest centers. The cost (or loss function) is formalized as $\texttt{cost}_{\mathbf{X}}(\mathbf{C}) \coloneqq \sum_{x\in\mathbf{X}}(\min_{c\in\mathbf{C}} \|x-c\|_2^2)$.

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 $\mathbf{X}$. The cost becomes $\texttt{cost}_{\mathbf{X}}(\mathbf{C}) \coloneqq \sum_{x\in\mathbf{X}} w(x)\cdot(\min_{c\in\mathbf{C}} \|x-c\|^2)$.

Theoretically, there is always a set of optimal k centers and the cost is denoted as $\texttt{OPT}_{\mathbf{X}}^k = \min_{|\mathbf{C}|=k} \texttt{cost}_{\mathbf{X}}(\mathbf{C})$. 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 $O(nmk)$. A notation, (β,λ)-approximate, is used to describe the utility guarantee of an approximate algorithm, such that $\texttt{cost}_{\mathbf{X}}(\mathbf{C}) \le \beta\cdot\texttt{OPT}_{\mathbf{X}}^k + \lambda$ with any $\mathbf{X}$ and k, where β is called *approximate 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 $\mathcal{M}$. In FM sketch, all the elements in $\mathcal{M}$ are hashed with $H_\zeta(\cdot)$, an ideal geometric-value hash function. The estimate of the cardinality is $(1+\gamma)^\alpha$, where $\alpha = \max\{H_\zeta(x)|x\in\mathcal{M}\}$ and γ is the parameter of hash function. Typically, multiple (e.g., 1000) hash functions (H with different hash keys ζ) are used, and we take the harmonic/geometric average of those maximums as the final α. 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 leaning (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 $\mathbf{X}$, where each row corresponds to a user, and each column corresponds to a feature. The setting of VFL is that $\mathbf{X}$ is vertically split into $\mathbf{X} = [\mathbf{X}^{(1)}|\ldots,|\mathbf{X}^{(S)}]$, so that each data party $\ell \in [S]$ has a local dataset $\mathbf{X}^{(\ell)}$ with $m^{(\ell)}$ features. We assume each user is labeled with a unique $\texttt{id}$ (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 $\mathbf{C}$ in the full domain (with all attributes) that can approximately minimize the k-means loss $\texttt{cost}_{\mathbf{X}}(\mathbf{C})$. 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 ℓ performs clustering to find k' local cluster centers $\mathbf{C}^{(\ell)} = \left\{c_1^{(\ell)},\ldots,c_{k'}^{(\ell)}\right\}$; and then sends the k' centers together with membership information $\mathbf{I}^{(\ell)} = \left\{\mathcal{M}_1^{(\ell)},\ldots,\mathcal{M}_{k'}^{(\ell)}\right\}$ to the central server, where $\mathcal{M}_a^{(\ell)} = \left\{\texttt{id} \mid a = \arg\min \left\|x_{\texttt{id}}^{(\ell)} - c_a^{(\ell)}\right\|_2^2\right\}$. 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., $\mathbf{G} = \{(c_{a_1}^{(1)},\ldots,c_{a_S}^{(S)}) \mid \forall(a_1,\ldots,a_S)\in[k']^S\}$; and assigns the cardinality of the intersection of the corresponding clusters as weights to them such that $w(\mathbf{G}_{(a_1,\ldots,a_S)}) = \left|\mathcal{M}_{a_1}^{(1)}\cap\ldots\cap\mathcal{M}_{a_S}^{(S)}\right|$.

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 $c_{a_\ell}^{(\ell)}$ is the exact average of the user data in $\mathcal{M}_{a_\ell}^{(\ell)}$;

moreover, data points in the intersection $\mathcal{M}_{a_1}^{(1)} \cap \ldots \cap \mathcal{M}_{a_S}^{(S)}$ are expected to distribute around the pseudo point $(c_{a_1}^{(1)}, \ldots, c_{a_S}^{(S)})$. If the intersection has a small cardinality or even is an empty set, we can know that the pseudo point can be ignored. The weighted grid is similar to a useful data synopsis in the k-means cost analysis, called *coreset* [31], which approximates the original dataset information. As long as the weighted grid nodes are representative enough for a subset of points, the central server can find final centers without accessing the distributed datasets.

3.3 Challenge in the Privacy-preserving Setting

The approach described in Section 3.2 does not consider the privacy leakage problem, as the local cluster centers and membership information sent to the server contain sensitive information. To protect users' private information, all the information sent to the central server, including (1) local cluster centers $\mathbf{C}^{(\ell)}$ and (2) local membership information, should be differentially private.

For the clustering centers, there already exists comprehensive research of the k-means algorithm in the central DP setting [8, 22, 38, 54–56, 65, 72]. Thus we can choose a method that works well.

Sending local membership information while satisfying DP is, however, very challenging. To the best of our knowledge, there is no effective DP algorithm for sharing membership information, especially in the scenario with more than two parties. Fortunately, the reason that we need to share the membership information is to estimate the weights of each pseudo data point. Thus, we do not need to share precise membership information, and just need a private way to *estimate the cardinality of the intersection among multiple parties*. The main technical contribution of this paper is a solution to this problem, which we will described in Section 4. Our proposed approach leverages DP FM sketch and is extended to support the intersection operation among parties.

3.4 The Overall Framework

Figure 1 is a visualized workflow of Algorithm 1 with two data parties (note that our algorithm/analysis work with the general 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 ϵ_0[1]. 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.

[1]We set $\epsilon_0 = 0.02\epsilon$ unless we specify in the following text.

Algorithm 1 Private Vertical Federated Clustering

Input: Local datasets $\{\mathbf{X}^{(\ell)} \in \mathbb{R}^{n \times m^{(\ell)}} \mid \ell \in [S]\}$, total privacy budget (ϵ, δ) is divided as $\epsilon_0 = (1-b)\epsilon$, $\epsilon_1 = \frac{b\epsilon}{2S}$, $\epsilon_2 = \frac{b\epsilon}{2S}$, $\delta_2 = \frac{\delta}{S}$ 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 $\ell \in [S]$:
   (a): $\{c_1^{(\ell)}, \ldots, c_{k'}^{(\ell)}\} \leftarrow \mathsf{LocCluster}(\mathbf{X}^{(\ell)}, \epsilon_1, k')$
2: Each data party $\ell \in [S]$:
   (a): $\mathbf{I}^{(\ell)} \leftarrow \mathsf{MemEnc}(\mathbf{X}^{(\ell)}, \{c_1^{(\ell)}, \ldots, c_{k'}^{(\ell)}\}, \epsilon_2, \delta_2, \mathsf{aux})$
   (b): sends $\mathbf{C}^{(\ell)} = \{c_1^{(\ell)}, \ldots, c_{k'}^{(\ell)}\}$ and $\mathbf{I}^{(\ell)}$ to server
3: Central server:
   (a): uses ϵ_0 to estimate the total number of user $\hat{n}$
   (b): receives $\{(\mathbf{C}^{(\ell)}, \mathbf{I}^{(\ell)}) | \ell \in [S]\}$ from all parties
   (c): computes grid by Cartesian product $\mathbf{G} \leftarrow \mathbf{C}^{(1)} \times \ldots \times \mathbf{C}^{(S)}$
   (d): computes $w(\mathbf{G}) \leftarrow \mathsf{WeightEst}(\hat{n}, \epsilon_2, \delta_2, \{\mathbf{I}^{(\ell)} | \ell \in [S]\})$
4: Central server:
   (a): computes and outputs $\{c_1, \ldots, c_k\} \leftarrow k\text{-means}(G, w(G), k)$

