APSIPA Transactions on Signal and Information Processing, 2023, 12, e4

Original Paper

Federated Analytics: A Survey

Ahmed Roushdy Elkordy[1], Yahya H. Ezzeldin[1], Shanshan Han[2], Shantanu Sharma[3], Chaoyang He[4], Sharad Mehrotra[2] and Salman Avestimehr[1]*

[1] *University of Southern California, USA*
[2] *University of California, Irvine, USA*
[3] *New Jersey Institute of Technology, USA*
[4] *FedML Inc., USA*

ABSTRACT

Federated analytics (FA) is a privacy-preserving framework for computing data analytics over multiple remote parties (e.g., mobile devices) or silo-ed institutional entities (e.g., hospitals, banks) without sharing the data among parties. Motivated by the practical use cases of federated analytics, we follow a systematic discussion on federated analytics in this article. In particular, we discuss the unique characteristics of federated analytics and how it differs from federated learning. We also explore a wide range of FA queries and discuss various existing solutions and potential use case applications for different FA queries.

Keywords: Federated analytics, distributed computing, privacy.

1 Introduction

Federated Analytics (FA) is a paradigm for collaboratively extracting insights from distributed data that is owned by multiple parties (e.g., individual mobile devices or institutional organizations) under the coordination of a central entity (e.g., a service provider) without any of the raw data leaving their local parties

*Corresponding Author: Yahya H. Ezzeldin, yessa@usc.edu

Received 11 September 2022; Revised 14 December 2022
ISSN 2048-7703; DOI 10.1561/116.00000063

or revealing information beyond the targeted insights. The core principles of this paradigm allow breaking the limitations for deriving analytics from limited centralized data, in terms of privacy concerns and operational costs. In the last decade, federated learning [62], a closely related area to federated analytics, has received significant interest both in academic and industry domains. Recently, the research community is extending federation beyond learning settings to address more generalized analytics questions. In this work, we summarize the diversity of questions within federated analytics and highlight research problems that can have significant theoretical and practical interests.

The term federated analytics was first coined by Google in 2020[1] to represent "collaborative data science without data collection". It was first explored in support of federated learning as a way for Google engineers to evaluate the quality of the learned machine learning models against real-world data. Beyond model evaluation, FA implementations have expanded to other applications with the flagship application being the discovery of popular elements across devices, e.g., popular out-of-dictionary words [97] or most popular songs recognized by phones. In these FA applications, the key challenge was to develop protocols that are efficient at scale while taking into account the limited communication bandwidth, as well as preserving the privacy of the participating parties.

Even with the success of these initial FA solutions and the recent interest in this collaborative paradigm, there is, unfortunately, no clear definition for what constitutes federated analytics, what kind of interesting analytical questions it can answer, and what are the possible real-world domains that can benefit from its applications. Very recent summarizing efforts in federated analytics have focused on queries of interest to particular domain applications such as video analytics [92]. However, there exists a wide range of other queries that can be supported (and are of interest) in an FA system. Summarizing these different query classes and the potential approaches for answering them in federated analytics provides a great starting point for new researchers in this area as well as the future development of generalized solutions for serving these queries within an FA system.

This paper aims to provide an introductory guide to federated analytics as follows (Figure 1). We first define federated analytics and how it relates to the more well-studied field of federated learning. Next, we provide a taxonomy of typical data analysis queries of interest in federated analytics and where they can find use in different domains. For the presented queries, we also discuss different existing approaches in the literature for addressing them. Finally, we discuss different challenges and opportunities within the federated analytics framework and discuss potential solutions for addressing these challenges and open directions. These open questions provide starting points for expanding and developing more practical scenarios in federated analytics, where research efforts are still needed.

[1] https://ai.googleblog.com/2020/05/federated-analytics-collaborative-data.html.

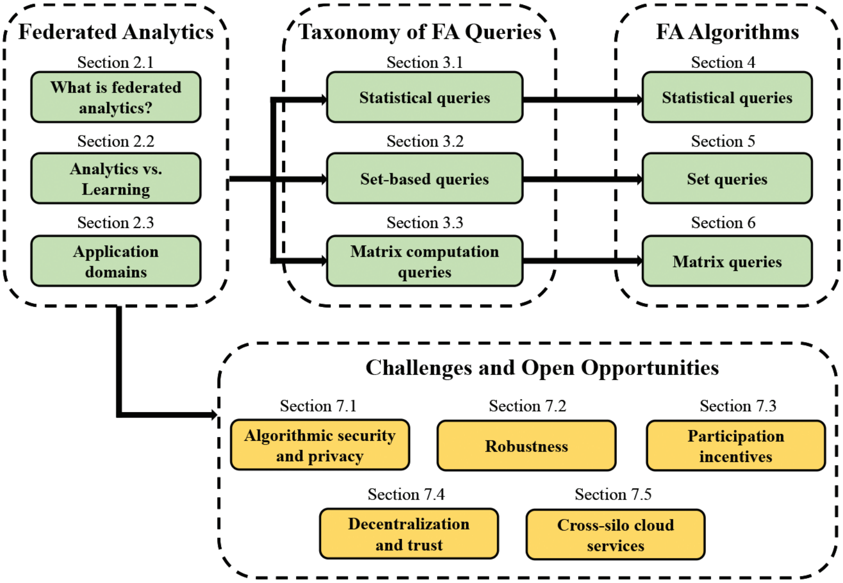

Figure 1: The schematic structure of federated analytics and the relationship between different sections. The body of this survey mainly contains the fundamentals of federated analytics, a taxonomy of different queries of federated analytics, federated analytics algorithms, applications, and discussions of challenges and opportunities in federated analytics in the presence of cloud-based services.

2 What is Federated Analytics?

In federated analytics, there is typically a central querier (the question asker) who wants to learn some property or answer a question based on data distributed across different clients (i.e., parties). Each of these clients owns a subset of the data, representing their local dataset. We will refer to these parties as clients or data owners interchangeably throughout this survey.

From a generalized perspective, **federated analytics** can be defined as a setting for data analysis where a querier wishes to answer a data analysis query through the collaboration of multiple data owners (clients) that own their local raw data. The raw data is not exchanged or transmitted, but instead, intermediate query replies that are meant for aggregation at the querier are transferred to answer the intended query.

In particular, from this generalized view, the goal of federated analytics is for a central querier to answer the following query Q

$$Q(\mathcal{D}) = F_{\omega}(\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_N). \tag{1}$$

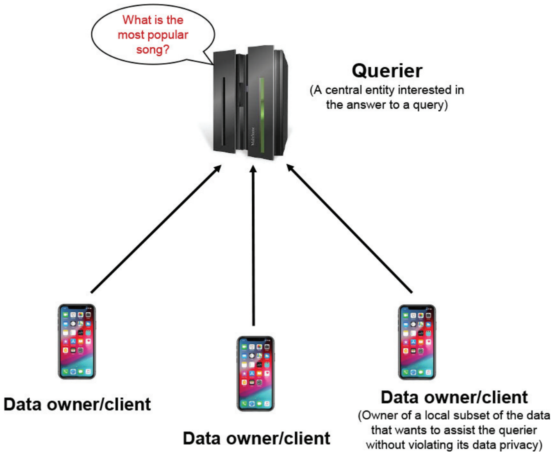
What is the most popular song?
Querier
(A central entity interested in the answer to a query)
Data owner/client
Data owner/client
Data owner/client
(Owner of a local subset of the data that wants to assist the querier without violating its data privacy)

Figure 2: An example federated analytics setting where a **querier** is discovering the most popular song in the collective datasets at the clients, where each client is a **data owner** of its local subset. To preserve the privacy of the clients' data the system seeks to answer the query distributively with only focused replies being sent back to the querier.

Here $\mathcal{D} = \{\mathcal{D}_i\}_{i=1}^m$ is the private datasets at the N data owners, and F_ω is the (potentially parameterized) function on the data describing the target query. For instance, given a pre-trained machine learning classification model parameterized by ω, the basic federated analytics query to test *the accuracy of the model* ω on the distributed datasets can be represented by the following query:

$$\begin{aligned} Q_\omega(\mathcal{D}) &= Acc(\omega; \{\mathcal{D}_1, \mathcal{D}_2, \cdots, \mathcal{D}_N\}) \\ &= \sum_{i=1}^{N} \frac{|\mathcal{D}_i|}{\sum_{i=1}^{N} |\mathcal{D}_i|} Acc(\omega; \mathcal{D}_i), \end{aligned} \tag{2}$$

with the query answer being the weighted average of each party's local test accuracy $Acc(\omega; \mathcal{D}_i)$. To compute the local accuracy, each party applies the model to its local labeled dataset and computes the local ratio of correct classifications.

2.1 Federated Learning vs. Federated Analytics

Federated analytics is very similar to federated learning [62] in the fact that both require collaborative use of distributed data without collecting the raw data at a centralized location. However, while federated learning, as a branch of distributed optimization, is about training machine learning models at the edge and aggregating learning outcomes back into the federated learning model, federated analytics is more generalized to include applying basic data science methods for data analysis but also includes optimization-based questions such as federated learning. Thus from a generalized perspective using the formulation of (1), federated learning can be viewed as a complex federated analytics query on the distributed datasets when the function F_ω is the following optimization empirical risk minimization problem:

$$F_\omega(\mathcal{D}_1, \mathcal{D}_2, \cdots, \mathcal{D}_N) = \arg\min_{\mathbf{w}} \sum_{i=1}^{N} \sum_{\mathbf{x} \in \mathcal{D}_i} \ell(\mathbf{w}; \mathbf{x}). \tag{3}$$

The analytics branch of federated learning has been extensively studied in recent years [62], while algorithms and approaches for basic data science queries have not seen similar exploration, even though they are critical to service federated learning models. In fact, one of the first application examples of non-learning queries in federated analytics is strongly coupled with federated learning, where engineers at Google wanted to evaluate the inference performance (e.g. in terms of accuracy) of trained federated learning models against real-world data not available at the data centers.

