Privacy-Preserving Multi-Party Clustering: An Empirical Study

Arlei Silva
University of California
Santa Barbara, CA
arlei@cs.ucsb.edu

Gowtham Bellala
C3 IoT
Redwood City, CA
gowtham.bellala@c3iot.com

***Abstract*—Enterprises are transitioning towards data-driven business processes. There are numerous situations where multiple parties would like to share data towards a common goal, if it were possible to simultaneously protect the privacy and security of the individuals and organizations described in the data. Motivated by the increasing demands for data privacy, this paper provides the first comprehensive evaluation of privacy-preserving multi-party computation. As a case study, we consider the clustering task, which consists of grouping a set of points based on their similarity. Our goal is to understand the trade-offs involved when different parties want to collaboratively perform a computation while preserving their data privacy. In particular, we study the implications of centralized and distributed privacy-preserving solutions (such as encryption and data perturbation) on clustering quality, privacy and computational performance. Our results offer a new perspective on multi-party computation for both service providers and users, highlighting the drawbacks of these approaches and opening opportunities for future research.**

I. Introduction

A hot topic in enterprises today is how to enable data-driven business processes. Advances in computer hardware and networks, the development of cloud computing technology, and the creation of "Big Data" analytics software, have made it possible to analyze vast quantities of data. However, one important challenge is how to extract information from a dataset that can facilitate good business decisions, without sacrificing the privacy of the individuals or organizations whose sensitive details may be contained in the data [15]. This challenge is compounded when it involves multiple organizations (or "parties") wishing to collaborate in order to obtain a broader understanding of a topic of mutual interest.

For example, consider a scenario where a group of hospitals wish to collaborate to improve their collective quality of healthcare [17]. Each hospital already collects a lot of data about its patients, including their demographics, past medical history, lab results, current diagnosis, and prescribed treatment and outcomes. This data contains a wealth of information that if shared across the group could mutually benefit all parties by enabling faster diagnosis and effective treatment for similar cases. However, this data contains extremely sensitive and private information both about the patients and the hospitals. Thus, for a variety of reasons (including regulatory[1]), sharing this sort of data can be problematic. This hinders the desire of the organizations to become more data-driven.

Current commercial solutions require each party to share their raw data or local results with a third party or a mediator, and have the mediator compute and share the final results. This mediator is often a cloud server assumed to be *semi-honest*, while data belonging to the other parties might also be in the cloud, possibly using distinct cloud service providers. However, for reasons such as recent cyber attacks and resulting massive privacy breaches (e.g., Target, Boston Medical and Goodwill), many organizations are understandably hesitant to share either raw or aggregate data with third parties[2]. In particular, ensuring the security of the data is usually a responsibility of the parties and not the mediator.

Privacy-Preserving Data Mining (PPDM) [18] enables data analytics while respecting privacy constraints. However, the several PPDM approaches proposed by previous work (e.g., encryption, data perturbation) have important trade-offs that have not been well-documented in the literature. In this work, we address this gap by providing an in-depth analysis of a diverse set of privacy-preserving techniques in the multi-party setting. In particular, we evaluate these techniques; quantify their trade-offs in terms of quality, privacy and computational performance; and study their scalability properties.

In this work, we focus on the privacy-preserving multi-party clustering problem. Clustering is a typical data analytics task that arises in various applications such as customer segmentation, information retrieval and frequent pattern extraction. We study and evaluate eight different privacy-preserving, multi-party clustering techniques on several datasets. Our experiments provide interesting insights into the trade-offs among these techniques and also motivate the design of new solutions for privacy-preserving multi-party data mining. This study is particularly relevant for software-as-a-service (SaaS) providers and users that have to choose the right multi-party privacy-preserving data mining approach based on the application. For instance, we highlight important security limitations of the so called *data lake* paradigm, where raw data is transferred to a centralized server, compared to distributed solutions, where only local results are shared during the computation.

Work done while the authors were at Hewlett Packard Labs, Palo Alto, CA

[1]https://www.congress.gov/bill/105th-congress/senate-bill/1368

[2]http://www.esecurityplanet.com/network-security/third-party-vendors-a-weak-link-in-security-chain.html

Computation	Mediator	Privacy	What is shared
Local	–	–	–
Centralized	Trusted	–	Local data
Centralized	Untrusted	ADP	Perturbed data
Centralized	Untrusted	RSP	Projected data
Distributed	Trusted	–	Partial results
Distributed	Untrusted	ADP	Perturbed results
Distributed	Untrusted	RSP	Projected results
Distributed	Untrusted	SMC	Encrypted results

TABLE I: Overview of the privacy-preserving multi-party clustering approaches studied in this paper.

II. Background

Privacy-preserving data mining (PPDM): PPDM studies the design of data management and mining algorithms that help extract useful knowledge while preserving the data privacy [2], [9], [18]. PPDM algorithms can be classified based on several aspects such as the data distribution model (e.g., single/multi-party, vertical/horizontal partitioning), the data obfuscation scheme (e.g., encryption, random noise), the adversarial model (e.g., malicious, semi-honest), and the data mining task (e.g. clustering, classification) [19]. The obfuscation schemes studied in the literature can be broadly classified into two categories, *randomization*-based techniques and *cryptography*-based techniques. Randomization techniques [2] perturb individual records of the database so that information regarding each individual record is masked while aggregate information can still be approximated using perturbed data. In this work, we study two popular randomization techniques for data obfuscation, namely additive data perturbation and random subspace projection. In addition, we also study a cryptography-based technique that leverages principles from secure multi-party computation (SMC). An important aspect of this study is the evaluation of privacy-preserving clustering solutions. Evaluating PPDM algorithms is a major problem in data mining and management [1], [3]. We consider three dimensions in the evaluation of the algorithms: *quality*, *privacy* and *computational performance*.