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

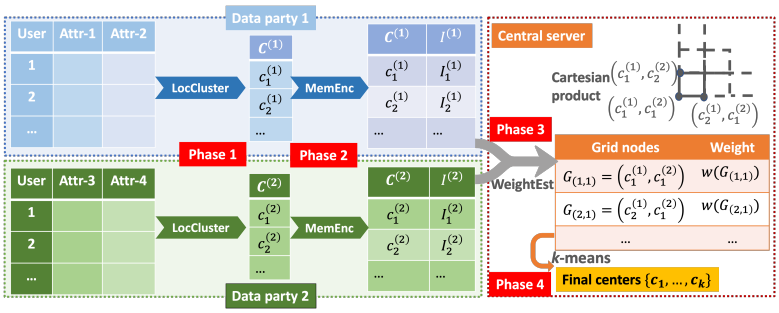


Figure 1: General framework for VFL k-means clustering.

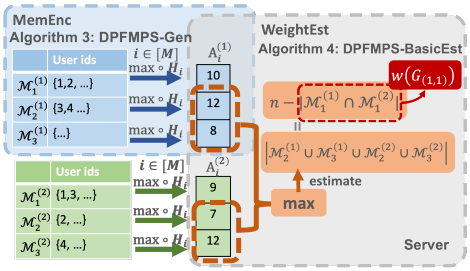


Figure 2: Example of Algorithm 3 and 4.

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 to $[-1, 1]$. However, our technique can be easily extended when different attributes are normalized to different target ranges, so long as these target ranges are public information, e.g., general domain knowledge; otherwise, normalization ranges can be inferred using other DP algorithms with reserved private budgets.

4 PRIVATE MEMBERSHIP ENCODING AND WEIGHT ESTIMATE

To avoid the privacy leakage when sharing the membership information $\mathbf{I}^{(\ell)}$, we introduce our private instantiations of MemEnc and WeightEst together in this section because how the central server can estimate the weights with WeightEst depends on how data parties encode the membership information with MemEnc. With our instantiations, the data parties generate differentially private membership information $\mathbf{I}^{(\ell)}$ and share them with the central server, and the central server estimates the cardinalities of the intersections $\left|\mathcal{M}_{a_1}^{(1)} \cap \ldots \cap \mathcal{M}_{a_S}^{(S)}\right|$ for all $(a_1, \ldots, a_S) \in [k']^S$ as weights.

4.1 Baselines

Baseline 1: Estimate weights assuming independence among attributes. In this approach, we assume that the distributions of attributes from one party are independent of those from all other parties. Under this assumption, we can compute the intersection cardinality using $\left|\mathcal{M}_{a_1}^{(1)} \cap \ldots \cap \mathcal{M}_{a_S}^{(S)}\right| \approx \hat{n} \prod_{\ell \in [S]} \frac{|\mathcal{M}_{a_\ell}^{(\ell)}|}{\hat{n}}$.

Following this idea, the private MemEnc only needs to generate a histogram of $\left[\left|\mathcal{M}_1^{(\ell)}\right|, \ldots, \left|\mathcal{M}_{k'}^{(\ell)}\right|\right]$ with Laplace mechanism and privacy budget ϵ_2. Denote the randomized histogram vector as $\tilde{\mathbf{f}}^{(\ell)}$. The central server's sub-procedure WeightEst is $w(\mathbf{G}_{(a_1,\ldots,a_S)}) = \hat{n} \prod_{\ell \in [S]} \frac{\tilde{\mathbf{f}}_{a_\ell}^{(\ell)}}{\hat{n}}$. 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 ϵ_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 $\mathcal{M}_{a_1}^{(1)} \cap \ldots \cap \mathcal{M}_{a_S}^{(S)}$. 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 $O\left(n/\epsilon_2^{2S}\right)$ for each intersection. The noise can easily overwhelm the true counts when ϵ_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, 35, 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 $H : \mathcal{X} \times \mathbb{Z} \to \mathbb{N}_+$ with parameter $\frac{\gamma}{1+\gamma}$. That is, given any finite set of distinct inputs $x_1, \ldots, x_\ell \in \mathcal{X}$, with a hash key $\zeta \sim \text{Uniform}(\mathbb{Z})$, the hashed values $H_\zeta(x_1), \ldots, H_\zeta(x_\ell)$ follow i.i.d. $\text{Geometric}\left(\frac{\gamma}{1+\gamma}\right)$ distribution.

Algorithm 2 DP FM Sketch Generation DPFM [63]

Input: a (multi)set $\mathcal{M}$, privacy parameter ϵ', and Geometric distribution parameter γ, an ideal random hash function $H_\zeta(x) \sim \texttt{Geometric}(\frac{\gamma}{1+\gamma})$ when $\zeta \sim \texttt{Uniform}(\mathbb{Z})$

Output: Sketch α for cardinality of $\mathcal{M}$

1: $n_p = \lceil \frac{1}{e^{\epsilon'}-1} \rceil$, $\alpha_{\min} = \lceil \log_{1+\gamma} \frac{1}{1-e^{-\epsilon'}} \rceil$
2: $\alpha_p = \max\{Y_1, \ldots, Y_{n_p}\}$ where $Y_i \sim \texttt{Geometric}(\frac{\gamma}{1+\gamma})$
3: $\alpha_{real} = \max\{H_\zeta(\texttt{id}) | \texttt{id} \in \mathcal{M}\}$
4: Return $\alpha = \max\{\alpha_p, \alpha_{real}, \alpha_{\min}\}$

Algorithm 2 generates a DP FM sketch. The intuition of the non-private FM sketch is that when elements in $\mathcal{M}$ are encoded as a set of geometric random variables and $\alpha_{real} = \max\{H_\zeta(\texttt{id}) | \texttt{id} \in \mathcal{M}\}$, we can expect $(1+\gamma)^{\alpha_{real}} \in \left[\frac{|\mathcal{M}|}{1+\gamma}, (1+\gamma)\cdot|\mathcal{M}|\right]$ 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 $\alpha_{\min}$. The phantom elements are used to ensure that the cardinality estimated by the final output is at least n_p; the $\alpha_{\min}$, 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 ϵ'-DP. Besides, repeating it M times with different hash keys and $\epsilon' = \frac{\epsilon}{4\sqrt{M\log(1/\delta)}}$ satisfies (ϵ, δ)-DP as long as $\epsilon \le 2\log(1/\delta)$.*

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 $\left\{\mathcal{M}_1^{(\ell)}, \ldots, \mathcal{M}_{k'}^{(\ell)}\right\}$ 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, $\text{aux} = \{\gamma, M, \boldsymbol{\zeta} = \{\zeta_1, \ldots, \zeta_M\}\}$, where γ is the Geometric distribution parameter, M is the number of sets of sketches and each $\zeta_i \sim \texttt{Uniform}(\mathbb{Z})$ 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 (ϵ_2, δ_2)-differential privacy.*

The above theorem is based on the following lemma about the privacy guarantee for each set of sketch, namely $\boldsymbol{A}_i$ in Algorithm 3.

