Federated Visualization: A Privacy-preserving Strategy for Decentralized Visualization

Wei Chen, Yating Wei, Zhiyong Wang, Shuyue Zhou, Bingru Lin, and Zhiguang Zhou

Abstract—We present a novel privacy preservation strategy for decentralized visualization. The key idea is to imitate the flowchart of the federated learning framework, and reformulate the visualization process within a federated infrastructure. The federation of visualization is fulfilled by leveraging a shared global module that composes the encrypted externalizations of transformed visual features of data pieces in local modules. We design two implementations of federated visualization: a prediction-based scheme, and a query-based scheme. We demonstrate the effectiveness of our approach with a set of visual forms, and verify its robustness with evaluations. We report the value of federated visualization in real scenarios with an expert review.

Index Terms—Privacy-preserving visualization, federated visualization, decentralized visualization

1 INTRODUCTION

Manuscript received xx.xxx. 201x; accepted xx.xxx. 201x. Date of Publication xx.xxx. 201x; date of current version xx.xxx. 201x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org.

In the big data era, data analysis can be performed in parallel. For example, in analyzing infectious disease data, it is necessary to combine data of disease materials and cases stored in different locations and medical institutions [28,30]. Similarly, urban data analysis demands the integration of mobility information acquired from subway, taxis and automobiles [27,35]. Roughly speaking, conventional client/server based visualization or web-based visualization takes a data-intensive mode, that is, datasets are assembled, processed and visualized in a main server [6,39]. Alternatively, distributed or parallel visualization [5] employs a decentralized mode, namely, data and tasks are divided into pieces over various clients, and required information can be transmitted among clients. In both modes, raw data or processed data is allowed to be transmitted among clients.

One recent trend in building big data infrastructures is the privacy awareness, as witnessed by numerous literature on security and pri-
Facing the challenges of data privacy in distributed machine learning, federated learning employs a shared global model to federate the learned local models that run only in individual clients [10-12]. Inspired by the idea of federating learned features rather than raw data, we design a novel decentralized visualization strategy, which divides the tasks of data transformation and visual mapping into pieces in clients (local modules) and then composes the secured results in a server (a shared globe module). However, practicing federated visualization is non-trivial. In particular, it faces a set of challenges as follows:

C1. A conventional visualization pipeline consists of multiple steps, including data processing, transformation, visual mapping, and user interaction. Data leakage can take place in each step. Thus, data encryption should be thoroughly incorporated as a necessary component.

C2. For the reason of privacy protection, data in local modules should be kept locally, and only desensitized information can be transmitted to the global module. How to achieve faithful visualization based on composed information in the global module is a difficult task.

C3. Enabling and evaluating the validity and usability of federated visualization is different from the ones of conventional visualization methods. Multiple perspectives like encryption, privacy, and effectiveness should be considered.

Inspired by federated learning, our solution employs a data encryption scheme in transmitting data between the server and clients (C1). To localize the underlying data and visualization tasks (C2), we design a two-stage pipeline: prior to the composition of encrypted visual features or parameters in the server, each client locally performs data transformations and visual encodings with its associated data. Specifically, we propose two implementations (C3): a prediction-based scheme, and a query-based scheme. The first one builds a prediction model to encode the visual features computed from local data, and decode them in the global module. Its result is an approximation of the targeted visualization in the local environment. In contrast, the query-based scheme yields accurate results by computing encrypted and specified range of visual features locally and composing them globally. Both are verified with a set of visual forms, together with case studies and an expert review (C3).

To our best knowledge, our approach is the first attempt to tackle data privacy issues in a decentralized visualization framework. In summary, the main contributions of our work are:

- A novel federated visualization framework for privacy-preserving decentralized visualization;
- Efficient implementations that support visual exploration and query of multi-sourced datasets, together with experimental verifications.

The rest of this paper is organized as follows. Section 2 reviews the related work. Section 3 explains our new strategy. Implementations and visual exploration are elaborated in Section 4. Section 5 presents the evaluations. We discuss the limitations and future work in Section 6 and conclude this paper in Section 7.