Clustering and the K-means algorithm Given a set of objects and a measure of similarity, the goal of clustering [20] is to discover groups of objects that are similar to each other. This problem arises in applications such as image segmentation, frequent pattern discovery, information retrieval, patient subgroup identification, customer analysis and Bioinformatics. K-means [11] is a simple, iterative clustering algorithm. Given a set of objects in a database $\mathcal{D}$ and a constant k, it partitions the objects into k groups by minimizing the within-cluster Euclidean distance, where each cluster is represented by a centroid. K-means consists of two steps: an assignment step where each object is assigned to the closest cluster (based on its distance to the cluster centroid), and an update step where the centroid of each cluster is updated based on the objects assigned in the above step. These steps are repeated a pre-determined number of times or until convergence. At the end of the computation, each object is assigned to a single cluster. Here, we consider a pre-determined number of iterations m as the stopping criteria for k-means.

III. Privacy-Preserving Multi-party Clustering

We study clustering in a multi-party setting, where parties want to collaboratively cluster their global data while preserving the data privacy. We assume that the data is distributed across a set of machines/servers owned by these parties.

In general, each party z owns a database D_z which is stored locally (e.g., in a database server owned by z). We define a conceptual global database $\mathcal{D} = \cup_z D_z$ as the union of the party-owned databases. Note that, in practice $\mathcal{D}$ may not be materialized as an actual database due to privacy, regulatory or other constraints. Each party's local data is considered private, and hence they would like to limit the amount of information that is shared during the computation.

We measure privacy as the amount of information the mediator can acquire from the data shared by the parties. In other words, it is the amount of private information that can be revealed if the mediator is compromised by an adversary. Raw data (i.e., individual record and attribute values) corresponding to each party is considered to be private information.

A. Local computation

Naive approach where each party performs clustering independently using k-means on their local data. There is no need for a mediator or data obfuscation as no data is shared between parties. This approach is expected to achieve low-latency and no communication cost, while complete privacy is guaranteed. However, the clustering quality might be compromised due to the lack of collaboration (i.e., data sharing).

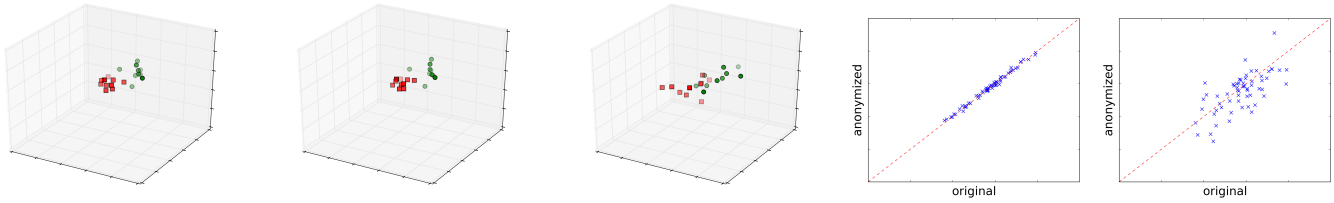
B. Centralized computation

In this approach, parties transfer their raw or obfuscated data to the mediator, who then applies the k-means algorithm on the global database $\mathcal{D}'$. Each resulting cluster C_j^m is divided into subsets $C_j^{m,1}, C_j^{m,2}, \cdots, C_j^{m,p}$, where $C_j^{m,i}$ corresponds to the set of objects in cluster j belonging to party i, i.e., $C_j^{m,i} = \{x \in C_j^m | x \in D_i\}$. Finally, the mediator shares each cluster subset $\{C_j^{m,z} : 1 \leq j \leq k\}$ with their respective data owner z, $1 \leq z \leq p$. We evaluate three different methods for centralized computation as described below.

In the *trusted mediator* setting, each party z sends its raw data D_z to the mediator and the joint database $\mathcal{D}'$ is the union of the local (raw) databases (i.e., $\mathcal{D}' = \mathcal{D}$).

In case of an *untrusted mediator*, the parties share obfuscated data with the mediator. We study two popular data obfuscation schemes from the literature: *Additive Data Perturbation (ADP)* and *Random Subspace Projection (RSP)*.

Additive Data Perturbation (ADP) [12]: Each party z generates a noisy version D'_z of its database by adding i.i.d (independent and identically distributed) Gaussian noise. An entry x_{ij} in the original database D_z is anonymized as $x'_{ij} = x_{ij} + \epsilon_{ij}$ to generate the noisy database D'_z, where $\epsilon_{ij} \sim N(0, \sigma)$. The mediator aggregates $D'_1, D'_2, \cdots, D'_p$ into a single database $\mathcal{D}'$, and applies k-means on the noisy data. The intuition behind this approach is that while the data objects are perturbed by the noise, the position of the cluster centroids are preserved by the law of large numbers. Figure 1 illustrates



(a) Original data (b) Anonymization I (c) Anonymization II (d) Correlation I (e) Correlation II

Fig. 1: Additive data perturbation on a toy dataset using $\sigma = 0.01$ (scenario I) and $\sigma = 0.1$ (scenario II). The correlation between the original and anonymized data for scenarios I and II is shown in (d) and (e), respectively.

ADP on a toy dataset with 20 objects and 2 clusters. Figure 1a shows the original data, while Figures 1b and 1c show the anonymized data using a noise level of $\sigma = 0.01$ and $\sigma = 0.1$, respectively. The relationship between original and anonymized data for the noise levels—shown in Figures 1d and 1e– demonstrates the privacy vs. quality trade-off in ADP.