Thus, in the remainder of the paper, we limit our attention to simple federated analytics queries that would not require optimization when solved in a centralized scenario, in contrast to the federated learning branch which would require optimization of parameters to solve in a centralized setting. Following this distinction, examples of simple queries for federated analytics include questions of the form: what is the mean or median value of a function applied on the distributed data; while federated learning would be confined to learning a parameterized function such as: what is the best model that maps features $\mathbf{x}$ to target variable $\mathbf{y}$. In fact, each round of federated learning invokes the simplest question in federated analytics after local training: *what is the sum of vectors (gradient updates) stored at the participating clients?*

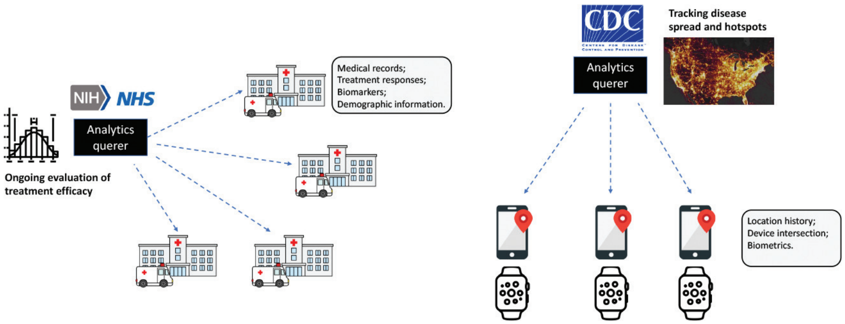
NIH
NHS
Analytics querer
Ongoing evaluation of treatment efficacy
Medical records; Treatment responses; Biomarkers; Demographic information.
CDC
Centers for Disease Control and Prevention
Tracking disease spread and hotspots
Analytics querer
Location history; Device intersection; Biometrics.

Figure 3: Examples of federated analytics applications in the healthcare domain.

2.2 Applications for Federated Analytics

We, next, discuss several canonical domains that benefit significantly from applying federated analytics. Figure 3 highlights a number of these applications of federated analytics in the healthcare domain.

- **Evaluation Analytics for Machine Learning Models.** The poster application that started garnering interest in federated analytics was the collaborative evaluation of the quality of trained machine learning models. For instance, Google uses federated analytics to evaluate the accuracy of Gboard next-word prediction models by using captured data from users' typing activities on their phones. Similar to accuracy evaluation, federated analytics can also be used to compute other evaluation metrics of the trained machine learning models, e.g., model robustness to unseen distributions/users as well as the fairness to different demographic groups [41] (for example, how different is the performance of an image tagging application to photos from the black vs white communities).

- **Analytics for Medical Studies and Precision Healthcare.** A key ingredient for realizing the full promise of precision medicine is allowing research analytics and diagnostics on large amounts of medical data that are not typically available through traditional medical research procedures. This kind of information can originate from data collected at medical institutions (e.g., the efficacy of applied treatments and onset symptoms associated with a diagnosis) to individual personal data such as location history of individuals for contact tracing (e.g. during COVID-19), or mental health studies based on bio-markers. Enabling these gains from big medical data is challenged by the legal and regulatory barriers for privacy that make collecting patient-level data outside a healthcare provider complex and time-consuming.

- **Guiding Advertisement Tactics.** Advertisers are keen to know whether their ads are attractive to their potential customers. For example, in the case of video ads, they would like to collect summary ads viewership data from users to understand the effectiveness of their advertisement concepts as well as guide future advertisement expenditure.

The aforementioned domains can make use of a large number of simple federated analytic metrics beyond the promise of federated learning models. In the following section, we give a taxonomy of different federated analytics queries and highlight to the reader some of their potential use cases in the discussed application domains.

3 A Taxonomy of Federated Analytics Queries

As described in Section 2, a federated analytics query is a general class that encompasses any question by a querer on distributed private datasets. However, from this general class of queries, there exist a number of queries that find greater exposure in different application domains and are explored more deeply in the literature. We can divide these queries of interest into three main categories: (1) Statistical testing queries, (2) Set queries, and (3) Matrix transformation queries. The statistical testing category includes different data science queries that aim to discover key statistical properties of the distributed private data. Examples of such queries would be the estimation of the mean median, heavy hitters, key-valued data frequencies, hypothesis testing, ..., etc. The set queries, on the other hand, include analytics for discovering data associations such as set intersection, set union, and intersection cardinality. Matrix transformation queries include but are not limited to operations such as dimensionality reduction using methods such as principal component analysis, and projections. In this section, we formally define the most popular queries in each of the aforementioned query types and present some of their real-world applications. Figure 4 summarizes the queries presented in the remainder of this section.

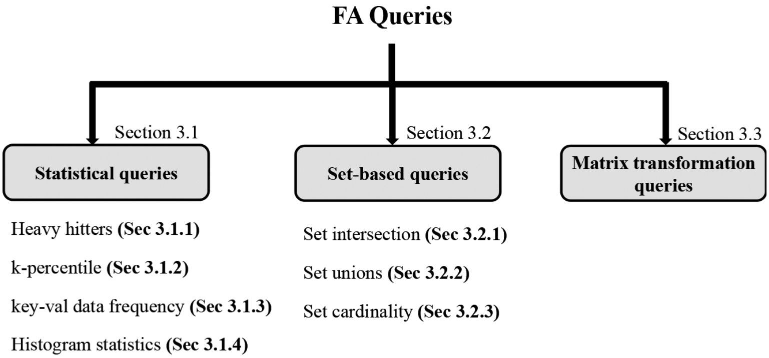
FA Queries
Section 3.1
Section 3.2
Section 3.3
Statistical queries
Set-based queries
Matrix transformation queries
Heavy hitters (Sec 3.1.1)
k-percentile (Sec 3.1.2)
key-val data frequency (Sec 3.1.3)
Histogram statistics (Sec 3.1.4)
Set intersection (Sec 3.2.1)
Set unions (Sec 3.2.2)
Set cardinality (Sec 3.2.3)

Figure 4: A taxonomy of federated analytics queries presented in Section 3.

3.1 Statistical Testing

We focus on four key statistical queries that have a wide variety of real-world applications in different domains, such as health, business, and user experience. For each of these statistical queries, we give its mathematical definition, followed by one of its main applications. We discuss some existing solutions in Section 4. We start by first assuming having a set $\mathcal{D} = \{\mathcal{D}_1, \ldots, \mathcal{D}_N\}$ of N datasets, where each dataset $\mathcal{D}_i = \{x_1^i, \ldots, x_{n_i}^i\}$ consists of n_i data points and is owned solely by one distributed node, i.e., an FA client.

3.1.1 Heavy Hitters

The objective of the heavy hitter problem is to construct a succinct histogram of the elements across the N parties datasets that contains only the most popular (heavy-hitter) elements; other elements are treated as if appearing with zero frequency. Typically, an element is denoted a heavy-hitter if its frequency in the distributed dataset is greater than or equal to a fraction ϕ of the dataset size. Formally the goal of the query is to return the following:

$$\begin{aligned} Q(\mathcal{D}) &= \{(x, \mathit{freq}(x)) | x \in \mathcal{D}_{\mathrm{HH}}\} \\ \text{where:} \quad \mathcal{D}_{\mathrm{HH}} &= \left\{ x \,\middle|\, x \in \bigcup_{i=1}^{N} \mathcal{D}_i, \quad \mathit{freq}(x) \geq \phi |\mathcal{D}| \right\}. \end{aligned} \tag{4}$$

Note that the heavy-hitters problem is closely related to another succinct histogram problem formulation, the top-K problem, where the goal is to find a succinct histogram with the K most frequent elements instead of all elements exceeding a threshold. If we target the top-1, this translates to the well-known *mode* statistic of the dataset.

Application (User Experience). One popular application of heavy hitters is to learn trendy out-of-dictionary words generated by users' devices. Learning trendy words is of high interest to service providers as it allows them to improve the service they provide to their users. These services could be the autocomplete feature in smart keyboards, or a powerful advertisement engine that could leverage the current public taste of people for more effective advertisement. A similar application is to learn the out-of-dictionary words, which can be used to improve the smart keyboard spell-auto-correction feature by adding such words to the keyboard's dictionary. Apple has already used differential privacy to protect the privacy of users' input data while collecting the top frequent emojis by users [8]. Similarly, Google has also proposed another differential privacy (DP) method to collect the out-of-dictionary words [70].

3.1.2 k-percentile Element

In the k-th percentile statistical query problem, the objective is to find the smallest element that is greater than k percent of the overall dataset available at the participating distributed nodes. This statistical query problem can be formalized as follows. Assuming the entries of the datasets in $\mathcal{D}$ are non-categorical values (i.e., numerical values), then by denoting $\mathcal{D}^s$ to be the non-decreasing sorted set of the elements of $\bigcup_{i=1}^{N} \mathcal{D}_i$, the k-percentile element x_k in this distributed parties datasets $\mathcal{D} = \{\mathcal{D}_1, \mathcal{D}_2, \cdots, \mathcal{D}_N\}$ is given by

$$Q(\mathcal{D}) = x_k = x \quad \text{such that} \quad \mathit{rank}_{\mathcal{D}^s}(x) = k \times |\mathcal{D}^s|, \tag{5}$$

where $rank^s_{\mathcal{D}}(x)$ is the order of element x in the dataset $\mathcal{D}^s$. An example of k-percentile values is the *median*, where k is 0.5.