2 RELATED WORK

2.1 Privacy-preserving Visualization

Privacy is the right of individuals to take complete control of their information and decide when, how, and to what extent this information is shared with others [1]. Visualisation is regarded as an effective means to make the data or data features easily recognizable and interpretable. Therefore, privacy issues of visualization exist not only in data processing but also visualization itself. Pioneered works on privacy-preserving visualization [11] generally leverage established privacy preservation schemes like $k$-anonymity [12], $t$-diversity [23], $r$-closeness [20] to protect information leakage in visualizations. For instance, syntactic anonymization is employed to protect the information exposed in parallel coordinates [11]. A privacy-preserving diversity method (ppDIV) [27] is presented to avoid disclosure of location privacy from trajectory heatmap. Distinctive visual interfaces are designed to depict and reduce the leakage risk in visualizing event sequence datasets [29], tabular data [34], and network datasets [7][33]. In general, there is a trade-off between the privacy gain and loss of utility. It has demonstrated to be favorable to support visual understanding of the quantitative relationships between privacy parameters and vulnerable visualization configurations [12].

Note that, above-mentioned works do not consider the disclosure of privacy in data transmission process, and are naturally not suitable for decentralized visualization.

2.2 Distributed Visualization

Distributed visualization [5] increases the scalability of visualization for data-intensive or computation-intensive tasks. A distributed environment is proposed for exploring correlations of large-scale simulation datasets. Well-designed data structures can be used to improve the performance of distributed visualization, like the PSH [19] and quadtree [36]. GPU-based parallel computing can further improve performance [31]. Likewise, the runtime efficiency of graph sampling [25] and sparsification [2] can be benefited from distributed computing. Note that these studies emphasize on the performance issues. There is little effort made on data privacy in distributed visualization. Saha et al. [30] introduce decentralized data stochastic neighbor embedding (dSNE) to enable embedding and visualization of sensitive neuroimaging data. Similarly, a decentralized brain imaging data analysis is proposed with new data processing and visualization algorithms [23].

2.3 Federated Learning

The fundamental idea of federated learning (FL) [18] is to learn an integrated model with data distributed on clients. It can balance performance and communication efficiency while preventing leakage of sensitive data. A large number of studies have been paid on the usability, scalability, and performance of FL [21]. For instance, an optimized federated learning strategy is proposed to train a high-quality centralized model [17]. By investigating enhanced privacy protection algorithms, different levels of privacy protection can be achieved in FL at a minor loss in model performance [14][24]. A recent trend is to incorporate FL within a variety of application scenarios [15][16][22]. Based on the distribution of training data, the federal learning approaches can be classified into three categories: horizontal federated learning, vertical federated learning, and Federated Transfer Learning [37].

3 FEDERATED VISUALIZATION

The goal of federated visualization is to protect privacy in a decentralized visualization framework. Suppose that a set of datasets $D = \{D_1, D_2, D_3, ... , D_N\}$, owned by clients $C = \{C_1, C_2, C_3, ... , C_N\}$, respectively. Assume that $V_i$ is the visualization of $D_i$, and $V$ is the visualization of $D$. Without loss of generality, $V_i$ and $V$ share the same visual transformations and mappings, resulting in a set of visual features, denoted as $V = \{V_{F_1}, V_{F_2}, ..., V_{F_M}\}$. Here, $V_{F_i} = \{v_{f_{i1}}, v_{f_{i2}}, ..., v_{f_{iM}}\}$ is a vector of visual features, such as position, color, size, and shape. $V$ represents the global visual features of $D$. In the conventional parallel visualization, $D$ is collected and processed globally in a server $S$, exposing a risk of privacy leakage. In this paper, we only consider the scenario that the visual features of each client have the same dimensions, which refers to the situation of horizontal federated learning [37].

We follow the workflow of horizontal federated learning and propose a two-stage federation strategy. Datasets are stored and processed locally in each client $C_i$. $C_i$ only transmits encrypted parameters $P_i$ which represents visual features and cannot be used to recover the original data. $P_i$ acts as the messenger of the original data $D_i$ and $V_i$ to avoid direct exposure of the raw data. The definition and construction of $P_i$ varies with different federation schemes (see Section 3.2).