Random Subspace Projection (RSP) [10]: In this approach, the privacy of the data is preserved by projecting the data onto a non-invertible low dimensional subspace:

$$x' = \frac{1}{\sqrt{q}\sigma} xR$$

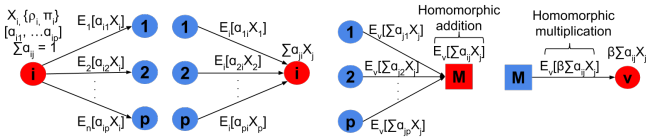
where R is a $d \times q$ ($q < d$) random Gaussian matrix (i.e., $r_{ij} \sim N(0, \sigma)$) with q and d denoting the number of dimensions in the projected and original space, respectively.

Each party z generates a projected version D'_z of its local data using a common random matrix R. While the random projection masks the original data, it preserves the pairwise Euclidean distances, enabling the mediator to perform clustering on the projected data. Figure 2 illustrates RSP on a toy dataset. Figure 2a shows the original 3-D dataset, while Figure 2b shows the data projected onto a 2-D space. Figure 2c shows data reconstructed from the 2-D projection if the random matrix R is revealed to the mediator (privacy attacks are described in Section IV-A). The relationship between the original and projected data is shown in Figure 2d. Note that while the projected data preserves the clusters, the reconstructed data differs significantly from the original, demonstrating the privacy-preserving property of RSP.

C. Distributed computation

In a distributed scenario, each party performs computations locally on its database D_z and shares the local results with the mediator. The mediator aggregates these results to compute the global centroids and sends them back to the parties. These steps are repeated m times until the parties receive the final clustering assignments. We use the symbol $C_j^{t,z}$ to denote the j-th cluster in the database owned by party z at iteration t. Local sums $S_j^{t,z}$ and counts $N_j^{t,z}$ sent by each party are used by the mediator in the computation of global centroids $\mu(C_j^t)$.

In case of a *trusted mediator*, parties share their local results with the mediator, without any data obfuscation. For an *untrusted mediator*, parties obfuscate their local results before sharing them with the mediator. We first describe the use of Additive Data Perturbation and Random Subspace Projection for data obfuscation in a distributed setting. We then describe a *Secure Multiparty Computation (SMC)* based approach that uses data encryption to preserve privacy.



(a) Secure random sharing (b) Secure aggregation

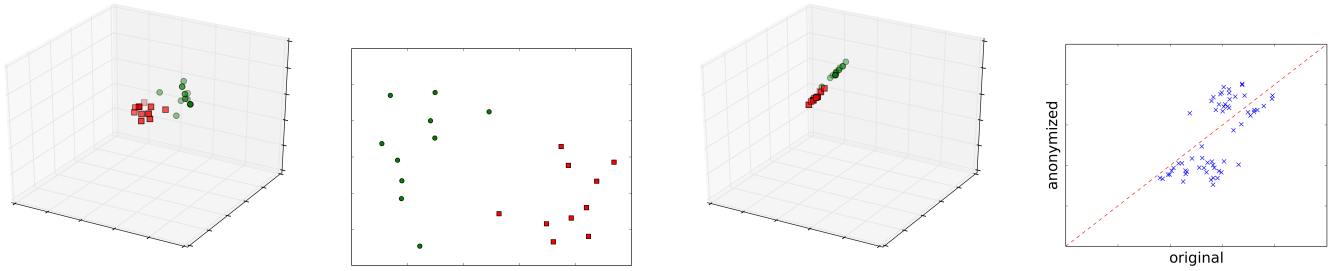
Fig. 3: Secure multiparty clustering

Additive Data Perturbation (ADP): Each party adds i.i.d. Gaussian noise $\epsilon \sim N(0, \sigma)$ to their local sums S_j^t before sharing them with the mediator. The mediator aggregates the local results and sends updated centroids to the parties.

Random Subspace Projection (RSP): Each party z projects their database D_z using a common random matrix R (similar to the centralized setting). However, instead of sharing projected data D'_z with the mediator, the parties compute local results using D'_z and share them with the mediator. Therefore, clustering is performed entirely in the projected space.

Secure Multiparty Computation (SMC) [7] Each party obfuscates its local results using a combination of randomized sharing and *partially homomorphic encryption*. We use the *Paillier cryptosystem* [13], which supports homomorphic addition ($E[x] * E[y] = E[x+y]$) and multiplication ($E[x]^y = E[x * y]$). SMC adds two additional steps to the distributed k-means algorithm: a secure random sharing (Figure 3a) and a secure aggregation (Figure 3b).

Without loss of generality, assume that the p parties have to compute a masked sum $\beta X = \beta \sum_{1 \le i \le p} X_i$. Each party i starts by generating a public-private key pair (ρ_i, π_i). Public keys are shared with all the parties involved and the mediator. In the secure random sharing, party i generates secret random shares $\alpha_{i,1}, \ldots, \alpha_{i,p}$ uniformly such that $\sum_j \alpha_{ij} = 1$ and $\alpha_{ij} \ge 0$. These shares are applied to decompose a party's local value X_i into p components $\alpha_{i,1}X_i, \ldots, \alpha_{i,p}X_i$ such that $X_i = \sum_j \alpha_{ij} X_i$. Using the public key ρ_j, parties encrypt local sum shares $\alpha_{ij} X_i$ and send them to party j. Party j then



(a) Original data (b) Anonymization (c) Reconstructed data (d) Correlation

Fig. 2: Random subspace projection with 2 dimensions using a toy dataset. Figure (c) shows the reconstructed (i.e. de-anonymized using an attack) and Figure (c) shows the correlation between the original and reconstructed data

decrypts these shares using its private key π_j and compute $\sum_i \alpha_{ij} X_i$. As a result, the value X_i is securely shared among the p parties, since the values of shares α_{ij} are kept secret.