Lemma 3. *Given an ideal geometric-value hash function H, each set of the partition sketch (i.e. a row $\boldsymbol{A}_i$) is ϵ'-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 (ϵ, δ)-DP following the same proof as in [63].

Algorithm 3 DPFMPS-Gen

Input: A set of k' centers $\mathbf{C}^{(\ell)}$, dataset $\mathbf{X}^{(\ell)}$, privacy parameter ϵ_2 and δ_2, Geometric distribution parameter γ, number of sketches M and the corresponding hash keys $\boldsymbol{\zeta} = \{\zeta_1, \ldots, \zeta_M\}$

Output: M sets of sketches $\boldsymbol{A}$

1: $\epsilon' = \frac{\epsilon_2}{4\sqrt{M\log(1/\delta_2)}}$
2: Generate $\left\{\mathcal{M}_1^{(\ell)}, \ldots, \mathcal{M}_{k'}^{(\ell)}\right\}$ where $\texttt{id} \in \mathcal{M}_j^{(\ell)}$ based on $\mathbf{C}^{(\ell)}$
3: $\boldsymbol{A} \leftarrow \mathbf{0}^{M\times k'}$
4: **for** $i \in [M], a \in [k']$ **do**
5: $\quad \boldsymbol{A}_{i,a} = \texttt{DPFM}(\mathcal{M}_a^{(\ell)}, \epsilon', \gamma, \zeta_i)$
6: **end for**
7: Return $\boldsymbol{A}$

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

$$\bigcap_{\ell=1}^{S} \mathcal{M}_{a_\ell}^{(\ell)} = \overline{\bigcup_{\ell=1}^{S} \overline{\mathcal{M}_{a_\ell}^{(\ell)}}} = \overline{\bigcup_{\ell=1}^{S} \left(\bigcup_{a \neq a_\ell} \mathcal{M}_a^{(\ell)} \right)}. \tag{1}$$

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, $\mathbf{U}$ is initialized to store the FM sketches for the cardinalities of the complementary set of intersections; $\mathbf{w}'$ is an intermediate vector representing the cardinalities of union of complements, i.e. $\left|\bigcup_{\ell=1}^{s} \overline{\mathcal{M}_{a_\ell}^{(\ell)}}\right|$.

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 ϵ_2 and δ_2, the FM partition sketches $\{\boldsymbol{A}^{(1)}, \ldots, \boldsymbol{A}^{(s)}\}$
Output: Estimate of the weights $w(\mathbf{G})$

1: $\mathbf{U} \leftarrow \mathbf{0}^{M\times(k')^s}$, $\mathbf{w}' \leftarrow \mathbf{0}^{(k')^s}$
2: $\epsilon' = \frac{\epsilon_2}{4\sqrt{M\log(1/\delta_2)}}$, $n_p = \lceil \frac{1}{e^{\epsilon'}-1} \rceil$
3: **for** $i \in [M]$, $(a_1, \ldots, a_s) \in [k']^s$ **do**
4: $\quad \mathbf{U}_{i,(a_1,\ldots,a_s)} = \max\{\boldsymbol{A}^{(\ell)}_{i,a} \mid \ell \in [s], a \neq a_\ell\}$
5: **end for**
6: **for** $(a_1, \ldots, a_s) \in [k']^s$ **do**
7: $\quad \boldsymbol{\alpha}_{(a_1,\ldots,a_s)} = \texttt{HarmonicMean}\left(U_{:,(a_1,\ldots,a_s)}\right)$
8: $\quad \mathbf{w}'_{(a_1,\ldots,a_s)} = (1+\gamma)^{\boldsymbol{\alpha}_{(a_1,\ldots,a_s)}} - s(k'-1)n_p$
9: $\quad w(\mathbf{G}_{(a_1,\ldots,a_s)}) = \hat{n} - \mathbf{w}'_{(a_1,\ldots,a_s)}$
10: **end for**
11: Ensure $\sum_{(a_1,\ldots,a_s)} w(\mathbf{G}_{(a_1,\ldots,a_s)}) = \hat{n}$ and $w(\mathbf{G})_{(a_1,\ldots,a_s)} \geq 0$
12: Return $w(\mathbf{G})$

$\overline{\mathcal{M}^{(\ell)}_{a_\ell}}$. The corresponding FM sketch can be obtained by taking the max of all other elements in the same sketch set, $\max\left\{A^{(\ell)}_{i,a} | a \neq a_\ell\right\}$, 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. $\bigcup_{\ell=1}^{s} \overline{\mathcal{M}^{(\ell)}_{a_\ell}}$. This union's FM sketches can be obtain by $\max\left\{\max\left\{A^{(\ell)}_{i,a} | a \neq a_\ell\right\} | \ell \in [s]\right\}$. Merging the two max operations give us the operation in Line 4.

We use the Harmonic mean over the M FM sketches to estimate the cardinality of $\bigcup_{\ell=1}^{s} \mathcal{M}^{(\ell)}_{a_\ell}$ in Line 7, because it is shown [63] that the harmonic mean estimate is more stable and accurate. As this sketch is obtained after $s(k'-1)$ union operations, totally $s(k'-1)n_p$ phantom elements are taken into account. Therefore, we need to subtract $s(k'-1)n_p$ elements from the final estimation in Line 8. Finally, the intersection cardinality can be calculated by subtracting the cardinalities of $\bigcup_{\ell=1}^{s} \overline{\mathcal{M}^{(\ell)}_{a_\ell}}$ from the total number of users $\hat{n}$. Before finally returning the weights, we need to make sure the output is valid by enforcing non-negativity and total-sum equal to $\hat{n}$ on the weights (Line 11). An example of running Algorithm 3 and 4 is shown in Figure 2 with two data parties and $k' = 3$.