3.1 Pipeline

The pipeline consists of four main stages, as shown in Figure 2.
1) **Data Preprocessing.** First, the data scope $E_i$ associated with $D_i$ is specified by analysts, e.g., data in a specific latitude and longitude range, or data in a specific time period. It can be defined uniformly for all clients, or be specified interactively on the visual interface during the exploration process.

2) **Data partitioning and aggregation.** For each client, its data is partitioned in small pieces uniformly. The resolution of the partition can be predefined or interactively specified subject to employed visual forms. For instance, creating a heatmap requires to divide the data over a 2D grid. Similarly, generating a histogram of traffic flow in one week needs to define the granularity along the time axis.

Then, data aggregation is performed over each binned range of the partition. The aggregated values are mapped into visual features, forming a visual feature set $A_i$. Each element $a_j$ of $A_i$ is a key-value pair, namely, $a_j = (\text{index}_j, d_{\text{index}_j})$. Here, $\text{index}_j$ denotes the index, and $d$ is the feature values of index $j$.

This process is performed in each client, resulting in visual features with identical dimensions.

3) **Data Representation.** This process requires to encode and encrypt visual features in each client and utilize them in the server. For that, we design two implementations, which are described in Section 3.2.

4) **Visualization Generation.** Based on the encrypted visual features from local clients, the server composes a set of global visual features and generates a visualization. Section 3.3 describes the ways of visualization creation associated with two representations introduced in Section 3.2.

3.2 Federated Representation

In the third stage, it is very important to keep visual features in each client from being exposed during the transmission process. We solve this issue by transmitting encrypted parameters that represent visual features. In particular, two implementations are designed:

- **Query-based**, which encodes local visual features using secure aggregation techniques and decodes them in the sever;
- **Prediction-based**, which trains a prediction model in each client through federated learning, and predicts the result in the server by means of all parameters of local prediction models.

3.2.1 The query-based scheme

We ensure that all clients possess pair-wise secure communication channels and use secure aggregation [3] for encryption.

The query-based scheme consists of three steps:

1) Sampling random vectors. Each pair of clients first prepares a random vector for each other. That is, each client $C_i$ locally samples a random vector $R_{i,j}, j \in [1,N] \land i \neq j$ for each other client $C_j$. Specifically, $R_{i,j}$ and $R_{j,i}$ are a one-to-one pair. For example, $R_{1,3}$ and $R_{3,1}$ form the random vectors prepared by $C_1$ and $C_3$ for each other [Figure 3].

2) Exchanging random vectors and computing perturbations. Client $C_i$ and $C_j$ exchange $R_{i,j}$ and $R_{j,i}$ over their secure channels. To ensure transmission security, perturbations $P_{i,j} = R_{i,j} - R_{j,i}, i \neq j$ are computed. The sum of visual feature and perturbation vectors is sent to the server: $D_{\text{upload}} = V F_i + \sum_{j=1}^{N} P_{i,j}$.

3) Computing global visual features in the server. The server receives the perturbed vectors $D_{\text{upload}}, i \in [1,N]$ uploaded by clients, and sums them:

$$D_{\text{sum}} = \sum_{i=1}^{N} D_{\text{upload}}, i \in [1,N]$$

The result $D_{\text{sum}}$ is guaranteed to be accurate visual features because the paired perturbations in $D_{\text{upload}}$ are neutralized. And the values from each client will not be inferred.

3.2.2 The prediction-based scheme

Encoding visual features can be accomplished by precomputing a representation with prediction-based approaches. Conventional solutions include data fitting like logic regression, hashing, and neural network approaches. We choose to use neural network-based methods, and mimic the federated learning framework where all clients contribute to training a shared global model that represents global data features. Below we introduce our scheme based on neural network-based models.

The deep neural network model is very suitable for encoding an array of information. We leverage a fully connected deep network [29], as shown in Figure 4 (a). It consists of an embedding layer that converts the input data into vectors to improve the efficiency of model training, and four fully connected layers. To enable the federation of the prediction models, we design an adapted NN model, which consists of a global part and local parts (See Figure 4 (b)). The server keeps the global part, while each client keeps a local part. We consider the visual feature set $A_i$ as the local model training set of $C_i$, where $\text{index}_j$ is used as the model input and $d_{\text{index}_j}$ is used as the label data, that is, the model output. We use the loss function defined as $\sum (d_{\text{index}_j} - d_{\text{index}_j})^2$, where $i$ is the index of the underlying data piece, $d_{\text{index}_j}$ denotes the value of the data predicted by the global model. We use the quadratic cost so that $d_{\text{index}_j}$ will be close to the average of $d_{\text{index}_j}$ over all clients.