In the secure aggregation step, parties collaboratively decide on an *aggregator* $1 \leq v \leq p$ and use ρ_v to encrypt the partial sums $\sum_i \alpha_{ij} X_i$. Encrypted sums are sent to the mediator, which applies homomorphic addition and multiplication to compute the encrypted value of $\beta X = \beta \sum_j \sum_i \alpha_{ij} X_i$. The constant β is a secret multiplier that masks the actual value of X. The mediator then sends the encrypted value of βX to party v, which decrypts it using its private key π_v.We apply this protocol to securely compute both the sums S_j^t and counts N_j^t in the distributed k-means algorithm. As long as the same value of β is applied for both the sums and the counts, the aggregator v can compute the centroid $\mu(C_j^{t+1})$ as $\frac{1}{\beta N_j^t} \beta S_j^t$.

IV. Privacy Evaluation

In this section, we describe how the multi-party clustering algorithms are evaluated in terms of privacy. We first study various attacks that can be launched by an adversary as a result of the mediator being compromised or due to collusion between parties and the mediator. We then present two metrics to evaluate the privacy of the algorithms under these attacks.

A. Attacks on multi-party clustering

We divide the privacy attacks into those for centralized and distributed settings and describe them per obfuscation scheme used. In each case, the adversary attempts to re-construct the original data with the information it obtains. We assume that the local method is not susceptible to attacks as no information is disclosed. Similarly, the proposed SMC method does not reveal any information even if the mediator is compromised, as only encrypted data is shared with the mediator.

Centralized: In the centralized approaches, parties disclose a value x' for each entry x in their private database. We apply four privacy attacks for the centralized ADP method: (1) *With no additional information:* Adversary reconstructs the original data x by assuming it to be equal to the noisy data x', i.e., $\widehat{x} = x'$; (2) *When noise distribution is known:* Adversary applies well-known properties of random matrices to reconstruct the original data [8], [5]; (3) *When noise and prior data distribution are known:* A maximum a posteriori (MAP) estimate is applied to infer the original data x from x' and the prior distribution $P(x)$[5]. In the case of centralized RSP, we consider two privacy attacks described in [10]: (1) *When noise distribution is known:* The adversary uses the noise distribution to generate a projection matrix R' that can be used to reconstruct x from x'; and (2) *When the projection matrix R is known:* Adversary performs a *least-squares* (or *pseudoinverse*) attack to reconstruct x. Figure 2 shows an example of the pseudoinverse attack on a toy dataset.

Distributed: In the distributed clustering methods, parties only share their local centroid information (sums and counts) with the mediator. Hence, an adversary with access to this data can at best reconstruct the data by approximating each data point x in the original database with its closest centroid. For ADP, we consider an attack where only the perturbed centroid data is available to the adversary. In this case, the adversary would re-construct the data as described above using the noisy centroids. In RSP, we assume an attack where the adversary has access to the projected centroids along with the random projection matrix R. In this case, the adversary reconstructs the original centroids using the least-squares solution, and then approximates the original data as described above.

B. Privacy metrics

Given the original data x and the reconstructed data $\widehat{x}$ using the attacks described above, we evaluate the privacy of an algorithm using two different metrics, the *Root Mean Squared Error (RMSE)* and the *Conditional Privacy Loss (CPL)*. Given an attribute j, the RMSE is defined as:

$$\mathcal{R}_j = \sqrt{\frac{\sum_{i=1}^{n} (x_{ij} - \widehat{x_{ij}})^2}{n}}$$

The conditional privacy loss [1] applies *entropy* to assess the impact of an attack. The CPL $\mathcal{P}(\mathbf{x}_j|\widehat{\mathbf{x}_j})$ of an attribute $\mathbf{x}_j$ given its reconstructed version $\widehat{\mathbf{x}_j}$ is computed as:

$$\mathcal{P}(\mathbf{x}_j|\widehat{\mathbf{x}_j}) = 1 - \frac{2^{h(\mathbf{x}_j|\widehat{\mathbf{x}_j})}}{2^{h(\mathbf{x}_j)}}$$

where $h(\mathbf{x}_j|\widehat{\mathbf{x}_j})$ is the conditional differential entropy of $\mathbf{x}_j$ given $\widehat{\mathbf{x}_j}$.

name	# objects	# dimensions	# clusters
SYNTHETIC	50K	10	10
HEART	920	13	4
CANCER	198	20	15
DIABETES	100K	12	12
GAS	320K	16	10

TABLE II: Dataset statistics.

V. EXPERIMENTS

A. Overview of the testbed

We used an *Amazon AWS EC2*cluster with 10-16 nodes as our testbed. Each node is a single-core, 2.40GHz Intel Xeon machine with 1GB of RAM and 30GB of hard disk running *Debian GNU/Linux*, and with an estimated network bandwidth of 2.7 MBps. We implemented all the clustering approaches under the same framework using *Python* version 2.7.3.

B. Datasets

We used both synthetic and real datasets to evaluate the clustering approaches described in this paper. Motivated by healthcare applications, we used several medical datasets in our study. We standardized the attribute values using a z-score normalization to have a zero mean and unit variance. Table II provides a summary of the datasets used.

SYNTHETIC: To facilitate a controlled study and support our scalability analysis, we used a synthetic dataset generated using a Gaussian mixture model. Each cluster follows a multivariate Gaussian distribution with a random center and within a bounding-box. The parameters of the generator include the number of objects, clusters, dimensions and the standard deviation of each cluster (i.e., amount of variance). We vary these parameters in our scalability analysis. To study the trade-offs of the clustering methods, we used a synthetic dataset with $50,000$ objects, 10 clusters and 10 dimensions.