Application (Business). It is well-known that the median is a more robust metric to represent central tendency compared to the mean, which is more sensitive to outliers. Hence, it is more useful in business use cases to assess different components such as company salaries. For instance, a possible application for federated median computation is for an authority to compute the median salary (or any other percentile) of all employees in a set of companies without revealing the exact salaries of the employees or which companies they belong to.

3.1.3 Key-valued Data

The Key-valued data is a statistical query problem in which each data point is represented by a key (e.g., identifier) and value associated with this key, while the objective is to learn the frequency of each key and the mean (or aggregate) of the values that appear paired with this particular key. To formalize the objective, we assume that the dataset $\mathcal{D}_i$, for $i \in [N]$ is a key-valued dataset such that $\mathcal{D}_i = \left\{x^i_j \middle| x^i_j = (k^i_j, v^i_j),\ \forall j \in [n_i]\right\}$. The objective is to find the following

$$Q(\mathcal{D}) = \left\{ \left(\mathit{freq}(k_i),\ \frac{1}{|\mathit{freq}(k_i)|} \sum_{v_j : (k_i, v_j) \in \mathcal{D}} v_j \right), \forall k_i \in \mathcal{D} \right\}. \tag{6}$$

Application (Business). A possible application can be in the business market, where the objective is to privately learn the distribution of the stocks and the investment amount of each stock from the private data of the investors. Specifically, in this stock market application, the key represents the stocks while the value represents the amount that a person invests in a given stock. The statistical query goal takes place when an analyst wants to learn how many agents invest in each stock (e.g., frequency distribution stocks) and the amount invested in each stock (e.g., average or aggregate amount) without collecting any private data which can cause a breach to their privacy.

3.1.4 Histogram-Based Statistics

This can be considered a special case of the key-valued data problem, where the objective is to learn only the frequency of each key.

Application (User Experience). One real-world application of histogram-based statistics is the Now Playing feature on Google's Pixel phones [48]. This feature uses an on-device database of song fingerprints to show users what song is playing in the surrounding room without an internet connection. The one-device database includes the most frequently recognized songs, which are maintained and updated by Google to ensure that the database contains only popular songs. The way it works is that on each phone, the Now Playing application computes the recognition rate (value) for each song (key) in its Now Playing History. Once the phone is plugged in and connected to WiFi, the users encrypt the rate of the songs and send them to the Google servers so that they can only compute a histogram distribution of all song counts. This allows Google to replace the less popular songs in the database with the more popular ones.

3.2 Private Set Queries

The distributed private set queries class can be broadly clustered into three different categories; distributed sets intersection, distributed sets union, and distributed cardinality computation. The main goal of this analytic problem is to compute these queries in a way that protects the privacy of the data owners being queried. Similar to the statistical testing class, we consider having N parties where each party i has a dataset $\mathcal{D}_i$ of n_i unique and private data points. Some of the existing solutions to set queries are presented in Section 5.

3.2.1 Private Set Intersection

The private set intersection (PSI) is a private set query problem that has a wide range of applications with the objective of computing the intersection between the sets owned by the different clients and nothing beyond that. This query is formally given as follows

$$Q(\mathcal{D}) = \bigcap_{i=1}^{N} \mathcal{D}_i. \tag{7}$$

Application (Business). One famous application of PSI in the two-party setting is the online-to-offline advertisement conversion [56] in which a company would like to know how much of its revenue can be attributed to an online advertisement in order to assess the future payment it spends on a paid ad (e.g., Facebook ad). On the other hand, the advertising company wants to know how successful its advertising campaign is. In this setting, the advertising companies have a database of the users and their status, whether they saw the ad or not, while the company knows the users who purchased their products as well as the amount they spent on their purchases. In other words, the data needed to compute these statistics are split across the two parties. In this setting, the two parties are typically unwilling to share their customers' data to protect the privacy of their business and their customers, but both parties would want to collaboratively learn how many users both saw an ad and made a corresponding purchase, as well as the amount of money those users spent on the company's products.

3.2.2 Union

Similar to private set intersection, the goal is to privately evaluate the union of the input sets of two or more parties privately without revealing anything about the sets beyond the union. This objective can be formally given by

$$Q(\mathcal{D}) = \bigcup_{i=1}^{N} \mathcal{D}_i. \tag{8}$$

Application (Security). One popular application is risk assessment and management [83]. The goal of this application is to aggregate the blacklists from different parties and across various attack types. This could help in improving the individual blacklists in identifying malicious sources.

3.2.3 Cardinality

The goal of this problem is to learn the cardinality of the intersection of the data set of multiple parties in a private manner, which can formally be given as follows

$$Q(\mathcal{D}) = \left| \bigcap_{i=1}^{N} \mathcal{D}_i \right|. \tag{9}$$

Application (Public Safety) One popular real-world application of PSI cardinality is the CSAM Detection system used by apple "Apple for Child Sexual Abuse Material (CSAM)". The main goal is to identify and report iCloud users who store known Child Sexual Abuse Material (CSAM) in their iCloud Photos accounts. The way it works is that intersection cardinality testing is carried on between a known database of CSAM images and individual iCloud users. When the cordiality of intersection exceeds a predefined threshold, Apple can provide relevant information to the National Center for Missing and Exploited Children (NCMEC).

3.3 Matrix Transformations

Singular value decomposition (SVD) is one of the most popular matrix operations that have a wide range of applications in either data analytics or machine learning. The main objective of this problem is to compute SVD over a set of distributed data without collecting any raw data or breaching the privacy of the data owners. This problem can be formally defined as follows: assume there are n parties, and each party i has a private data matrix $\mathbf{D}_i \in \mathrm{R}^{m \times n_i}$. The n parties would like to compute the SVD jointly on the combined dataset $\mathbf{D} = [\mathbf{D}_1, \ldots, \mathbf{D}_n]$, where $\mathbf{D} \in \mathrm{R}^{m \times n}$ and $n = \sum_{i=1}^{n} n_i$. The private computation of SVD on the combined dataset takes the following form

$$Q(\mathcal{D}) = \mathbf{U}\Sigma[\mathbf{v}_1^T, \ldots, \mathbf{v}_n^T] \tag{10}$$

where $\mathbf{U}$ and Σ are shared across all the parties, while $\mathbf{V}_i$, $\forall i \in [n]$, is kept secret by party i and never shared with any other parties. From (10), each node i can get its SVD by using the shared matrices $\mathbf{U}$ and Σ, and the secret matrix $\mathbf{V}_i$ as $\mathbf{D}_i = \mathbf{U}\Sigma\mathbf{V}_i$.

Another variant of SVD called Funk-SVD is applied to the sparse rating matrix used in the recommendation systems [23] such that it composes the sparse matrix into two embedding matrices that can be used to predict the missing rating in the rating matrix.

Application (Machine Learning). SVD is an essential building block in many studies and applications, such as principal component analysis (PCA). PCA is used to reduce the feature space of the data used in machine learning. Reducing dimensionality in statistical machine learning can prevent the model from overfitting, which reduces the ability of the model to generalize beyond the examples in the training set. One challenge of performing PCA in a distributed setting is having the data distributed across multiple nodes while collecting and gathering the data is prevented by the law (e.g., GDPR [90]). We discuss some existing solutions for the matrix transformation query in Section 4.

4 Existing Solutions to Statistical Testing Queries

A taxonomy of the privacy-preserving techniques used for the statistical testing queries is given in Table 1. We consider different variants of privacy-preserving techniques represented by differential privacy (DP), secure multi-party computing (MPC), and a combination of DP with MPC.

4.1 Heavy Hitters

The heavy hitter problem has been well studied in the literature either in the centralized setting with no privacy requirements where the data is already collected and stored at a central server or in a distributed federated setting where the queerer wishes to learn the “heavy hitters” in the clients’ data while guaranteeing the privacy of each contributing client at minimal computation/communication costs [4, 5, 8, 10, 11, 25, 27, 42, 51, 97]).

Table 1: Taxonomy of the privacy-preserving techniques used in the statistical query.

Query	Privacy technique	Related works	Noisy response
Heavy hitters	Non-private	[25, 27]	No
	DP	[4, 5, 8, 10, 11, 51, 97]	Yes
	MPC	[20]	No
	DP + MPC	[17]	Yes
Median	Non-private	[57]	No
	DP	[16, 18]	Yes
	MPC	[6, 47, 89]	No
Key-valued data	DP	[49, 93]	Yes

4.1.1 Non-private Centralized Setting

In the non-private centralized setting, the main objective is to develop efficient heavy hitters algorithms with low storage requirements and provable error bound. The low storage requirement is of significant importance when dealing with a large online data stream that memory-intensive solutions such as sorting the stream or keeping a counter for each distinct element are infeasible (e.g., [25, 27]). [25] proposes an approximate heavy hitter algorithm that is memory efficient with proven theoretical error bound. The algorithm is based on sketch counting that relies on using a set of hashes that map each element in the data stream to different bins, such that when running the sketch counting algorithm along with a max-heap data structure, the algorithm can find the k heavy hitters in a stream of d unique items with storage cost logarithmic in d (e.g., $O(K \log d)$) instead of being linear in d.

4.1.2 Private Distributed Setting

There is a rich body of works on private heavy hitters and frequency estimation in the distributed setting while ensuring users' privacy by leveraging DP [4, 5, 8, 10, 11, 51, 97], MPC [20], or combine DP with MPC [17].