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 (ϵ, δ)-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 $\alpha_{\min}$ 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 $\hat{\alpha} = \max\{\alpha_{real}, \alpha_p\}$ and α be the return of Algorithm 2. If H is an ideal geometric-value hash function and ϵ' is fixed, we have $\mathbb{E}[\alpha]/\mathbb{E}[\hat{\alpha}] \to 1$ and $\mathrm{Var}[\alpha]/\mathrm{Var}[\hat{\alpha}] \to 1$ when $|\mathcal{M}|$ is sufficiently large.*

With Lemma 5, we can claim that the lower bound $\alpha_{\min}$ does not affect the mean and variance of the sketch $\hat{\alpha}$ too much. So we can treat the α 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 ρ 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 $\mathbf{U}$, and each set has n_p phantom elements. So the cardinality estimated by $\mathbf{U}_{:,(a_1,\ldots,a_S)}$ becomes $N_{(a_1,\ldots,a_S)} = \left|\bigcup_{\ell\in[S]} \cup_{j\neq a_\ell} \mathcal{M}^{(\ell)}_j\right| + s(k'-1)n_p = \hat{n} - w^*(\mathbf{G}_{a_1,\ldots,a_S}) + s(k'-1)n_p$, where $w^*(\mathbf{G}_{a_1,\ldots,a_S})$ represent the true intersection cardinality $|\mathcal{M}^{(1)}_{a_1} \cap \ldots \cap \mathcal{M}^{(S)}_{a_S}|$. Based on the property of the non-private FM sketch and the value of M and n_p, we can state the following lemma.

Theorem 6. *Given a constant ρ 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 n_p and ϵ' set as in Algorithm 4, each intersection cardinality estimate generated has standard deviation*

$$\sigma_{(a_1,\ldots,a_s)} = \frac{\rho(n - w^*(\mathbf{G}_{(a_1,\ldots,a_s)}))}{\sqrt{M}} + \frac{4\rho s(k'-1)\sqrt{\log(1/\delta)}}{\epsilon_2}$$

The result is directly derived after plugging in the value of $N_{(a_1,\ldots,a_S)}$ and n_p, 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 (β, λ)-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 (non-private) $(\beta_o, 0)$-approximate k-means algorithm. Algorithm 1 with DPFMPS-Gen and DPFMPS-BasicEst is a (β, λ)-approximation algorithm with probability $1-\omega$, where*

$$\beta = 2\beta_{priv} + 4\beta_0 + 4\beta_0\beta_{priv},$$

$$\lambda = 2(\beta_0+1)S\lambda_{priv} + O\left(\frac{\beta_0 m^2 k^{1.5S}}{\sqrt{\omega}}\left(\frac{n}{\sqrt{M}} + \frac{S(k-1)\sqrt{\log(1/\delta)}}{\epsilon_2}\right)\right)$$

The multiplicative error β is composed of β_{priv} from the LocCluster and β_0 from the final non-private clustering on central server. By the latest theoretical result [26], β_{priv} and β_0 are close to 1 when k is a constant. So our approximate ratio ($\beta \approx 10$) is slightly larger

than the bound in the non-private algorithm ($\beta \approx 9$) [18], because of the randomness when estimating the cardinality satisfying DP. Besides, DP k-means algorithms are unavoidable to have an additive error besides the multiplicative error [26]. The first term of our additive error λ can be understood as the cumulative error of S local DP k-means algorithm results[2], and the second term comes from the cumulative cardinality estimation error of k^S nodes. The second term can dominate the error when the number of centers k or the number of data parties S is not small. If both n and M are large enough, then the averaged additive error (divided by n) will still vanish. Besides, we also empirically show that the losses can be small and even close to the central private k-means losses on some datasets when the privacy budget is large enough.

Communication and computation cost. There is only one round of communication between the data parties and the central server. Compared with the non-private baseline [18], the additional computation cost for DPFMPS-Gen on each data party is $O(nM)$ hashing operations for M sets of sketches. However, this process can be easily accelerated by parallel computation because each DPFM can run independently. The communication cost is $O\left((m^{(\ell)} + M)k'\right)$. Our algorithm's communication cost is independent of n and it can even smaller than the non-private solution $O\left(m^{(\ell)}k' + n\right)$ [18] when $n > Mk'$. Compared with the non-private algorithm requiring $O(nS)$ operations for the intersection cardinality, our private algorithm needs $O(MSk'^S)$ for the server to estimate the weights.

5 IMPROVING UTILITY OF THE ALGORITHM

We introduce two heuristic methods in this section to improve the empirical performance of our methods when S is large.

5.1 Improving Post-processing Estimation Algorithm for More than Two Data Parties

As the number of data parties increases, the accuracy of the weight estimation will decrease dramatically. From the error analysis of the cardinality estimation (Theorem 6), one can see that the second term in standard deviation $\sigma_{(a_1,\ldots,a_S)}$ increases proportionally with the number of data parties S. Furthermore, the total possible intersection combinations grow exponentially as $(k')^S$, which means fewer expected number of data points fall in the intersection, i.e., the expected $w^*(\mathbf{G}_{(a_1,\ldots,a_S)})$. The combination of the two factors means that the relative error of the intersection estimation explodes as the number of parties grows.

Two observations give us hope to lessen the negative effect. The first observation is that estimation of two-party intersection cardinalities is relatively accurate. The second observation is based on the distributive property of set intersection:

$$\left|\mathcal{M}^{(\ell_1)}_{a'_{\ell_1}} \bigcap \mathcal{M}^{(\ell_2)}_{a'_{\ell_2}}\right| = \sum_{\substack{(a_1,\ldots,a_S)\in[k']^S \\ a_{\ell_1}=a'_{\ell_1}, a_{\ell_2}=a'_{\ell_2}}} \left|\bigcap \mathcal{M}^{(\ell)}_{a_\ell}\right|, \tag{2}$$

which give the connection between the all-party intersection cardinality (i.e., $w(\mathbf{G})$) and the two-party version (i.e., $w(\mathbf{G}^{(\ell_1,\ell_2)})$). Thus, we propose to deal with this challenge by 1) *computing only all pair-wise intersection cardinalities*, and 2) using these two-party intersection cardinalities with Equation (2) as constraints, and iteratively update the $w(\mathbf{G})$ to fulfill all these constraints.

The improved estimation is described in Algorithm 5 DPFMPS-2PEst. The central server first estimates the single-party cardinality of all local clusters with only $\boldsymbol{A}^{(\ell)}$ (Line 3). Namely, $w^{(\ell)}_a$ is an estimate for $|\mathcal{M}^{(\ell)}_a|$. Then the server initializes the full grid weights $w(\mathbf{G})$ with only the single-party cardinality information assuming no correlation between the inter-party attributes (Line 5). Next, the server estimates all two-party weights with DPFMPS-BasicEst as a sub-procedure (Line 7). To be more detailed, the grid weights $w(\mathbf{G}^{(\ell_1,\ell_2)})$ returned by DPFMPS-BasicEst with $\boldsymbol{A}^{(\ell_1)}$ and $\boldsymbol{A}^{(\ell_2)}$ are the estimates for $\left\{\left|\mathcal{M}^{(\ell_1)}_{a_{\ell_1}} \bigcap \mathcal{M}^{(\ell_2)}_{a_{\ell_2}}\right| \mid a_{\ell_1}, a_{\ell_2} \in [k']\right\}$. With all pairs of $w(\mathbf{G}^{(\ell_1,\ell_2)})$, the server iteratively updates the grid weights $w(\mathbf{G})$ to make them consistent with all the two-party intersection cardinalities $w(\mathbf{G}^{(\ell_1,\ell_2)})$. In each iteration, the server first randomly selects a pair of data parties (Line 10), groups the current full grid weights $w(\mathbf{G})$ by the cluster indices of the chosen two parties, and sums the weights in the same group to generate a two-party grid weights $\tilde{w}(\mathbf{G}^{(\ell_1,\ell_2)})$ according to Equation (2):

$$\tilde{w}(\mathbf{G}^{(\ell_1,\ell_2)}_{(a'_{\ell_1},a'_{\ell_2})}) = \sum_{\substack{(a_1,\ldots,a_S)\in[k']^S \\ a_{\ell_1}=a'_{\ell_1}, a_{\ell_2}=a'_{\ell_2}}} w(\mathbf{G}_{(a_1,\ldots,a_S)})$$