HEART: This dataset consists of 920 patient records from four hospitals [4]. Each patient record contains 13 attributes (e.g., age, cholesterol level) related to diagnostics for heart disease. This dataset also contains the ground truth labels for a coarse clustering that denotes the likelihood of a heart disease in a patient: 0 for healthy, and 1-4 for unhealthy.

CANCER: This dataset is primarily used for cancer research and consists of gene expression levels for 198 tissue samples and 16,063 genes [14]. Each object is a tissue sample with genes denoting the attributes. To remove redundant features and reduce the data dimensionality, we applied principal component analysis (PCA) and extracted the top 20 features in the data. The dataset has ground truth labels corresponding to 15 clusters that assign each tissue to being normal or to one of the 14 different tumor classes (e.g., prostate, colon, and melanoma). Among the 198 tissue samples, 90 are normal and 108 belong to one of the tumor classes.

DIABETES: This dataset consists of patient records from a large-scale Diabetes study with 101,765 patients [16]. We selected 12 numerical patient attributes (e.g., age, length of stay in hospital) as features in our study. This dataset does not include ground truth information. Hence, we used the cluster assignments obtained by the standard k-means algorithm on the global dataset as ground truth.

GAS: This dataset consists of chemical sensor readings from a gas delivery platform facility[3]. It consists of time-series measurements from 16 different sensors, where each time measurement is a sample and each sensor is an attribute in the clustering algorithm. The dataset consists of 320,000 time samples for each sensor. We used this dataset as part of our scalability experiments in Section V-E. This dataset does not have ground truth clusters and hence we assumed the number of clusters to be 10 for experimental purposes.

C. Performance metrics

We evaluate the privacy-preserving multi-party clustering algorithms in terms of quality, privacy, and computational performance. Below, we describe the metrics applied.

Quality: We use two different measures to quantify the quality or accuracy of the clustering result. The *intra-cluster distance* measures the tightness of the clusters given by the total sum of pairwise distances between objects assigned to the same cluster. It does not require ground truth information.

$$\sum_{j=1}^{k} \sum_{x \in C_j} \sum_{y \in C_j} ||x - y||_2$$

The *Adjusted rand-score* computes the purity of the clusters based on ground truth. Given the ground truth labels $\mathcal{C}$ and the clustering result $\mathcal{C}'$, it computes an adjusted fraction of correctly assigned objects [6]. The adjustment takes into account the expected accuracy of a random assignment:

$$\frac{\binom{n}{2}(a+d) - ((a+b)(a+c)+(c+d)(b+d))}{\frac{1}{2}\binom{n}{2}^2 - ((a+b)(a+c)+(c+d)(b+d))}$$

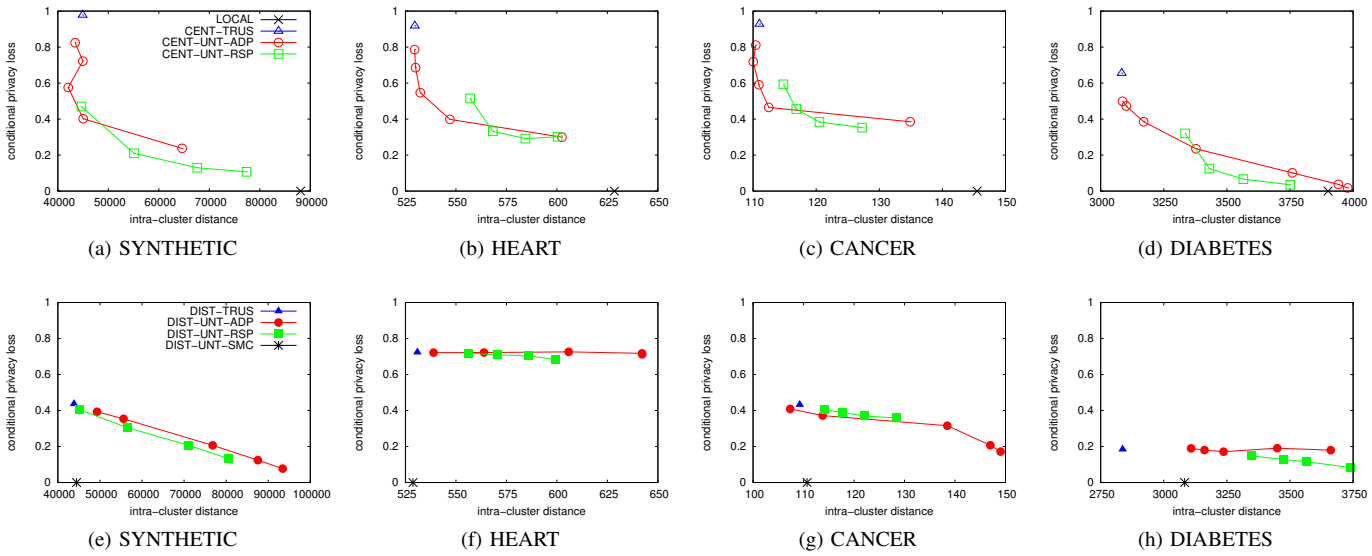
where a is the number of pairs of objects in the same cluster in $\mathcal{C}$ and $\mathcal{C}'$, b is the number of pairs of objects that are in different clusters in $\mathcal{C}$ and $\mathcal{C}'$, c is the number of pairs of objects that are in the same cluster in $\mathcal{C}$ but in different clusters in $\mathcal{C}'$, and d is the number of pairs of objects in different clusters in $\mathcal{C}$ but in the same cluster in $\mathcal{C}'$.

Computational performance: We evaluate the computational performance of these algorithms using *latency*, measured by the total wall-clock time (in seconds), and *communication cost*, as the total amount of data transferred (in bytes).