Heavy Hitters with Differential Privacy. Researchers have proposed multiple *efficient* private heavy hitter algorithms that have a computation time, communication cost, and storage cost polynomial in n (number of users) and logarithmic in d, $log(d)$, where d is the size of the data universe (dictionary of the data points to check). [51] proposed several efficient (ϵ, δ)-differentially private algorithms for the heavy hitter problem for n parties, each of which possesses a single element from a universe of size d. However, their algorithms experience high error between the estimated frequency for the heavy hitter items and their true frequency, where the error rate is given by $\mathcal{O}\sqrt[6]{\frac{log(d)log(\frac{1}{\delta})}{\epsilon^2 n}}$, which does not match their error lower bound $\Omega(\frac{1}{\sqrt{n}})$. In contrast to [51], Bassily and Smith [11] provide the first polynomial time local $(\epsilon, 0)$-differentially private protocol for heavy hitters that has worst-case error $\mathcal{O}(\sqrt{\frac{log(d)}{\epsilon^2 n}})$. They also show that using the public coin model, each user can send only one bit to the server. However, one of the main limitations of their approach is the high time complexity, where their algorithm requires a server running time of $O(n^{5/2})$ and a user running time of $O(n^{3/2})$.

In later work, Bassily *et al.* [10] have proposed two algorithms, TreeHist and Bitstogram, which require a server running time of $\mathcal{O}(n)$ and a user running time of $\mathcal{O}(1)$. The TreeHist algorithm is based on a noisy, compressed version of the count sketch proposed in [25]. From the practical point of view, in a concurrent work [8], Apple has proposed the Sequence Fragment Puzzle (SFP) algorithm, a state-of-the-art sketching-based algorithm for discovering heavy hitters using local DP and an unknown dictionary. In this work, they have proven expressions for balancing the trade-offs among privacy, accuracy, transmission cost, and computation cost, allowing a trade-off of these parameters in different practical use cases. There are some other works (e.g., [42]) that propose a heuristic algorithm that can be used for finding the heavy hitter with an unknown dictionary. While the work in [10] requires public randomness and coordination between the server and users, the authors in [5] have proposed an algorithm based on Hadamard Response (HR) that is used in general for frequency estimation and does not require any public randomness, but at the cost of a per-user communication cost of $\log(d)$, while working for all privacy regime (e.g., $\forall\epsilon$). In contrast to [5] that trades the need for public randomness with more per-user communication cost, [4] proposes an algorithm that requires only 1-bit per user while not requiring any public randomness. However, their algorithm gives an optimal error rate only at the high privacy regime, i.e., $\epsilon < 1$.

The previously mentioned works utilize local DP to ensure privacy, yet it is known that local DP often leads to a significant reduction in utility [35, 61, 63]. On the other hand, the choice of using central DP requires having a trusted server that can first collect the clean data and then perturbs it. Since in the central DP setting, noise is only applied once by a trusted server, central DP has better utility than local DP. To overcome the limitations of central DP and local DP, [97] propose trie-based heavy hitters (TrieHH) algorithm that is interactive (e.g., multi-round algorithm) and leverages its interactivity to achieve central DP without the need to centralize raw data while also avoiding the significant loss in utility incurred by local differential privacy. The DP privacy guarantee of their algorithm is achieved by leveraging the randomness from the user sampling and the anonymity properties of their distributed algorithm, which make their algorithm inherently differentially private without requiring additional noise. This is different from the previously discussed works that are non-interactive and achieve local DP using the randomized response. It is also different from the work in [10] that relies on public randomness. They have also studied the trade-off between privacy and utility and shown that their algorithm can achieve good utility while ensuring strong privacy guarantees, compared with the works that rely on DP, such as [8].

Secure Multi-party Computing. Leveraging secure multiparty computing primitives is another direction for privately computing the heavy hitters without impacting the utility [20] or requiring a large number of users as in [97] to get reasonable utility. The proposed protocol by Boneh *et al.* [20] for solving the private heavy-hitter problem leverages a lightweight cryptographic tool called incremental distributed point functions instead of using DP, which could reduce the utility. The proposed protocol relies on the assumption of having two non-colluding servers, which is one of the main limitations of this work. Additionally, it requires at least one of the two servers to not collude with any client. Apart from these limitations, this protocol can guarantee correctness in the presence of malicious clients who can manipulate its input string to alter the protocol execution. The proposed protocol is interactive, requiring all users to participate only once in the protocol execution, where each client can send only a single message of size linear in the length of the input string to the servers. Similar to most works that utilize DP, the proposed protocol requires any public-key cryptographic operations except for establishing secret channels between the parties.

Secure Multi-party Computing with DP. By combining MPC and DP, Böhler and Kerschbaum [17] have proposed a heavy hitters protocol that provides high utility even for a small number of users, which is the most challenging regime for DP [97]. The proposed algorithm, in contrast to [20], considers the existence of only one server that wishes to compute the K-heavy hitters on the input strings of the clients.

4.2 Median

Similar to the heavy hitter problem, the works for distributed median computation are also broadly classified from the perspective of privacy into works that leverage MPC primitives and DP.

Secure Multi-party Computing. As pointed out by [6], the problem of private computing of the k-th ranked element on the private dataset of several parties can be solved by constructing a combinatorial circuit that is evaluated securely by the parties (e.g., [47]). However, the main limitation of these generic protocols is the communication overhead. In particular, for a two-party setting, where the combined data set size is n, and the elements of the dataset are drawn from a field of size M, the communication cost of this circuit-based solution is $\Omega(n \log M)$. For applications where the data size is large, these generic solutions are impractical. By using an interactive protocol that relies on the binary search and secure comparison using Yao's garbled circuit, Aggarwal *et al.* [6] have provided the first specialized protocols for computing the k-th ranked element with sublinear communication and computation overhead for the two-party setting and the multi-party setting where parties in both settings are interested in knowing the k-th ranked element. In the two-party case, the cost of computing the k-th ranked element is $O(\log M \cdot \log k)$ compared to $O(\log^2 M)$ in the multi-party setting. The number of rounds of the proposed algorithm for the two-party is logarithmic in the number of input items, whereas the number of rounds of the multi-party algorithm is logarithmic in the size of the domain of possible input values (e.g., $\log M$). The proposed protocol provides security against malicious parties. One of the main limitations of this work for the multi-party setting is that it requires lots of coordination between all pairs of parties for establishing pairwise communication channels, thus impacting its practicality. Another practical limitation is that it is very interactive, where the number of rounds to complete the protocol scales logarithmic with the field size. To overcome such limitations, Tueno *et al.* [89] have proposed efficient algorithms that leverage the client-server architecture. In this client-server setting, there are communication channels only between each client and the server, while only clients provide inputs to the computation. The rule of the server in this setting is to make their computational resources available for the computation but have no input to the computation and receive no output. By using this setting, their proposed algorithm is less interactive, as it only requires a fixed number of rounds with the server (e.g., at most four rounds) compared to $O(\log^2 M)$ for the algorithm in [6]. The highest computation cost of their algorithms is $O(\log^2 M)$.

Differential Privacy. Computing the exact median value and revealing it to the clients using the algorithms proposed by [6, 47, 89] can violate the privacy of the parties that own this median value. To overcome such a challenge, [16] proposes an efficient algorithm for computing a differential private median between two parties by utilizing the exponential mechanism. The proposed algorithm has a computation complexity sublinear in the size of the data universe (e.g., $\log M$). Böhler and Kerschbaum [18] proposed another algorithm for private median computation in the multi-party setting while using the exponential mechanism. Their algorithm for the multi-party setting also has a computation complexity sublinear in the data size. The threat model considered in this setting is the semi-honest (non-malicious) clients. They also discuss how to extend their algorithm to malicious clients, and implement it using the SCALE-MAMBA framework [7].

Non-private. From the distributed optimization perspective, Iutzeler [57] has proposed distributed synchronous and asynchronous algorithms for computing median and other elements of specified ranks of the clients' data. Unlike the works in [6, 16, 18] that connect all nodes as a fully connected graph, this work considers a general undirected connected graph. To distributedly solve the median problem, they first design a convex optimization problem whose solution meets the median or the quantile to compute. They solve the problem using the distributed formulation of ADMM proposed by [21, 72].

4.3 Key-Valued Data

The objective of this problem is to collect two fundamental statistics of key-value pairs, including frequency of keys and mean of values. One naive solution is to apply local DP independently at the keys and values. Since keys are categorical data, some existing DP methods (e.g., [40, 63]) can be applied to each key, while each value can be perturbed using (e.g., [36, 76]). However, the main challenge for this naive approach of applying local DP is to achieve a good utility-privacy trade-off, since the data contains two dimensions, and a user may have multiple key-value pairs. Additionally, this naïve independent perturbation does not preserve the correlation between the keys and values. To address this challenge, Ye *et al.* [93] proposed the first specialized LDP algorithms for this problem by modifying the Harmony randomized response-based protocol [76] to better maintain the relationships between the keys and values to improve the accuracy of statistics while still achieving local differential privacy. Their first proposed algorithm, PrivKV, is a non-iterative (non-interactive) algorithm that is suitable for low communication cost scenarios. Additionally, they have proposed another two interactive protocols (PrivKVM and PrivKVM+) to iteratively improve the estimation of a key's mean value PrivKVM trades the communication cost with the accuracy while PrivKVM+ balances between accuracy and communication bandwidth. The main limitation of their non-interactive algorithms is the large number of rounds required to get an unbiased mean estimation and to improve the estimation of a key's mean value. In general, their key limitations, which have also been highlighted by [49] include (1) A large number of rounds requires all users to be always online, thus limiting its practicality. (2) The privacy budget increases with the number of rounds. For a fixed privacy budget, the budget for each round decreases as the number of rounds increases. This decrease in per-round privacy budget increases the amount of noise added, which can negatively impact performance. (3) Their privacy analysis lacks improved budget composition for local differential privacy that can capture the correlation between key and value given by their algorithms. (4) Finally, their proposed random key sampling method, which is part of their algorithms, does not work well for a large key domain. Follow-up work by Gu *et al.* [49]

introduced a non-interactive framework called PCKV with a better utility-privacy trade-off that overcomes the aforementioned limitations. In particular, they apply an advanced sampling procedure to enhance utility over the naive random sampling done by PrivKVM. They also require only a single iteration and provide a tighter analysis of the privacy budget consumption.