The training process consists of three steps:

1) Initialization. The server S initializes a global neural network model $M_{feed}$. Each client keeps a copy of the initialized model from the server. The server sends encrypted initial parameters of the global NN model to each client.
2 Local training. Each client decodes the parameters sent from the server, and uses them to update the local models. Then, the local NN model is iteratively trained for several rounds with $A_t$ as the training set, yielding updated parameters of the local NN model. The parameters are encrypted and sent back to the server.

3 Federated averaging. The server performs a federated averaging over the set of encrypted parameters (in the form of $P(t)$, where $t$ represents one round, including the embedding matrix and weight matrix) from all clients, and computes updated global parameters, $P(t)$: $P(t) = \frac{1}{\sum_{i=1}^{N} P_i(t)}$. The parameters are then encrypted and sent to each client.

The 2nd and 3rd steps are iteratively performed, until the loss function converges, or say, values of two consecutive iterations are adequately close. The trained NN model parameters are stored on the server for reuse.

It is approximately the average of aggregated values of each client within the data range $index_j$. Therefore, visual features of the global data on $index_j$ are $N \times d_j$. In this way, we can get all visual features by feeding all indices into the global model $M_{fed}$, as shown in Figure 4(b).

4 VISUAL INTERFACE

We design and develop a prototype system for federated visualization.

4.1 Interface

The visual interface [Figure 1] has two main modules: the analysis view (a-d), and model view (e). The analysis view includes the control panel and several visualization chart views. In the control panel, we provide several widgets to support interactive configurations. Users can select the data range according to some criteria, such as a time range, a certain attribute value, etc. Users can also interactively control the accuracy of the training model. The visualization results are presented in chart views which can be configured upon user preferences or application requirements. Supported interactions for each chart include brushing, range selection, data navigation. To indicate the loading process, a loading circle is shown in the top right corner of each chart view [Figure 1(b-d)]. The model view shows information of the server and participating clients, and details of the training process. In the monitor panel, a liquid fill gauge shows the dynamic change of each client’s loss, and the box plot below illustrates the global loss variation along the rounds. The slide bar below can adjust the display interval.

4.2 Federated chart generation

Below, we show the creation of a variety of visualizations, ranging from structured tabular data to hierarchical datasets. Both the query-based scheme and prediction-based scheme can be leveraged to get global visual feature values for visualization creation. The flexibility of federated visualization is highlighted in each example.

Histogram. To create a histogram, the number and height of bars need to be determined. Suppose the target histogram has 7 bars which represent 7 days of a week, and the height of each bar represents the traffic volume of a day. To generate it, the local data in each client is divided into 7 pieces, each of which the total traffic volume is computed. Thus, a local visual feature set for visual mapping is obtained in each client. To get the global visual features of the entire dataset, on one hand, we can use query-based scheme to directly encrypt and merge visual features of each client; on the other hand, prediction-based scheme can be used to fit the global visual features where the local visual feature set is used as the local training set. To further explore the finer granularity of the information, we create a stacked histogram. The underlying data of each bar is divided and computed to obtain a new visual feature set for visual mapping of the stacked histogram.

In general, the way of creating a histogram can be applied to generating other chart types, such as line chart, pie chart, violin plot, area chart, and radar chart.

Heatmap. The heatmap in Figure 1 consists of 15960 (190×84) grids divided by uniform longitude and latitude. Each grid cell has its index and value. The accumulated density in each grid cell is encoded by hue, which represents the number of records in the corresponding geographic area. On each client, a heatmap is calculated by local data. Those heatmaps, reflecting the distributions of dataset on each client, have the same cell index. Our goal is to get an overall heatmap which reflects the distribution of all clients’ dataset without leaking privacy.