D. Evaluation of trade-offs

We evaluate the performance of privacy-preserving clustering algorithms on four datasets (SYNTHETIC, HEART, CANCER and DIABETES), and quantify their trade-offs in terms of quality, privacy, and computational performance. To simplify the discussion, we present our results in two groups, one consisting of centralized and local approaches, and the

[3]https://archive.ics.uci.edu/ml/datasets/Gas+sensor+array+under+dynamic+gas+mixtures



(a) SYNTHETIC (b) HEART (c) CANCER (d) DIABETES

(e) SYNTHETIC (f) HEART (g) CANCER (h) DIABETES

Fig. 4: Privacy (conditional privacy loss) vs. quality (intra cluster distance) for *local, centralized trusted, centralized untrusted ADP* and *centralized untrusted RSP* (a-d), and *distributed trusted, distributed untrusted ADP, distributed untrusted RSP* and *SMC* (e-h) using SYNTHETIC (a,e), CANCER (b,f), HEART (c,g), and DIABETES (d,h) datasets.

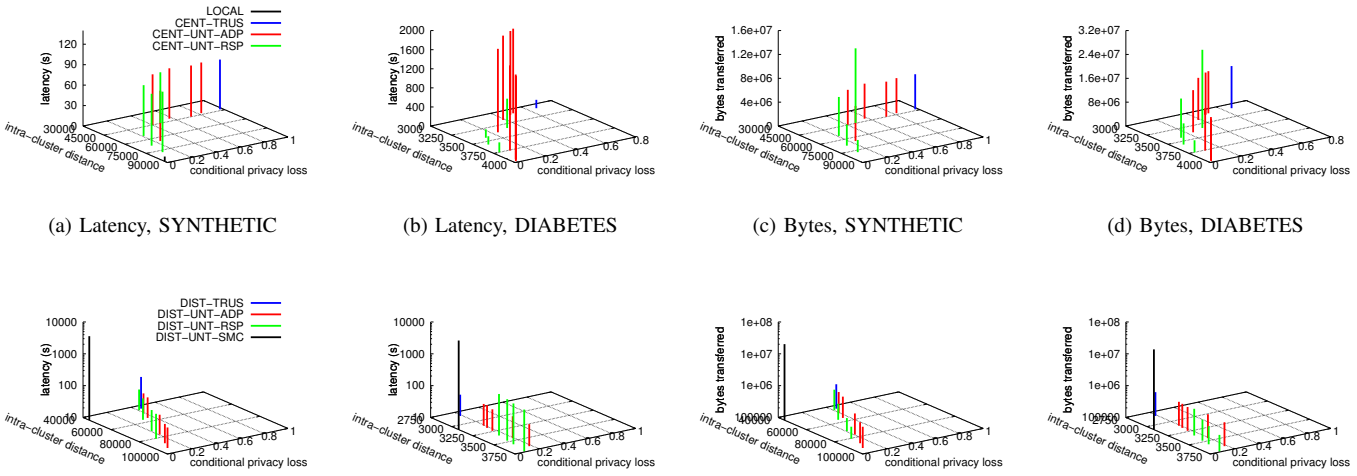
other consisting of distributed approaches. We use 10 parties for all the experiments in this section, and divide the data uniformly across the parties. For methods using ADP, we varied the noise level $\sigma = \{0.01, 0.02, 0.04, 0.08, 0.16\}$, and for RSP, we varied the dimensionality of the projected space as $q = \{10\%, 20\%, 40\%, 80\%\} * d$ (i.e., as a percentage of the original number of dimensions). Due to space constraints, we only present results for intra-cluster distance as quality metric and conditional privacy loss as privacy metric. Similar results were observed for other quality and privacy metrics.

Centralized and local: Figures 4a-4d demonstrate the trade-off between quality and privacy for the *local, centralized trusted, centralized untrusted ADP and centralized untrusted RSP* methods. Low intra-cluster distance and conditional privacy loss are desirable. The *centralized trusted* method achieves high accuracy, but it does not provide any protection to the data. On the other hand, the *local* method does not disclose any data, but achieves poor clustering quality. While these methods serve as two extremes, centralized *ADP* and *RSP* span the privacy vs. quality trade-off space. *RSP* tends to outperform *ADP* in the low-accuracy/high-privacy range, but *ADP* tends to be more effective in the high-accuracy/low-privacy range. Figures 5a and 5b demonstrate the trade-off between quality, privacy and latency. The local method is the fastest as it does not collaborate. The latency of *RSP* decreases with the dimensionality of the projected space. Moreover, the latency of *ADP* is an order of magnitude higher than that of the *centralized trusted* method, which is an overhead due to anonymization. With respect to the communication cost (Figures 5c and 5d), the *trusted* and *ADP* methods have similar performance. However, the communication cost of *RSP* decreases with the size of the projection space.

Distributed We study the same trade-offs for the distributed approaches: *distributed trusted, distributed untrusted ADP, distributed untrusted RSP*, and *distributed untrusted SMC*. Figures 4e-4h show the quality vs. privacy results. While the *distributed trusted* method attains high accuracy, it achieves only partial privacy as a result of sharing local aggregates with the mediator. *Distributed ADP* and *RSP* span the trade-off space as their centralized counterparts. However, compared to the centralized counterparts, these distributed methods tend to provide better privacy for similar quality levels. Finally, the *SMC* method outperforms all the other methods by achieving high accuracy and complete privacy (as a result of encryption). Nevertheless, note from Figures 5e-5h that the *SMC* method has at least two orders of magnitude higher latency and communication cost compared to other distributed methods, and is at least one order of magnitude worse than all centralized methods. This is a result of the use of data encryption and computationally intensive homomorphic operations over encrypted data. The *distributed ADP* and *RSP* methods outperform their centralized counterparts both in terms of latency and communication. For instance, *distributed ADP* achieves a $36\times$ speedup with around two orders of magnitude less communication, compared to *centralized ADP* with 10 parties.