5 Existing Solutions to Set Queries

Private set intersection/union computations have had a number of practical use cases that is large enough to garner the attention of researchers over the last two decades [80]. Below, we discuss a number of key approaches to solving these set query problems, mainly from the MPC community. A taxonomy of the privacy-preserving techniques used for these set queries is given in Table 2.

Table 2: Taxonomy of the privacy-preserving techniques used in the set queries.

Query	Privacy technique	Related works
Private set intersection	Homomorphic encryption	[26, 33, 43, 50, 53, 55, 75]
	Oblivious polynomial evaluation	[31, 44]
	Oblivious transfer	[68, 77–79, 84]
	Garbled circuit	[34, 52, 54]
Private set union	Homomorphic encryption	[45, 67]
	Oblivious polynomial evaluation	[59, 69]
Private cardinality testing	Homomorphic encryption	[9, 46]
	Oblivious transfer	[22]

5.1 Private Set Intersection

The existing approaches for the two-party setting include works based on homomorphic encryption (HE) [26, 33, 43, 53, 55, 75], works based on Oblivious Polynomial Evaluation [31, 44], works based on Oblivious Transfer [77–79, 84], and works based on garbled circuit [34, 52]. Although these techniques are for the two-party setting, some of them were extended to the multi-party setting. Specifically, Kolesnikov *et al.* [68] have proposed oblivious programmable pseudo-random functions that are based on the idea of using oblivious transfer. Garbled bloom filter has been used in [54], and HE has been used in [50].

5.2 Private Set Union

Kissner and Song [67] have proposed the first protocol for the private set union, which leverages threshold additively HE and polynomial representation. Another approach [45] that adopts a similar technique can reduce the communication/computation complexity of [67]. Instead of using polynomial representation, [32] uses an inverted Bloom Filter. While the above works use public key operations, which result in increasing their computation complexities, Kolesnikov *et al.* [69] proposed the first scalable PSU protocol using only symmetric-key techniques while using polynomial representation for computing the private set unions. However, their protocol requires repeated high-degree polynomial interpolations on the parties' datasets. To overcome such limitation, Jia *et al.* [59] proposed an algorithm that relies on using data shuffling and avoids using HE and repeated operations.

5.3 Private Cardinality Testing

The problem of cardinality testing has been considered in the two-party setting [14, 46], and in different works for the multi-party setting [9, 22] where these different works have developed efficient solutions in terms of the computation and communication costs while preserving the privacy of the users' data.

6 Existing Solutions to Matrix Transformation

To solve the problem in (10), Chai *et al.* [24] proposed an efficient lossless federated SVD solution over billion-scale data called FedSVD ensures the accuracy of the SVD computation is not impacted. This is guaranteed by avoiding using DP methods; instead, they rely on masking their data in a way such that the masks are canceled out when the response from the different parties is aggregated by the server. Thus, this approach guarantees the same performance as the centralized case where all the data are located in one place. Liu and Tang [73] have proposed an algorithm that uses additive HE. On the other hand, Chai *et al.* [13] and Berlioz *et al.* [23] have proposed distributed privacy-preserving algorithms for recommendation systems that rely on matrix factorization. The proposed algorithm by Chai *et al.* [23] is based on HE, while the one proposed by Berlioz *et al.* [13] leverages differential privacy. The taxonomy of the privacy-preserving techniques used for the set queries is summarized in Table 3.

Table 3: Taxonomy of the privacy-preserving techniques for matrix transformation.

Query	Privacy technique	Related works
Matrix factorization	Homomorphic encryption	[23, 73]
	MPC	[24]
	DP	[13]

7 Challenges and Open Opportunities

7.1 Algorithmic Security and Privacy

In the previous Sections 4–6, we presented a number of privacy-preserving approaches to compute the FA queries. However, unlike FL, there does not exist a single common framework or algorithm for privately computing a diverse number of queries. A unifying approach to evaluate FA queries without leaking unnecessary information is an open question of great importance for deploying FA systems, as it will allow them the flexibility to deal with a wide range of queries. Note that if the target is to solve the query while disregarding privacy, then a number of queries discussed earlier can be computed and then used to derive answers for other queries. For example, the mode, mean and median statistical queries can all be computed by first computing the FA histogram query and then deriving the target answers (e.g. median) from it. This, however, leaks unnecessary information to the querer beyond the intended goal.

One solution to address this information leakage is to employ secure enclaves [29] at the querer to isolate a code execution and memory in a trusted environment where the code can be attested and verified while keeping its state a secret until it publishes an output. Using this in our previous example, the querer can run a code to aggregate the histogram and then extract the required target query from it. Although secure enclaves can theoretically address the security challenges arising from using a non-specialized analytics algorithm, current secure enclave models are only limited to CPU resources and provide limited memory resources, which limits their potential universal deployment.

With these limitations, it remains an open problem when and how much to make use of these trusted secure enclaves in the logic for computing the target query, and whether there exists a universal approach to securely and privately computes federated analytics queries that does not need to use secure enclaves.

7.2 Robustness to System Failures

The quality of computed analytics in a federated analytics system can be prone to performance degradation due to a number of malicious or non-malicious system failures. Malicious failures can arise due to attempts by some system parties to alter their data or responses in order to either degrade the system performance or targets its deviation towards a premeditated result. In addition to malicious failures, the distributed nature of federated analytics and its reliance on parties that are not co-owned can cause it to suffer from party dropout or straggling which can potentially happen during the execution of the federated analytics algorithm. The use of privacy-preserving mechanisms in federated analytics such as secure aggregation [19] as well as other MPC protocols, can hinder the detection or recovery from these malicious or non-malicious faults. How to make federated analytics robust to such failures without giving up any or little privacy is an interesting open problem in the area.

Although a universal solution for robustness in federated analytics is still open, there exist some approaches for handling failures in federated learning that can lend themselves easily to the federated analytics framework. The non-malicious failure of clients was an overarching limitation of the vanilla secure aggregation protocol [19]. While the protocol design was inherently able to recover from these failures and compute the sum (mean) from the surviving clients, a huge recovery cost is incurred that is can grow quadratically with the number of clients. Recent advances [60, 87] have proposed more efficient approaches for designing secure aggregation keys that allow for a more efficient recovery. These techniques lend themselves to algorithms that rely on aggregating from all clients simultaneously. Some federated queries, however, require structured responses where a particular subset of clients need to be active in each round. In this case, recovering the aggregate response from the surviving clients may be useless in some cases, and more sophisticated secure aggregation protocols are in great need. For example, one simple method would be checking if the subset of surviving clients does not satisfy particular properties, and if so, abandon the aggregation over this subset of clients in this round.

For malicious failures that try to poison a client's dataset, data sanitization [30, 88] and anomaly-detection [15] techniques, which aim to detect or remove anomalous data, have typically been used to address this. However, these techniques typically rely on access to some subset of the clients' data at the server or the availability of data that is sampled from the same distribution, which makes them incompatible with privacy-preserving approaches employed in federated analytics. It remains an open problem whether we can use these failure mitigation techniques in federated analytics without giving up privacy or if new defense approaches need to be developed to address malicious failures in federated analytics.

7.3 Participation Incentive Mechanisms

In parallel to the development of efficient and secure approaches for federated analytics, developing appropriate mechanisms to incentivize participation is a critical open question for federated analytics systems. This is particularly important in scenarios where the data owners are competitive entities such as financial institutions or enterprises, where the default strategy is not to collaborate with other competitors. Forms of incentive in the cross-silo setting can be regulatory by a governing entity (for example, the FDIC wants to detect fraudulent activity across different banks [38]), or for shared operational stability, by jointly computing the salary quantiles across a cohort of companies [65]. In the case of cross-device (individual) clients, incentives can include provided services, and/or monetary gain. From a service perspective, federated analytics promises users potential improvement in the quality of their service experience, e.g., a higher accuracy word predictor in Gboard or better estimation of travel times in navigation applications. In other scenarios, the incentive can be individual welfare, similar to the contact tracing analytics performed using private set intersections during the COVID-19 pandemic.

In either cross-silo or cross-device, a central challenge is balancing incentive with the heterogeneity of data and contribution (e.g., in terms of the data size). To address this, careful design should be taken into account to ensure clients with more data are not discouraged due to the non-proportionality of the incentives to their contributions, as well as, not pushing away clients with less data by not implementing worthwhile incentives.

7.4 Decentralized and Trust

Our discussions so far always considered a central querier that poses intermediate questions to the clients and aggregates their responses in order to arrive at the query answer (this can be in one-shot or iteratively). Such a model makes sense for queries where the question implies an authoritative entity (for fraud detection for instance) or a large company (for product analytics) is asking the query. However, for a population of clients that wish to collaboratively learn a property of their joint dataset, handling the query computation distributively can be more desirable. The key idea of decentralized analytics is to rely on peer-to-peer communications between the clients to answer the query, while still maintaining the privacy and security of exchanged information about the local datasets. Computing decentralized analytics can find application in scenarios such as the evaluation of trained models that are stored on the blockchain [86] or to crowd-source the computation of percentiles (e.g., median) of employee salaries of the technology sectors without the pre-requisite of having the parent companies agree to perform this federated computation.

There has been a wide array of works in MPC that develop decentralized solutions for secure computation, particularly for private set intersection problems (see Section 5.1). However, such solutions assume that the communication graph of clients is fully-connected and undirected. This can lead to inefficient protocols, particularly as the number of parties increases. Furthermore, sparse and directed communication graphs can model more diverse scenarios, for instance, when the clients are not co-located or when communication goes in a single direction (e.g., due to different social network connection tiers).