Similar but more complex, to create an ODMap (Origin-Destination flow map), the local data is partitioned according to the latitude and longitude of the map. In each resultant part, the partition is performed along the latitude and longitude of the destinations. Then, the traffic flow from the origins to the destinations is computed, resulting in a four-dimensional division. When the division is relatively dense, the predictive-based scheme is more time-consuming than the query-based one, because considerable visual feature values need to be trained.

Sankey diagram. Sankey diagrams are a specific type of node-link diagrams. Several entities are represented by rectangles or text, and
their links are represented with arrows or arcs whose widths are proportional to their flow quantities. To create a sankey diagram, flow quantities from different clients should be integrated without merging data. To solve this issue, each client defines data statistics rules according to the visual analysis task, divides and counts the local data according to the rules. For example, when analyzing population migration between different countries, each client divides and counts in and out fields based on different countries. These processed data are then encrypted and merged for final visualization.

Squarified Treemaps. Federated visualization is capable of creating recursive layouts, such as treemap, sunburst, and circle packing diagram. The data structure of hierarchical data (where the data is organized into a tree structure) is more complex than that of tabular data. Without loss of generality, the tree structures in all clients are assumed to be identical. For example, in the case of aggregating street sign-in information from various regions of the country from different clients, the hierarchy of the geographic locations of the clients is the same. Thus, to craft a treemap, each client only needs to count the numbers of corresponding leaf nodes and send their encryptions, then the server can summarize and render the treemap.

4.3 Federated Visual Analysis

Visual exploration with federated visualization forms a seamless composition of multiple client and server steps. First, the visualization configurations of federated visualization are determined in the front-end. Users can specify the data range, the accuracy, and the representation mode in the control panel. Second, the visualization configurations are sent to the server. The server determines the scheme based on the chosen representation mode and sends the configurations to all clients. Third, clients process the local data according to the representation mode and send the results back to the server. Fourth, the front-end gets visual feature values from the server and creates the visual forms.

Several basic interactions, such as selection, navigation, filtering and connection are supported. The information transmission between the server and clients is triggered once a user interaction happens. For the query-based scheme, the dataset stored in each client is queried. For the prediction-based scheme, the model needs to be re-trained, and related parameters are updated for each new query. Pre-computing results of all or parts of model configurations can eliminate the workload of model re-training in run-time visual queries, as has been employed in previous works.

5 Evaluation

We implement a web-based prototype system. The back-end is written with Python. The front-end visualization is implemented with a combination of HTML5, JavaScript, and D3.js. SocketIO is used to communicate between clients and the server. TensorFlowFlow is used to implement our deep learning model. All clients are simulated on a single machine with eight simulation nodes, which has an i7-8700 processor and a GTX 1660Ti GPU.

The dataset used for our prototype system is a Urban-Mobility dataset, which contains 8,283,605 records, and is obtained from Didi Chuxing Technology Co., the biggest ride-hailing service company in China. The basic unit of recorded data is a taxi order, which contains information about the latitude and longitude of order start and order end, the time of order start and order end. The time spans from May 1st, 2017 to October 31st, 2017. These records are distributed over eight clients and are non-i.i.d. across clients (i.e., they are not independent and identically distributed (i.i.d.)). We use a set of synthetic data (8 million items) to generate other representative charts.

5.1 Quantitative evaluation

The quantitative evaluation compares our two schemes with the centralized counterpart with three measures: performance, accuracy and visual quality.

5.1.1 Performance

The response time includes time spent on data preprocessing, data partition and aggregation, data representation and visualization generation. The difference between federated visualization and traditional visualization methods lies in the data representation phase. The timing measurement indicates the impact of factors such as the granularity of data partitioning, the number of training rounds, the number of training epochs, and the number of clients.

Query-based scheme. Compared with traditional centralized visualization, extra time is spent on data encryption. It mainly includes three parts: generating random vectors, exchanging random vectors, encrypting and decrypting data. We assume that both the number of clients and the granularity of data partitioning affect the response time.