We use the differences between $w(\mathbf{G}^{(\ell_1,\ell_2)})$ and $\tilde{w}(\mathbf{G}^{(\ell_1,\ell_2)})$ to update the weight evenly with a step size η_t (Line 15). After sufficient update iterations, the full-party grid weight $w(\mathbf{G})$ is expected to approximately satisfy the constraints (Equation (2)) with all two-party weights $w(\mathbf{G}^{(\ell_1,\ell_2)})$. Because both DPFMPS-2PEst and DPFMPS-BasicEst are post-processing components in the DP definition, the privacy guarantee in Theorem 4 still holds for DPFMPS-2PEst.

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 $w^*(\mathbf{G}_{a_1,\ldots,a_S})$) are expected to be assigned to the grid node. According to Theorem 6, a larger k' can make the error of $w(\mathbf{G}_{a_1,\ldots,a_S})$ 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 λ plays an important role. We propose an empirical rule to set $k' = \max\{k_0, k^{1/S}\}$ for DPFMPS-2PEst to prevent the true cardinalities from being overwhelmed by the noise, where k_0 is the smallest integer that satisfies $2\sigma_{(a_1,\ldots,a_s)} \geq \frac{\hat{n}}{k_0^2}$. Because $w^*(\mathbf{G}_{(a_1,\ldots,a_S)})$ in Theorem 6 is unknown, we approximate it as $\frac{\hat{n}}{k_0^2}$; we set $s = 2$ in the standard deviation because DPFMPS-2PEst only decodes pairs of intersection cardinalities; and 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.

[2] According to the best theoretical results of DP k-means in central setting [26], $\lambda_{priv} = O_{\beta,\eta}\left(\epsilon_1^{-1}(km + k^{O_\eta(1)})\,\mathrm{polylog} n\right)$ with a small positive constant η.

Algorithm 5 DPFMPS-2PEst

Input: Estimated total number of users $\hat{n}$, privacy parameter ϵ_2 and δ_2, the FM partition sketches $\{\boldsymbol{A}^{(1)}, \ldots, \boldsymbol{A}^{(s)}\}$
Output: Estimate of the weights $w(\mathbf{G})$

$\epsilon' = \frac{\epsilon_2}{4\sqrt{M \log(1/\delta_2)}}$, $n_p = \lceil \frac{1}{e^{\epsilon'}-1} \rceil$
for $\ell \in [S], a \in [k']$ **do**
  $w_a^{(\ell)} \leftarrow (1+\gamma)^{\texttt{HarmonicMean}\left(\boldsymbol{A}_{\cdot,a}^{(\ell)}\right)} - n_p$.
end for
$\forall \{a_1, \ldots, a_S\} \in [k']^S, w(\mathbf{G}_{(a_1,\ldots,a_S)}) \leftarrow \hat{n} \times \prod_{\ell \in [S]} \frac{w_{a_\ell}^{(\ell)}}{\hat{n}}$
for $(\ell_1, \ell_2) \in [S]$ and $\ell_1 \neq \ell_2$ **do**
  $w(\mathbf{G}^{(\ell_1,\ell_2)}) \leftarrow$ DPFMPS-BasicEst$(\hat{n}, \epsilon_2, \delta_2, \{\boldsymbol{A}^{(\ell_1)}, \boldsymbol{A}^{(\ell_2)}\})$
end for
for $t \in [T]$ **do**
  Randomly select a pair $(\ell_1, \ell_2) \in [S] \times [S]$
  $\tilde{w}(\mathbf{G}^{(\ell_1,\ell_2)}) \leftarrow \texttt{Proj}(w(\mathbf{G}), \ell_1, \ell_2)$
  $\Delta^{(\ell_1,\ell_2)} \leftarrow \tilde{w}(\mathbf{G}^{(\ell_1,\ell_2)}) - w(\mathbf{G}^{(\ell_1,\ell_2)})$
  for $(a'_{\ell_1}, a'_{\ell_2}) \in [k']^2$ **do**
    $\forall (a_1, \ldots, a_S) \in [k']^S, a_{\ell_1} = a'_{\ell_1}, a_{\ell_2} = a'_{\ell_2}$
    $w(\mathbf{G}_{(a_1,\ldots,a_S)}) \leftarrow w(\mathbf{G}_{(a_1,\ldots,a_S)}) - \frac{\eta_t}{(k')^{S-2}} \Delta^{(\ell_1,\ell_2)}_{(a'_{\ell_1}, a'_{\ell_2})}$
  end for
end for
Return $w(\mathbf{G})$

6 EXPERIMENTS

Datasets. We use four different datasets in our experiments, preprocess the data so that all attributes are normalize 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., $\frac{1}{n}\sum_{x \in \mathbf{X}}(\min_{c \in \mathbf{C}} \|x - c\|^2)$, 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.

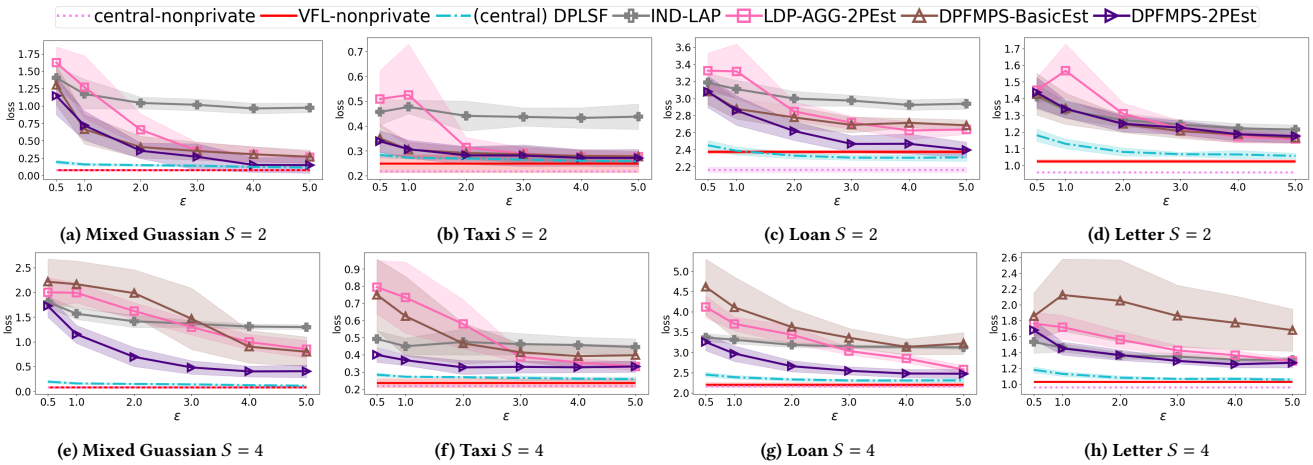
Compared methods. The adapted DPLSF is used as the LocCluster in all private VFL experiments. So we name the end-to-end private VFL clustering method based on their MemEnc and WeightEst instantiations. Our experiments compare the following methods:

- 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.