E. Scalability experiments

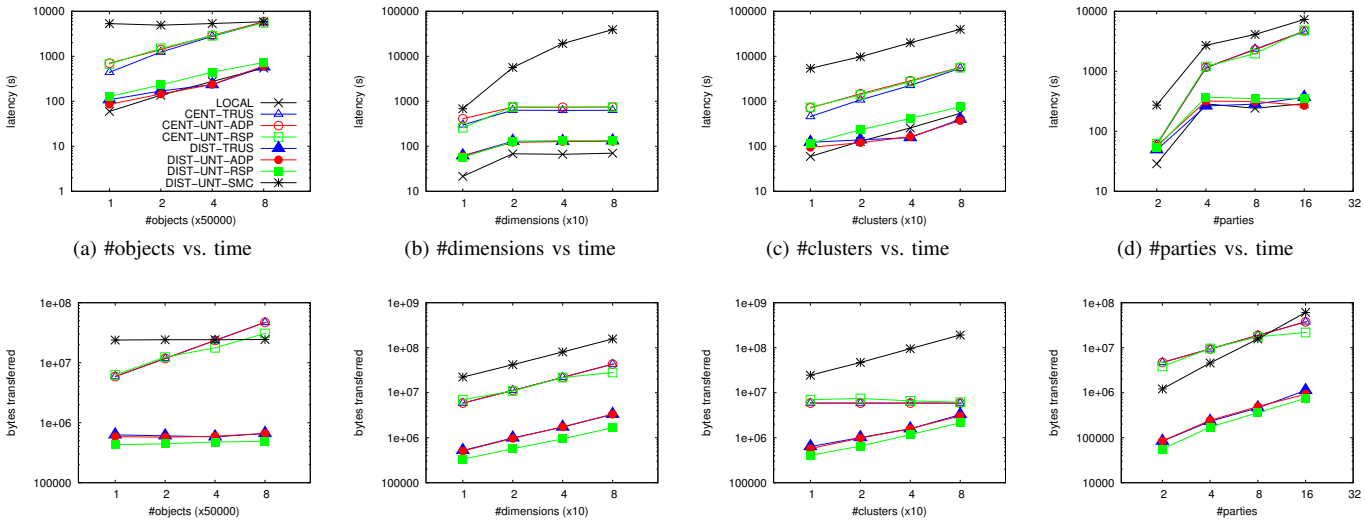
We also performed a scalability analysis to study the computational performance of the eight clustering methods as we increase the number of samples, number of dimensions (or attributes), number of clusters, and the number of parties. We used our synthetic data generator to produce datasets for these experiments. A base data configuration of 50K samples, 10 dimensions and 10 clusters was applied, and we scaled each



(a) Latency, SYNTHETIC (b) Latency, DIABETES (c) Bytes, SYNTHETIC (d) Bytes, DIABETES

(e) Latency, SYNTHETIC (f) Latency, DIABETES (g) Bytes, SYNTHETIC (h) Bytes, DIABETES

Fig. 5: Privacy vs. quality vs. latency (a,b,e,f) and privacy vs. quality vs. communication cost in bytes, (c,d,g,h) for local and centralized (a-d); and distributed (e-h) approaches using SYNTHETIC and DIABETES *This figure is better seen in color.*



(a) #objects vs. time (b) #dimensions vs time (c) #clusters vs. time (d) #parties vs. time

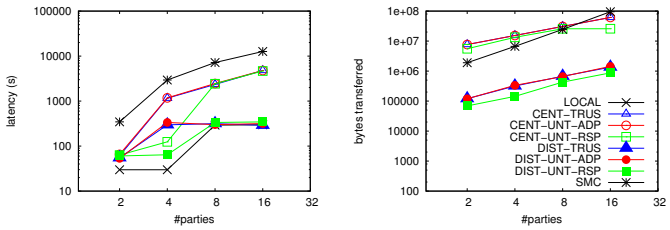
(e) #objects vs. bytes (f) #dimensions vs. bytes (g) #clusters vs. bytes (h) #parties vs. bytes

Fig. 6: Scalability results, in terms of latency and communication (bytes), using synthetic datasets and varying the number of objects (a,e), attributes (b,f), and clusters (c,g) in the dataset, and also the number of parties (d,h).

of these parameters by a factor of 2, 4 and 8. We varied the number of parties from 2 to 16 by a factor of 2. For the scalability analysis on the number of parties, we fixed the size of the dataset per party. We replicated this analysis on a real dataset (GAS), where we only increased the number of parties.

Figures 6a-6d show the scalability results for the latency, and Figures 6e-6h show the same for communication cost. Overall, the results show the methods can be grouped into three sets based on their scalability behavior: (1) *local*, *distributed trusted, distributed ADP,* and *distributed RSP*; (2) all *centralized* methods; and (3) *SMC*.