An interesting aspect of decentralized federated analytics is its decreased robustness to system failures (see the discussion in Section 7.2) due to the absence of a centralized entity that can potentially filter out malicious contributions or recover the system in the case of party drops. The design of incentive mechanisms for participation in a decentralized scenario is also a critical open research direction, as coordinating incentives is also impacted by the absence of a central coordinator.

One recent promising approach to address decentralized analytics challenges is to use blockchains to keep track of intermediate updates and verify that intermediate clients in the communication graph do not act maliciously during the aggregation of updates. The Biscotti framework [86] in the context of federated learning can be easily extended to mechanisms that rely on iterative updates and secure aggregation. In Biscotti, the blockchain ledger uses verifiable random functions to ensure that the aggregation contributed by a user is truly the resultant of the stored encoded intermediate updates. It also uses DP to ensure the privacy of these stored encodings. An adaptation of a blockchain solution for decentralized federated analytics can lead to more flexible algorithms that are crowd-operated without the requirement to trust a centralized aggregator/querier entity.

7.5 Cross-silo Federated Analytics on the Cloud

In previous sections, we assume that FA clients own their data and process the data in local and trusted environments when responding to a query. However, in real-world deployments, instead of maintaining local data centers and keeping the data on the local side, FA clients typically would use third-party public cloud services such as Microsoft Azure, Google Cloud, Amazon Web Services, IBM Cloud, and Alibaba Cloud, to store and process their data. Outsourcing data to such third-party clouds has emerged as the de facto model for data storage and processing for numerous benefits, such as improved availability, lower cost, and improved service.

Using clouds in an FA system, however, poses additional security and privacy challenges due to the untrusted nature of public clouds. A public cloud may be curious and wish to learn some information about the data of the FA clients. To protect data from such adversarial clouds, FA clients can use two classes of solutions for secure data outsourcing. The first is called *single cloud-based solutions*, in which clients encrypts their data and use a single cloud to store the data. The second is called *multi-cloud-based solutions*, in which a client partitions their data into several parts, e.g., secret-sharing shares, and stores those parts in different clouds so that no single cloud can get the complete data.

In the following subsections, we will discuss solutions in the literature that address security and privacy challenges in the two aforementioned outsourcing settings. We will use the set intersection query as the running example throughout our discussions, since it is difficult, complex, and important in query processing, and most existing works focus on this type of query.

7.5.1 Single Cloud-Based Solutions

In single cloud-based solutions, clients encrypt their local data and use a single cloud to store the data. [1–3, 64, 66, 74, 82, 94] allow clients to outsource their private datasets and process Private set Intersection (PSI) tasks without downloading the datasets. [3] ensures that the cloud can only compute set intersection after obtaining the permission of all the clients, and the computation results will be protected from the cloud. [2, 64] provided a watermark-based verification approach for queries over outsourced encrypted datasets. [2] can also detect malicious cloud (i.e., an adversarial cloud that may tamper the data stored on it) by inserting secret values in the real datasets to the cloud each time to process a PSI query. By checking whether the result set contains the secret values, the clients will know whether the query result is correct or not. [66] shares secrets between the cloud and the clients to pre-process datasets when outsourcing the datasets. This approach is collusion-resistant if one client and the public cloud collude. However, it requires a client to encrypt the datasets with different encryption keys for set intersections with different clients. [74] delegates PSI computation over randomized datasets to a cloud. Each client computes the hash value of its dataset using a general-purpose hash function, then randomizes each hashed data with a random integer. [82] applied fine-grained authorization that enables the cloud to perform queries without leaking any data. When a client A asks for a matching request with another client B, A first negotiates a token with B so that A can delegate the computation over the outsourced encrypted datasets to the cloud server, and such operations require a trusted third party to generate a token on behalf of the clients.

With the exception of [64], the aforementioned techniques have quadratic/exponential complexity or use expensive cryptographic techniques [82], and as a result, do not support large-sized datasets at the FA clients. While [64] scales better, it does not support aggregation, and, moreover, reveals which item is in the intersection set. Fed-K-PSI [38] is a different variant of the server-based federated PSI. Each record on the client's side is represented by a key-value pair, and the server is the entity that is interested in knowing the set of identifiers that appears associated with the same value at least K times. One of the main components of Fed-K-PSI is the secure aggregation protocol that has been widely used in FL setting [19, 37, 39, 58, 87].

7.5.2 Multi-cloud-based Solutions

In multi-cloud-based solutions, a client partitions his/her local data into several parts, i.e., shares, and stores each share at different clouds. Each cloud only has partial information, thus a single cloud can not learn actual dataset [12, 28, 71, 91]. To partition data into shares, Shamir's secret-sharing [85] is the most widely-used technique.

Prio [28] is a privacy-preserving system for collecting statistics that allows multiple clients to upload their data in shares to multiple clouds, and these clouds execute only aggregation operations – count, max/min/median. Prio allows servers to verify the data they receive before storing it at their end. However, Prio only offers a mechanism for confirming the maximum number if the maximum number is known while does not provide any mechanism to compute the maximum/minimum number. Concalve [91] is an additive sharing-based system that allows to execute SQL queries over multiple clients. Conclave allows partitioning the computation such that parts of the computation can be executed at the client over cleartext and the remaining parts can be executed over additive shares. For example, a join query with selection can be partitioned such that the selection condition can be executed at clients, and then the clients create additive shares of the data that qualifies the selection condition. On the additive shares, a join query over the additive shared data belonging to multiple clients can be executed. Two other systems similar to Conclave are Senate [81], which allows collaborative SQL processing among multiple clients without using the cloud, and SMCQL [12], which is a garbled circuit based system supporting PSI via join and aggregation operations. However, these systems are inefficient when processing large datasets due to either potential memory outage and/or multiple communication rounds in the cloud. For example, SMCQL takes $\approx$ 23 hours over 23M values, while Conclave takes 8 mins over 4M values. Furthermore, to execute PSI via join operation, Conclave needs to reveal the joining column in cleartext to a trusted third party. Helen [96] and Cerebro [95] are two recent systems that perform collaborative machine learning tasks without using the cloud. Another recent system for executing queries in the multi-cloud-based is Prism [71]. Prism uses both additive shares to support Private Set Intersection (PSI)/Union (PSU) operations and multiplicative shares to offer aggregation. Furthermore, Prism [71] is able to support query executions over large datasets and multiple clients. To securely execute a computation, Prism needs at most three non-colluding cloud servers. Prism does not require communication among servers during/after/before the computation, and, consequently, is able to support PSI/PSU over 20 million values in 8 seconds. Furthermore, Prism is the only system that supports result verification operations.

8 Conclusion

In this article, we provide an overview of federated analytics, a privacy-preserving paradigm to solve queries over distributed data owned by multiple clients. We discussed the unique properties of federated analytics and how it relates to FL. We also provide a proposed taxonomy for different classes of queries in federated analytics and a survey of existing solutions in classical areas of distributed computing and secure computation. Finally, we discussed several challenges and open directions for the application and deployment of FA systems at scale. Addressing these challenges can help bring FA systems closer to being deployed in more practical scenarios to answer a wider range of queries.

Acknowledgements

This work is supported by NSF grants CCF-1763673, CNS-2002874, Defense Advanced Research Projects Agency (DARPA) under Contract No. FASTNICS HR001120C0088 and HR001120C0160, ARO grant W911NF-22-1-0165, and gifts from Intel, Qualcomm, and Cisco.

References

[1] A. Abadi, S. Terzis, and C. Dong, "Feather: Lightweight Multi-Party Updatable Delegated Private Set Intersection," *IACR Cryptol. ePrint Arch.*, 2020, 2020, 407.

[2] A. Abadi, S. Terzis, and C. Dong, "VD-PSI: Verifiable Delegated Private Set Intersection on Outsourced Private Datasets," in *International Conference on Financial Cryptography and Data Security*, Springer, 2016, 149–68.

[3] A. Abadi, S. Terzis, R. Metere, and C. Dong, "Efficient Delegated Private Set Intersection on Outsourced Private Datasets," *IEEE Transactions on Dependable and Secure Computing*, 16(4), 2017, 608–24.

[4] J. Acharya and Z. Sun, "Communication Complexity in Locally Private Distribution Estimation and Heavy Hitters," in *International Conference on Machine Learning*, PMLR, 2019, 51–60.

[5] J. Acharya, Z. Sun, and H. Zhang, "Hadamard Response: Estimating Distributions Privately, Efficiently, and with Little Communication," in *The 22nd International Conference on Artificial Intelligence and Statistics*, PMLR, 2019, 1120–9.

[6] G. Aggarwal, N. Mishra, and B. Pinkas, "Secure Computation of the Median (and Other Elements of Specified Ranks)," *Journal of cryptology*, 23(3), 2010, 373–401.

[7] A. Aly, M. Keller, D. Rotaru, P. Scholl, N. P. Smart, and T. Wood, "Scale–Mamba Documentation," in, 2020, https://homes.esat.kuleuven.be/~nsmart/SCALE/.

[8] Apple, "Apple. Learning with Privacy at Scale," *Apple Machine Learning Journal*, 2017.

[9] S. Badrinarayanan, P. Miao, S. Raghuraman, and P. Rindal, “Multi-Party Threshold Private Set Intersection with Sublinear Communication,” in *IACR International Conference on Public-Key Cryptography*, Springer, 2021, 349–79.

[10] R. Bassily, K. Nissim, U. Stemmer, and A. Guha Thakurta, “Practical Locally Private Heavy Hitters,” *Advances in Neural Information Processing Systems*, 30, 2017.