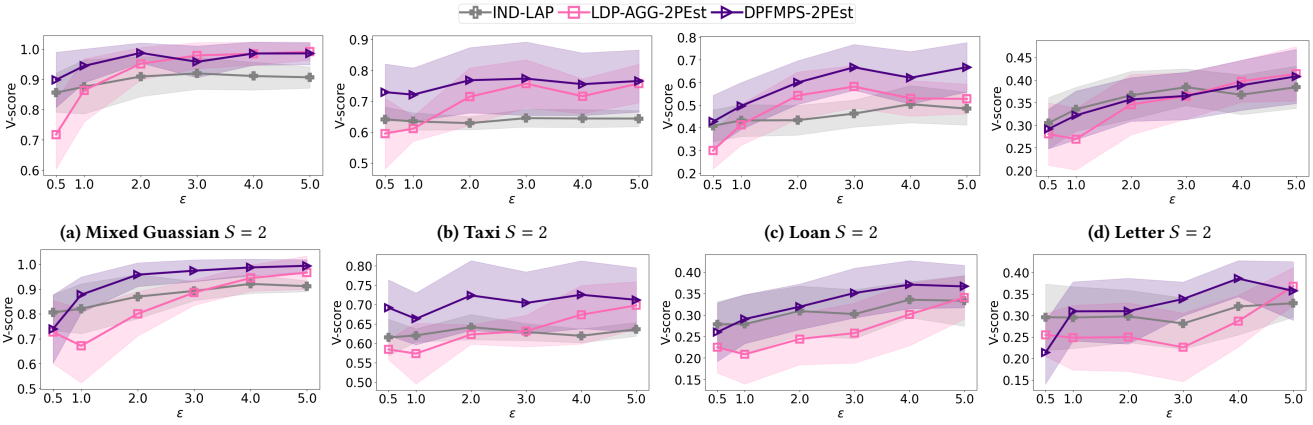
***k*-means loss results.** Figure 3 shows the loss of the final centers produced by different algorithms. As we can see, the losses of non-private vertical federated k-means are higher than the that of the central version in most cases. Notice that we set $k' = k = 5$ for the non-private VFL algorithm, 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



(a) **Mixed Guassian** $S = 2$ (b) **Taxi** $S = 2$ (c) **Loan** $S = 2$ (d) **Letter** $S = 2$

(e) **Mixed Guassian** $S = 4$ (f) **Taxi** $S = 4$ (g) **Loan** $S = 4$ (h) **Letter** $S = 4$

Figure 3: k-means loss with final k centers.



(a) **Mixed Guassian** $S = 2$ (b) **Taxi** $S = 2$ (c) **Loan** $S = 2$ (d) **Letter** $S = 2$

(e) **Mixed Guassian** $S = 4$ (f) **Taxi** $S = 4$ (g) **Loan** $S = 4$ (h) **Letter** $S = 4$

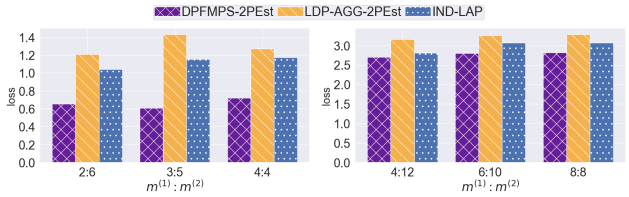
Figure 4: V-scores with final k centers.

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 on the Letter dataset (Figure 3(d) and 3(h)), where IND−LAP approach has slightly better performance when ϵ 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 ϵ 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].

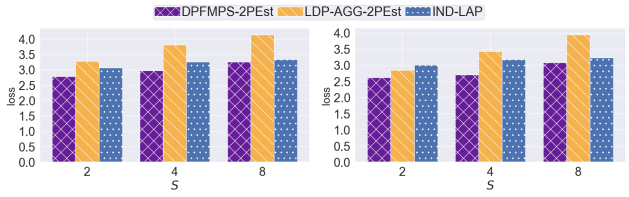
Moreover, comparing the private central with the private VFL algorithms, we can see that the private VFL almost always has a higher cost than the central one. The gap exists because of two different reasons. 1) Only the local centers and the sketches are shared with the central server in VFL, so some 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



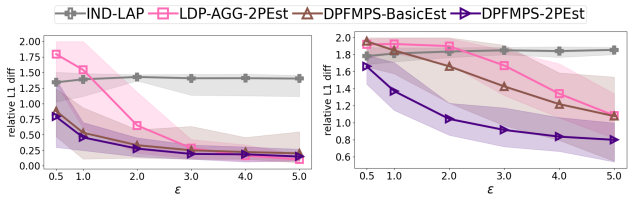
(a) Mixed Guassian $\epsilon = 1$ (b) Loan $\epsilon = 1$

Figure 5: Effect of unevenly split dataset.



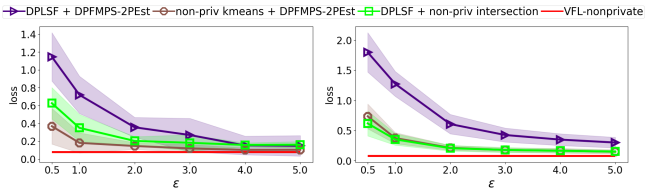
(a) Loan $\epsilon = 1$ (b) Loan $\epsilon = 2$

Figure 6: Effect of different S on Loan dataset ($m = 16$).



(a) Mixed Gaussian $S = 2$ (b) Mixed Gaussian $S = 4$

Figure 7: Errors of estimate the intersection cardinalities.



(a) Mixed Gaussian $S = 2$ (b) Mixed Gaussian $S = 4$

Figure 8: Impact of enforcing privacy on components.

for the dataset. Thus, the closer the score is to 1, the more similar the clustering result is to the real labels (for Mixed Gaussian) or the central non-private k-means clustering (Taxi, Loan and Letter).

Figure 4 presents the result. The results align with the k-means loss results. In most settings, we can observe that DPFMPS-2PEst outperforms the other two VFL private baseline methods in Figure 4. For the Letter dataset, we can see that all three methods have lower scores than the ones of other datasets when $S = 2$. It is because the data in Letter have very independent attributes and bring advantages to the IND-LAP. However, DPFMPS-2PEst still outperforms other methods on other datasets with different privacy budgets, and it also has performance close to best when $S = 2$.