Group 1 contains the most efficient methods in terms of latency and communication cost. Their latency scales at most linearly with the four parameters. Regarding communication, the local method is omitted from the figures, as it does not transfer any data. Moreover, since the distributed methods only share aggregated results (a kxd matrix of local centroids), their communication cost does not change with the number of samples, but increases linearly with the remaining parameters. Group 2 consists of centralized methods where each party shares their entire raw or obfuscated data with the mediator to perform clustering. These methods inherently take more time



(a) #parties vs. time (b) #parties vs. bytes

Fig. 7: Scalability results, in terms of latency and communication (bytes), using the GAS dataset and varying the the number of parties.

than their distributed counterparts in Group 1. Furthermore, their communication increases linearly with the number of samples, dimensions, and parties, but is independent of the number of clusters. Finally, group 3 consists of the distributed SMC method that uses encryption to maintain privacy of the data. We noted earlier that the SMC method takes the most time and has the highest communication cost. However, as the number of samples is increased to $400K$, the SMC method takes the same amount of time and incurs a lower communication cost than the centralized methods. Similarly, as the number of parties is increased, the SMC tends to outperform the centralized methods.

We observed similar results on the GAS dataset (Figure 7).

VI. Discussion

We evaluated several multi-party clustering strategies that differ in terms of the *computation* model (local, centralized or distributed) used and the type of *mediator* (trusted or untrusted) assumed. For the scenarios with an untrusted mediator, we evaluated three privacy-preserving techniques: additive data perturbation, random subspace projection, and secure multi-party computation. Cloud service providers (e.g. SaaS) and users can apply these results as guidelines to select existing privacy-preserving multi-party analytics solutions and also as a motivation for the development of new solutions that address the limitations of existing alternatives.

There are interesting trade-offs involving clustering quality, privacy and computational performance, with no single algorithm being a winner in all possible scenarios. Moreover, while local computations (that do not involve collaboration among parties) achieve high privacy and high computational performance, the resulting quality is significantly low. In case these databases are private, randomization-based obfuscation techniques offer a trade-off among quality and privacy, with RSP outperforming ADP in most of the settings but ADP covering a broader privacy versus quality spectrum. When these methods are combined with distributed computation, they lead to highly scalable solutions and achieve similar or better privacy than their centralized counterparts without compromising quality. Finally, a secure multi-party computation approach achieves high quality and privacy, but tends to add a significant overhead in terms of latency and communication cost. However, as the size of the dataset or the number of parties increase, it outperforms the centralized solutions on all dimensions - quality, privacy, latency and communication cost.

This work opens several opportunities for future investigation. Although approaches such as the perturbation and projection-based methods enable the user to increase privacy by trading off on quality, none of these methods provide a similar knob to control the trade-off among privacy, quality and computational performance simultaneously. Moreover, general frameworks for privacy-preserving multi-party analytics, allowing different choices of privacy-preserving schemes and computation models, would facilitate the design of novel solutions for other relevant tasks (e.g. graph mining).

References

[1] D. Agrawal and C. C. Aggarwal. On the design and quantification of privacy preserving data mining algorithms. In *PODS*, 2001.

[2] R. Agrawal and R. Srikant. Privacy-preserving data mining. In *Sigmod Record*, 2000.

[3] E. Bertino, D. Lin, and W. Jiang. A survey of quantification of privacy preserving data mining algorithms. *Privacy-preserving data mining*, 34(1):183–205, 2008.

[4] R. Detrano, A. Janosi, W. Steinbrunn, M. Pfisterer, et al. International application of a new probability algorithm for the diagnosis of coronary artery disease. *American Journal of Cardiology*, 64(5):304–310, 1989.

[5] Z. Huang, W. Du, and B. Chen. Deriving private information from randomized data. In *SIGMOD*, 2005.

[6] L. Hubert and P. Arabie. Comparing partitions. *Journal of classification*, 2(1):193–218, 1985.

[7] G. Jagannathan and R. N. Wright. Privacy-preserving distributed k-means clustering over arbitrarily partitioned data. In *SIGKDD*, 2005.

[8] H. Kargupta, S. Datta, Q. Wang, and K. Sivakumar. On the privacy preserving properties of random data perturbation techniques. In *ICDM*, 2003.

[9] K. Liu, C. Giannella, and H. Kargupta. A survey of attack techniques on privacy-preserving data perturbation methods. *Privacy-Preserving Data Mining*, 34(1):359–381, 2008.

[10] K. Liu, H. Kargupta, and J. Ryan. Random projection-based multiplicative data perturbation for privacy preserving distributed data mining. *TKDE*, 18(1), 2006.

[11] J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, 1967.

[12] S. R. Oliveira and O. R. Zaiane. Privacy preserving clustering by data transformation. *Journal of Information and Data Management*, 1(1):37–51, 2010.

[13] P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *EUROCRYPT*, 1999.

[14] S. Ramaswamy, P. Tamayo, R. Rifkin, S. Mukherjee, et al. Multiclass cancer diagnosis using tumor gene expression signatures. *PNAS*, 98(26):15149–15154, 2001.

[15] M. D. Ryan. Cloud computing privacy concerns on our doorstep. *Communications of the ACM*, 54(1):36–38, 2011.

[16] B. Strack, J. P. DeShazo, C. Gennings, et al. Impact of hba1c measurement on hospital readmission rates: analysis of 70,000 clinical database patient records. *BioMed Research International*, 2014.

[17] H. Tang, X. Jiang, X. Wang, S. Wang, H. Sofia, D. Fox, K. Lauter, B. Malin, A. Telenti, L. Xiong, et al. Protecting genomic data analytics in the cloud: state of the art and opportunities. *BMC medical genomics*, 9(1):63, 2016.

[18] J. Vaidya and C. Clifton. Privacy-preserving data mining: Why, how, and when. *Security & Privacy*, (6):19–27, 2004.

[19] V. S. Verykios, E. Bertino, I. N. Fovino, L. P. Provenza, Y. Saygin, and Y. Theodoridis. State-of-the-art in privacy preserving data mining. *Sigmod Record*, 33(1):50–57, 2004.

[20] R. Xu, D. Wunsch, et al. Survey of clustering algorithms. *Transactions on Neural Networks*, 16(3):645–678, 2005.