[11] R. Bassily and A. Smith, “Local, Private, Efficient Protocols for Succinct Histograms,” in *Proceedings of the Forty-Seventh Annual ACM Symposium on Theory of Computing*, 2015, 127–35.

[12] J. Bater, G. Elliott, C. Eggen, S. Goel, A. N. Kho, and J. Rogers, “SMCQL: Secure Query Processing for Private Data Networks,” *Proceedings of the VLDB Endowment*, 10(6), 2017, 673–84.

[13] A. Berlioz, A. Friedman, M. A. Kaafar, R. Boreli, and S. Berkovsky, “Applying Differential Privacy to Matrix Factorization,” in *Proceedings of the 9th ACM Conference on Recommender Systems*, 2015, 107–14.

[14] A. Bhowmick, D. Boneh, S. Myers, K. Talwar, and K. Tarbe, “The Apple PSI System,” 2021.

[15] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, “Machine Learning with Adversaries: Byzantine Tolerant Gradient Descent,” *Advances in Neural Information Processing Systems*, 30, 2017.

[16] J. Boehler and F. Kerschbaum, “Secure Sublinear Time Differentially Private Median Computation,” US Patent 11,238,167, February 2022.

[17] J. Böhler and F. Kerschbaum, “Secure Multi-Party Computation of Differentially Private Heavy Hitters,” in *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, *CCS '21*, Virtual Event, Republic of Korea: Association for Computing Machinery, 2021, 2361–77.

[18] J. Böhler and F. Kerschbaum, “Secure Multi-Party Computation of Differentially Private Median,” in *29th USENIX Security Symposium (USENIX Security 20)*, 2020, 2147–64.

[19] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, “Practical Secure Aggregation for Federated Learning on User-Held Data,” *ArXiv*, abs/1611.04482, 2016.

[20] D. Boneh, E. Boyle, H. Corrigan-Gibbs, N. Gilboa, and Y. Ishai, “Lightweight Techniques for Private Heavy Hitters,” in *2021 IEEE Symposium on Security and Privacy (SP)*, 2021, 762–76.

[21] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, *et al.*, “Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers,” *Foundations and Trends® in Machine learning*, 3(1), 2011, 1–122.

[22] P. Branco, N. Döttling, and S. Pu, "Multiparty Cardinality Testing for Threshold Private Intersection," in *IACR International Conference on Public-Key Cryptography*, Springer, 2021, 32–60.

[23] D. Chai, L. Wang, K. Chen, and Q. Yang, "Secure Federated Matrix Factorization," *IEEE Intelligent Systems*, 36(5), 2020, 11–20.

[24] D. Chai, L. Wang, J. Zhang, L. Yang, S. Cai, K. Chen, and Q. Yang, "Practical lossless federated singular vector decomposition over billion-scale data," in *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, KDD '22*, Washington DC, USA: Association for Computing Machinery, 2022, 46–55.

[25] M. Charikar, K. Chen, and M. Farach-Colton, "Finding Frequent Items in Data Streams," *Theoretical Computer Science*, 312(1), 2004, 3–15.

[26] H. Chen, K. Laine, and P. Rindal, "Fast Private Set Intersection from Homomorphic Encryption," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS '17*, Dallas, Texas, USA: Association for Computing Machinery, 2017, 1243–55.

[27] G. Cormode, F. Korn, S. Muthukrishnan, and D. Srivastava, "Finding Hierarchical Heavy Hitters in Data Streams," in *Proceedings 2003 VLDB Conference*, Elsevier, 2003, 464–75.

[28] H. Corrigan-Gibbs and D. Boneh, "Prio: Private, Robust, and Scalable Computation of Aggregate Statistics," in *14th USENIX Symposium on Networked Systems Design and Implementation (NSDI 17)*, 2017, 259–82.

[29] V. Costan and S. Devadas, "Intel SGX Explained," *Cryptology ePrint Archive*, 2016.

[30] G. F. Cretu, A. Stavrou, M. E. Locasto, S. J. Stolfo, and A. D. Keromytis, "Casting Out Demons: Sanitizing Training Data for Anomaly Sensors," in *2008 IEEE Symposium on Security and Privacy (sp 2008)*, IEEE, 2008, 81–95.

[31] D. Dachman-Soled, T. Malkin, M. Raykova, and M. Yung, "Efficient robust private set intersection," in *Applied Cryptography and Network Security*, ed. M. Abdalla, D. Pointcheval, P.-A. Fouque, and D. Vergnaud, Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, 125–42.

[32] A. Davidson and C. Cid, "An Efficient Toolkit for Computing Private Set Operations," in *Australasian Conference on Information Security and Privacy*, Springer, 2017, 261–78.

[33] E. De Cristofaro, J. Kim, and G. Tsudik, "Linear-complexity private set intersection protocols secure in malicious model," in *Advances in Cryptology - ASIACRYPT 2010*, ed. M. Abe, Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, 213–31.

[34] C. Dong, L. Chen, and Z. Wen, "When private set intersection meets big data: An efficient and scalable protocol," in *Proceedings of the 2013 ACM SIGSAC Conference on Computer and Communications Security*,

CCS '13, Berlin, Germany: Association for Computing Machinery, 2013, 789–800, DOI: 10.1145/2508859.2516701.

[35] J. C. Duchi, M. I. Jordan, and M. J. Wainwright, "Local Privacy and Statistical Minimax Rates," in *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, IEEE, 2013, 429–38.

[36] J. C. Duchi, M. I. Jordan, and M. J. Wainwright, "Privacy Aware Learning," *Journal of the ACM (JACM)*, 61(6), 2014, 1–57.

[37] A. R. Elkordy and A. S. Avestimehr, "HeteroSAg: Secure Aggregation with Heterogeneous Quantization in Federated Learning," *IEEE Transactions on Communications*, 70(4), 2022, 2372–86, DOI: 10.1109/TCOMM.2022.3151126.

[38] A. R. Elkordy, Y. H. Ezzeldin, and S. Avestimehr, "Federated K-private set intersection," in *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, *CIKM '22*, Atlanta, GA, USA: Association for Computing Machinery, 2022, 436–45, DOI: 10.1145/3511808.3557321.

[39] A. R. Elkordy, J. Zhang, Y. H. Ezzeldin, K. Psounis, and S. Avestimehr, "How Much Privacy Does Federated Learning with Secure Aggregation Guarantee?" *arXiv preprint arXiv:2208.02304*, 2022.

[40] Ú. Erlingsson, V. Pihur, and A. Korolova, "Rappor: Randomized Aggregatable Privacy-Preserving Ordinal Response," in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, 2014, 1054–67.

[41] Y. H. Ezzeldin, S. Yan, C. He, E. Ferrara, and S. Avestimehr, "Fairfed: Enabling Group Fairness in Federated Learning," *arXiv preprint arXiv:2110.00857*, 2021.

[42] G. Fanti, V. Pihur, and Ú. Erlingsson, "Building a RAPPOR with the Unknown: Privacy-Preserving Learning of Associations and Data Dictionaries," *arXiv preprint arXiv:1503.01214*, 2015.

[43] M. J. Freedman, C. Hazay, K. Nissim, and B. Pinkas, "Efficient Set Intersection with Simulation-Based Security," *Journal of Cryptology*, 29(1), 2016, 115–55.

[44] M. J. Freedman, K. Nissim, and B. Pinkas, "Efficient private matching and set intersection," in *Advances in Cryptology - EUROCRYPT 2004*, ed. C. Cachin and J. L. Camenisch, Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, 1–19.

[45] K. Frikken, "Privacy-Preserving Set Union," in *International Conference on Applied Cryptography and Network Security*, Springer, 2007, 237–52.

[46] S. Ghosh and M. Simkin, "The Communication Complexity of Threshold Private Set Intersection," in *Annual International Cryptology Conference*, Springer, 2019, 3–29.

[47] O. Goldreich, S. Micali, and A. Wigderson, "How to Play Any Mental Game, or a Completeness Theorem for Protocols with Honest Majority," in *Providing Sound Foundations for Cryptography: On the Work of Shafi Goldwasser and Silvio Micali*, 2019, 307–28.

[48] A. Google, "Federated Analytics: Collaborative Data Science without Data Collection," 2020.

[49] X. Gu, M. Li, Y. Cheng, L. Xiong, and Y. Cao, "PCKV: Locally Differentially Private Correlated Key-Value Data Collection with Optimized Utility," in *29th USENIX Security Symposium (USENIX Security 20)*, 2020, 967–84.

[50] C. Hazay and M. Venkitasubramaniam, "Scalable Multi-Party Private Set-Intersection," in *IACR International Workshop on Public Key Cryptography*, Springer, 2017, 175–203.

[51] J. Hsu, S. Khanna, and A. Roth, "Distributed Private Heavy Hitters," in *International Colloquium on Automata, Languages, and Programming*, Springer, 2012, 461–72.

[52] Y. Huang, D. Evans, and J. Katz, "Private Set Intersection: Are Garbled Circuits Better than Custom Protocols?" In *NDSS*, 2012.

[53] B. A. Huberman, M. Franklin, and T. Hogg, "Enhancing Privacy and Trust in Electronic Communities," in *Proceedings of the 1st ACM conference on Electronic commerce*, 1999, 78–86.

[54] R. Inbar, E. Omri, and B. Pinkas, "Efficient Scalable Multiparty Private Set-Intersection via Garbled Bloom Filters," in *International Conference on Security and Cryptography for Networks*, Springer, 2018, 235–52.