Effect of uneven number of attributes. The previous results show how DPFMPS-2PEst outperforms other baselines method when the attributes are evenly split. We also explore how the methods perform when the each party has different number of attributes. Denote $m^{(i)}$ as the number of attributes in the i-th party's dataset. We split the mixed Gaussian dataset so that $m^{(1)} : m^{(2)}$, is $3 : 5$ or $2 : 6$, and split the Loan dataset to $6 : 10$ and $4 : 12$. The results shown in Figure 5 indicate that our DPFMPS-2PEst work consistently well in the sense that the losses of different split ratios do not change much. Besides, DPFMPS-2PEst losses are smaller than the other two baselines in the figure.

Effect of number of parties S. We also compare how the number of parties S can affects the final clustering results in Figure 6. The loss generally increases as S increases. This is mainly because the WeightEst component introduces a larger error because of the random noise for privacy. However, DPFMPS-2PEst always provides the lowest loss among the three.

6.2 Ablation Study of Components

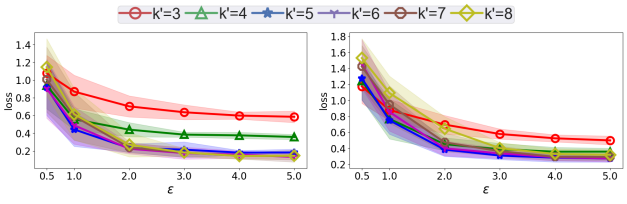
We perform the following ablation studies to demonstrate the impact of enforcing privacy on different components.

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 $\frac{1}{n}\sum_{(a_1,\ldots,a_S)} \left\| w(\mathbf{G}_{(a_1,\ldots,a_S)}) - w^*(\mathbf{G}_{(a_1,\ldots,a_S)}) \right\|_1$.

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 can outperform other methods in most experiment settings. The DPFMPS-BasicEst performs similar as DPFMPS-2PEst in the $S = 2$ setting as expected. DPFMPS-2PEst can significantly improve the accuracy when there are more parties ($S = 4$). The LDP baseline LDP-AGG-2PEst still has relative error larger than the DPFMPS-2PEst, even with the 2-way iterative updating. Moreover, as we can see from the figure, the error of the 1-way approach IND-LAP is dominated by loss of dependency information between the attributes and barely going down as we increase the privacy budget.

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 ϵ_2 on the intersection estimation algorithm. For "DPLSF + non-priv intersection" experiments, each data party spends ϵ_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 λ 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.



(a) Mixed Gaussian, $S = 2$ (b) Mixed Gaussian, $S = 4$

Figure 9: Different local k' with different privacy budget.

Table 1: Communication and computation cost comparison.

k'	method	Comm cost (per party)	Grid compute time $S = 2$	Grid compute time $S = 4$
5	DPFMPS-2PEst ($M = 2048$)	≈ 82 kB	4.75 s	35.40 s
	DPFMPS-2PEst ($M = 4096$)	≈ 164 kB	7.47 s	39.08 s
	IND-LAP	≈ 0.19 kB	0.61 s	1.32 s
	LDP-AGG-2PEst	≈ 160 kB	5.46 s	38.48 s
	non-priv	≈ 160 kB	0.15 s	1.12 s
8	DPFMPS-2PEst ($M = 2048$)	≈ 131 kB	11.68 s	98.91 s
	DPFMPS-2PEst ($M = 4096$)	≈ 260 kB	11.95 s	101.14 s
	IND-LAP	≈ 0.28 kB	0.64 s	3.83 s
	LDP-AGG-2PEst	≈ 160 kB	12.13 s	100.63 s
	non-priv	≈ 160 kB	0.19 s	3.26 s

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 ϵ 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 ADMM framework in the VFL setting. Another paper [12] discussing asynchronous supervised learning with VFL and DPSGD [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 [57] 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 sketches 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.

REFERENCES

[1] California consumer privacy act. https://leginfo.legislature.ca.gov/faces/codes_displayText.xhtml?division=3.&part=4.&lawCode=CIV&title=1.81.5.

[2] Eu general data protection regulation. https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32016R0679.

[3] Home credit default risk. https://www.kaggle.com/competitions/home-credit-default-risk/overview.

[4] New york city taxi trip. https://www.kaggle.com/c/nyc-taxi-trip-duration.

[5] 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 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, page 308–318, New York, NY, USA, 2016. Association for Computing Machinery.

[6] D. Aloise, A. Deshpande, P. Hansen, and P. Popat. Np-hardness of euclidean sum-of-squares clustering. *Machine learning*, 75(2):245–248, 2009.

[7] M.-F. Balcan, T. Dick, Y. Liang, W. Mou, and H. Zhang. Differentially private clustering in high-dimensional euclidean spaces. In *International Conference on Machine Learning*, pages 322–331. PMLR, 2017.

[8] A. Blum, C. Dwork, F. McSherry, and K. Nissim. Practical privacy: the SuLQ framework. In *Proceedings of the twenty-fourth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 128–138, 2005.

[9] N. Carlini, C. Liu, Ú. Erlingsson, J. Kos, and D. Song. The secret sharer: Evaluating and testing unintended memorization in neural networks. In *28th USENIX Security Symposium (USENIX Security 19)*, pages 267–284, 2019.

[10] A. Chang, B. Ghazi, R. Kumar, and P. Manurangsi. Locally private k-means in one round. *arXiv preprint arXiv:2104.09734*, 2021.

[11] M. S. Charikar. Similarity estimation techniques from rounding algorithms. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 380–388, 2002.

[12] T. Chen, X. Jin, Y. Sun, and W. Yin. VAFL: a method of vertical asynchronous federated learning, 2020.

[13] S. G. Choi, D. Dachman-soled, M. Kulkarni, and A. Yerukhimovich. Differentially-private multi-party sketching for large-scale statistics. *Proceedings on Privacy Enhancing Technologies*, 3:153–174, 2020.

[14] E. D. Cristofaro, P. Gasti, and G. Tsudik. Fast and private computation of cardinality of set intersection and union. In *International Conference on Cryptology and Network Security*, pages 218–231. Springer, 2012.

[15] D. Desfontaines, A. Lochbihler, and D. Basin. Cardinality estimators do not preserve privacy. *Proceedings on Privacy Enhancing Technologies*, 2:26–46, 2019.

[16] C. Dickens, J. Thaler, and D. Ting. (nearly) all cardinality estimators are differentially private, 2022.

[17] W. DIFFIE and M. E. HELLMAN. New directions in cryptography. *IEEE TRANSACTIONS ON INFORMATION THEORY*, 22(6), 1976.

[18] H. Ding, Y. Liu, L. Huang, and J. Li. K-means clustering with distributed dimensions. In *International Conference on Machine Learning*, pages 1339–1348. PMLR, 2016.

[19] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *TCC*, pages 265–284, 2006.

[20] C. Dwork and K. Nissim. Privacy-preserving datamining on vertically partitioned databases. In *CRYPTO*, pages 528–544, 2004.

[21] C. Dwork, A. Smith, T. Steinke, and J. Ullman. Exposed! a survey of attacks on private data. *Annual Review of Statistics and Its Application*, 4:61–84, 2017.