[55] M. Ion, B. Kreuter, E. Nergiz, S. Patel, S. Saxena, K. Seth, D. Shanahan, and M. Yung, "Private Intersection-Sum Protocol with Applications to Attributing Aggregate Ad Conversionsprivate Intersection-Sum Protocol with Applications to Attributing Aggregate Ad Conversions," *IACR Cryptology ePrint Archive*, 2017, 2017, 738.

[56] M. Ion, B. Kreuter, E. Nergiz, S. Patel, M. Raykova, S. Saxena, K. Seth, D. Shanahan, and M. Yung, "Private intersection-sum protocols with applications to attributing aggregate ad conversions," in *2020 IEEE European Symposium on Security and Privacy (EuroS&P)*, 2020, 370–89, https://eprint.iacr.org/2019/723.pdf.

[57] F. Iutzeler, "Distributed Computation of Quantiles via ADMM," *IEEE Signal Processing Letters*, 24(5), 2017, 619–23.

[58] T. Jahani-Nezhad, M. A. Maddah-Ali, S. Li, and G. Caire, "Swiftagg: Communication-Efficient and Dropout-Resistant Secure Aggregation for Federated Learning with Worst-Case Security Guarantees," *arXiv preprint arXiv:2202.04169*, 2022.

[59] Y. Jia, S.-F. Sun, H.-S. Zhou, J. Du, and D. Gu, "Shuffle-Based Private Set Union: Faster and More Secure," *IACR Cryptol. ePrint Arch.*, 2022, 2022, 157.

[60] S. Kadhe, N. Rajaraman, O. O. Koyluoglu, and K. Ramchandran, "Fastsecagg: Scalable Secure Aggregation for Privacy-Preserving Federated Learning," *arXiv preprint arXiv:2009.11248*, 2020.

[61] P. Kairouz, K. Bonawitz, and D. Ramage, "Discrete Distribution Estimation Under Local Privacy," in *International Conference on Machine Learning*, PMLR, 2016, 2436–44.

[62] 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), 2021, 1–210.

[63] P. Kairouz, S. Oh, and P. Viswanath, "Extremal Mechanisms for Local Differential Privacy," *Advances in Neural Information Processing Systems*, 27, 2014.

[64] S. Kamara, P. Mohassel, M. Raykova, and S. Sadeghian, "Scaling Private Set Intersection to Billion-Element Sets," in *International Conference on Financial Cryptography and Data Security*, Springer, 2014, 195–215.

[65] K. Kenthapadi, S. Ambler, L. Zhang, and D. Agarwal, "Bringing Salary Transparency to the World: Computing Robust Compensation Insights via LinkedIn Salary," in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, 2017, 447–55.

[66] F. Kerschbaum, "Collusion-Resistant Outsourcing of Private Set Intersection," in *Proceedings of the 27th Annual ACM Symposium on Applied Computing*, 2012, 1451–6.

[67] L. Kissner and D. Song, "Privacy-Preserving Set Operations," in *Annual International Cryptology Conference*, Springer, 2005, 241–57.

[68] V. Kolesnikov, N. Matania, B. Pinkas, M. Rosulek, and N. Trieu, "Practical Multi-Party Private Set Intersection from Symmetric-Key Techniques," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 2017, 1257–72.

[69] V. Kolesnikov, M. Rosulek, N. Trieu, and X. Wang, "Scalable Private Set Union from Symmetric-Key Techniques," in *International Conference on the Theory and Application of Cryptology and Information Security*, Springer, 2019, 636–66.

[70] "Learning New Words," https://patentimages.storage.googleapis.com/c0/fa/46/b0ab4cec65eef2/US9594741.pdf.

[71] Y. Li, D. Ghosh, P. Gupta, S. Mehrotra, N. Panwar, and S. Sharma, "Prism: Private Verifiable Set Computation Over Multi-Owner Outsourced Databases," in *Proceedings of the 2021 International Conference on Management of Data*, 2021, 1116–28.

[72] P.-L. Lions and B. Mercier, "Splitting Algorithms for the Sum of Two Nonlinear Operators," *SIAM Journal on Numerical Analysis*, 16(6), 1979, 964–79.

[73] B. Liu and Q. Tang, "Privacy-Preserving Decentralised Singular Value Decomposition," in *International Conference on Information and Communications Security*, Springer, 2019, 703–21.

[74] F. Liu, W. K. Ng, W. Zhang, S. Han, *et al.*, "Encrypted Set Intersection Protocol for Outsourced Datasets," in *2014 IEEE International Conference on Cloud Engineering*, IEEE, 2014, 135–40.

[75] C. Meadows, "A More Efficient Cryptographic Matchmaking Protocol for Use in the Absence of a Continuously Available Third Party," in *1986 IEEE Symposium on Security and Privacy*, IEEE, 1986, 134–4.

[76] T. T. Nguyên, X. Xiao, Y. Yang, S. C. Hui, H. Shin, and J. Shin, "Collecting and Analyzing Data from Smart Device Users with Local Differential Privacy," *arXiv preprint arXiv:1606.05053*, 2016.

[77] B. Pinkas, M. Rosulek, N. Trieu, and A. Yanai, "Spot-light: Lightweight Private Set Intersection from Sparse ot Extension," in *Annual International Cryptology Conference*, Springer, 2019, 401–31.

[78] B. Pinkas, T. Schneider, G. Segev, and M. Zohner, "Phasing: Private Set Intersection Using Permutation-Based Hashing," in *24th USENIX Security Symposium USENIX Security 15)*, 2015, 515–30.

[79] B. Pinkas, T. Schneider, and M. Zohner, "Faster Private Set Intersection Based on OT Extension," in *23rd USENIX Security Symposium USENIX Security 14)*, 2014, 797–812.

[80] B. Pinkas, T. Schneider, and M. Zohner, "Scalable Private Set Intersection Based on OT Extension," *ACM Transactions on Privacy and Security (TOPS)*, 21(2), 2018, 1–35.

[81] R. Poddar, S. Kalra, A. Yanai, R. Deng, R. A. Popa, and J. M. Hellerstein, "Senate: A Maliciously-Secure MPC Platform for Collaborative Analytics," in *30th USENIX Security Symposium (USENIX Security 21)*, 2021, 2129–46.

[82] S. Qiu, J. Liu, Y. Shi, M. Li, and W. Wang, "Identity-Based Private Matching Over Outsourced Encrypted Datasets," *IEEE Transactions on cloud Computing*, 6(3), 2015, 747–59.

[83] S. Ramanathan, J. Mirkovic, and M. Yu, "Blag: Improving the Accuracy of Blacklists," in *NDSS*, 2020.

[84] P. Rindal and M. Rosulek, "Improved Private Set Intersection Against Malicious Adversaries," in *Advances in Cryptology – EUROCRYPT 2017*, ed. J.-S. Coron and J. B. Nielsen, Cham: Springer International Publishing, 2017, 235–59.

[85] A. Shamir, "How to Share a Secret," *Communications of the ACM*, 22(11), 1979, 612–3.

[86] M. Shayan, C. Fung, C. J. Yoon, and I. Beschastnikh, "Biscotti: A Blockchain System for Private and Secure Federated Learning," *IEEE Transactions on Parallel and Distributed Systems*, 32(7), 2020, 1513–25.

[87] J. So, C.-S. Yang, S. Li, Q. Yu, R. E Ali, B. Guler, and S. Avestimehr, "Lightsecagg: A Lightweight and Versatile Design for Secure Aggregation in Federated Learning," *Proceedings of Machine Learning and Systems*, 4, 2022.

[88] J. Steinhardt, P. W. W. Koh, and P. S. Liang, "Certified Defenses for Data Poisoning Attacks," *Advances in neural information processing systems*, 30, 2017.

[89] A. Tueno, F. Kerschbaum, S. Katzenbeisser, Y. Boev, and M. Qureshi, "Secure Computation of the k-th Ranked Element in a Star Network," in *International Conference on Financial Cryptography and Data Security*, Springer, 2020, 386–403.

[90] P. Voigt and A. Von dem Bussche, "The EU General Data Protection Regulation (GDPR)," *A Practical Guide, 1st Ed., Cham: Springer International Publishing*, 10(3152676), 2017, 10–5555.

[91] N. Volgushev, M. Schwarzkopf, B. Getchell, M. Varia, A. Lapets, and A. Bestavros, "Conclave: Secure Multi-Party Computation on Big Data," in *Proceedings of the Fourteenth EuroSys Conference 2019*, 2019, 1–18.

[92] D. Wang, S. Shi, Y. Zhu, and Z. Han, "Federated Analytics: Opportunities and Challenges," *IEEE Network*, 2021.

[93] Q. Ye, H. Hu, X. Meng, and H. Zheng, "PrivKV: Key-Value Data Collection with Local Differential Privacy," in *2019 IEEE Symposium on Security and Privacy (SP)*, 2019, 317–31, DOI: 10.1109/SP.2019.00018.

[94] E. Zhang, F. Li, B. Niu, and Y. Wang, "Server-Aided Private Set Intersection Based on Reputation," *Information Sciences*, 387, 2017, 180–94.

[95] W. Zheng, R. Deng, W. Chen, R. A. Popa, A. Panda, and I. Stoica, "Cerebro: A Platform for {Multi-Party} Cryptographic Collaborative Learning," in *30th USENIX Security Symposium (USENIX Security 21)*, 2021, 2723–40.

[96] W. Zheng, R. A. Popa, J. E. Gonzalez, and I. Stoica, "Helen: Maliciously Secure Coopetitive Learning for Linear Models," in *2019 IEEE Symposium on Security and Privacy (SP)*, IEEE, 2019, 724–38.

[97] W. Zhu, P. Kairouz, B. McMahan, H. Sun, and W. Li, "Federated Heavy Hitters Discovery with Differential Privacy," in *International Conference on Artificial Intelligence and Statistics*, PMLR, 2020, 3837–47.