[22] D. Feldman, A. Fiat, H. Kaplan, and K. Nissim. Private coresets. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pages 361–370, 2009.

[23] P. Flajolet, É. Fusy, O. Gandouet, and F. Meunier. Hyperloglog: the analysis of a near-optimal cardinality estimation algorithm. In *Discrete Mathematics and Theoretical Computer Science*, pages 137–156. Discrete Mathematics and Theoretical Computer Science, 2007.

[24] P. Flajolet and G. N. Martin. Probabilistic counting algorithms for data base applications. *Journal of computer and system sciences*, 31(2):182–209, 1985.

[25] M. J. Freedman, K. Nissim, and B. Pinkas. Efficient private matching and set intersection. In *International conference on the theory and applications of cryptographic techniques*, pages 1–19. Springer, 2004.

[26] B. Ghazi, R. Kumar, and P. Manurangsi. Differentially private clustering: Tight approximation ratios. *Advances in Neural Information Processing Systems*, 33, 2020.

[27] Google. Differentially private k-means clustering (experimental). https://github.com/google/differential-privacy/tree/main/learning/clustering, 2022.

[28] A. Groce, P. Rindal, and M. Rosulek. Cheaper private set intersection via differentially private leakage. *Proceedings on Privacy Enhancing Technologies*, 2019(3), 2019.

[29] B. Gu, Z. Dang, X. Li, and H. Huang. Federated doubly stochastic kernel learning for vertically partitioned data. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2483–2493, 2020.

[30] O. Gupta and R. Raskar. Distributed learning of deep neural network over multiple agents. *Journal of Network and Computer Applications*, 116:1–8, 2018.

[31] S. Har-Peled and S. Mazumdar. On coresets for k-means and k-median clustering. In *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, pages 291–300, 2004.

[32] J. A. Hartigan and M. A. Wong. Algorithm as 136: A k-means clustering algorithm. *Journal of the royal statistical society. series c (applied statistics)*, 28(1):100–108, 1979.

[33] C. Hazay and Y. Lindell. Efficient protocols for set intersection and pattern matching with security against malicious and covert adversaries. In *Theory of Cryptography Conference*, pages 155–175. Springer, 2008.

[34] C. Hazay and K. Nissim. Efficient set operations in the presence of malicious adversaries. In *International Workshop on Public Key Cryptography*, pages 312–331. Springer, 2010.

[35] C. Hu, J. Li, Z. Liu, X. Guo, Y. Wei, X. Guang, G. Loukides, and C. Dong. How to make private distributed cardinality estimation practical, and get differential privacy for free. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 965–982, 2021.

[36] Y. Hu, P. Liu, L. Kong, and D. Niu. Learning privately over distributed features: An admm sharing approach, 2019.

[37] Y. Hu, D. Niu, J. Yang, and S. Zhou. FDML: A collaborative machine learning framework for distributed features. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, page 2232–2240, New York, NY, USA, 2019. Association for Computing Machinery.

[38] Z. Huang and J. Liu. Optimal differentially private algorithms for k-means clustering. In *Proceedings of the 37th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*, pages 395–408, 2018.

[39] S. Jarecki and X. Liu. Efficient oblivious pseudorandom function with applications to adaptive ot and secure computation of set intersection. In *Theory of Cryptography Conference*, pages 577–594. Springer, 2009.

[40] B. Kacsmar, B. Khurram, N. Lukas, A. Norton, M. Shafieinejad, Z. Shang, Y. Baseri, M. Sepehri, S. Oya, and F. Kerschbaum. Differentially private two-party set operations. In *2020 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 390–404. IEEE, 2020.

[41] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings, et al. Advances and open problems in federated learning. *Foundations and Trends® in Machine Learning*, 14(1–2):1–210, 2021.

[42] L. Kissner and D. Song. Privacy-preserving set operations. In *Annual International Cryptology Conference*, pages 241–257. Springer, 2005.

[43] B. Kreuter, C. W. Wright, E. S. Skvortsov, R. Mirisola, and Y. Wang. Privacy-preserving secure cardinality and frequency estimation. 2020.

[44] K. J. Lang. Back to the future: an even more nearly optimal cardinality estimation algorithm. *arXiv preprint arXiv:1708.06839*, 2017.

[45] J. Li, N. Li, and B. Ribeiro. Membership inference attacks and defenses in classification models. In *Proceedings of the Eleventh ACM Conference on Data and Application Security and Privacy*, pages 5–16, 2021.

[46] Z. Li, B. Ding, C. Zhang, N. Li, and J. Zhou. Federated matrix factorization with privacy guarantee. *Proc. VLDB Endow.*, 15(4):900–913, dec 2021.

[47] Z. Li, T. Wang, and N. Li. Differentially private vertical federated clustering. *arXiv preprint arXiv:2208.01700*, 2022.

[48] Y. Liu, Y. Liu, Z. Liu, Y. Liang, C. Meng, J. Zhang, and Y. Zheng. Federated forest. *IEEE Transactions on Big Data*, (01):1–1, 2020.

[49] J. MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA, 1967.

[50] J. Matoušek. On approximate geometric k-clustering. *Discrete & Computational Geometry*, 24(1):61–84, 2000.

[51] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y. Arcas. Communication-Efficient Learning of Deep Networks from Decentralized Data. In A. Singh and J. Zhu, editors, *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pages 1273–1282, USA, 20–22 Apr 2017. PMLR.

[52] H. B. McMahan, D. Ramage, K. Talwar, and L. Zhang. Learning differentially private recurrent language models. In *International Conference on Learning Representations*. OpenReview.net, 2018.

[53] I. Mironov. Rényi differential privacy. In *2017 IEEE 30th computer security foundations symposium (CSF)*, pages 263–275. IEEE, 2017.

[54] K. Nissim, S. Raskhodnikova, and A. Smith. Smooth sensitivity and sampling in private data analysis. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 75–84, 2007.

[55] K. Nissim and U. Stemmer. Clustering algorithms for the centralized and local models. In *Algorithmic Learning Theory*, pages 619–653. PMLR, 2018.

[56] K. Nissim, U. Stemmer, and S. Vadhan. Locating a small cluster privately. In *Proceedings of the 35th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*, pages 413–427, 2016.

[57] R. Pagh and N. M. Stausholm. Efficient differentially private F0 linear sketching. In *24th International Conference on Database Theory, ICDT 2021, March 23-26, 2021, Nicosia, Cyprus*, volume 186 of *LIPIcs*, pages 18:1–18:19. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.

[58] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, 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.

[59] 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.

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

[61] 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.

[62] D. J. Slate. Letter recognition data set. https://archive.ics.uci.edu/ml/datasets/letter+recognition.

[63] 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.

[64] R. Stanojevic, M. Nabeel, 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.

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

[66] 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.

[67] 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.

[68] 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.

[69] 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.

[70] 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.

[71] 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.

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

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

[74] 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.

[75] 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.

[76] 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):2090–2103, 2020.

[77] 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.

[78] 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 543–546. IEEE, 2009.

[79] 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.