Number of clients: The number of clients ranges from 3 to 27. The response time presents a positive growth trend with the increasing of the number of clients (Figure 5(a)). This result means that the response time increases when the number of participating clients gradually increases. In addition, the response time of heatmap is the longest and that of treemap is longer than that of histogram. This is because their granularity of data partitioning decreases and is 380 × 168, 6000, 24 × 7, respectively. It is clear that the increasing rate of the response time depends on the underlying encryption mechanisms. Figure 5(b) shows the variation trend of different time parts of heatmap as the number of clients increases. Exchanging random vectors dominates the computing time. This is because its time complexity is \(O(n^2)\).

Granularity of data partitioning: Different chart types have different granularities, that is, the number of visual features, yielding varied response times (Figure 5(a)). To further analyze the relationship between the granularity of data partitioning and response time, we test the change of the time for the heatmap with various granularities of data partitioning: 95 × 42, 190 × 84, 380 × 168, 760 × 336, and 1520 × 672. Figure 5(c) shows that the response time has a linear relationship with the granularity of data partitioning. The result indicates that the response time increases linearly with the increasing of visual features.

Prediction-based scheme. The time overhead for the prediction-based scheme lies in model training. Several factors may influence the time of model training, such as the granularity of data partitioning, the training settings, the number of clients, and the data distribution.

Granularity of data partitioning: Figure 5(d) shows the relationship between the response time and the granularity of data partitioning. The result is similar to that of the query-based scheme, indicating that when the number of visual features to be fit increases, the train-
Training settings: The response time presents a positive growth trend as the number of training round or training epoch increases (Figure 6 (a,d)). Meanwhile, the response time of charts with a finer granularity will be relatively longer to fit more visual feature values.

Number of clients: The response time increases proportionally with the number of clients (Figure 6). As the number of participating clients increases, the amount of training data increases, more time is needed to reach a model convergence.

Training data distribution: Many works [38] have studied the distribution of training data in the field of federated learning. The training data on each client can be i.i.d. or non-i.i.d. In practice, we randomly generate two pieces of treemap data, which are i.i.d. and non-i.i.d., to simulate actual usage scenarios and analyze the impact of data distribution on the response time. As shown in Figure 6 (a,d,g), there is almost no difference in response time between two distributions, indicating that the data distribution has little effect on the response time with our method.

5.1.2 Accuracy

With the query-based scheme, queried visual features are accurate. In contrast, errors occur with the prediction-based scheme because visual features are approximately generated. Several factors, such as the granularity of data partitioning, the training settings, the number of clients, and the training data distribution, may influence the accuracy of the visualization. Two accuracy measurements are considered. We use Jensen–Shannon divergence (JSD) metric to evaluate the similarity between the approximate and accurate visualization result distributions. The JSD is bounded by 1 for two distributions in our experiment (i.e. \( \text{JSD} \in [0, 1] \)). A smaller JSD value indicates that two distributions are more similar. We also use relative error (RE) metric to evaluate the difference between two visualizations. Suppose that a visualization contains \( N \) feature values. Federated visualization fits original \( N \) visual features \( y_1, \ldots, y_N \), and returns \( N \) values \( \hat{y}_1, \ldots, \hat{y}_N \). The RE of the visualization is defined as:

\[
RE = \frac{\sum_{i=1}^{N} |y_i - \hat{y}_i|}{\sum_{i=1}^{N} |y_i|}
\]

Training settings: JSD and RE are inversely proportionally to the
Weekly traffic flow is visualized (Figure 1 (b)), depicting the temporal distribution of mobility in city H. It indicates that traffic in July and August is heavier than in other months, which confirms that the city attracts a large number of tourists in the summer vacation.

Next, we query the traffic flow from May 1st to October 31st. The heatmap depicts the traffic flow. The more intensive the location is, the redder it will be on the heatmap. heatmap and ODMap.

The first case generates charts of taxi trajectories in an urban area. Exploring temporal distribution of mobility. First, a calendar diagram is generated to show the overview pattern of taxi trajectories. Weekly traffic flow is visualized (Figure 1 (b)), depicting the temporal distribution of mobility in city H. It indicates that traffic in July and August is heavier than in other months, which confirms that the city attracts a large number of tourists in the summer vacation.

Exploring spatial pattern of mobility. Next, we query the traffic flow from May 1st to October 31st. The heatmap depicts the traffic flow. The more intensive the location is, the redder it will be on the heatmap. heatmap and ODMap.

5.2 Case studies
5.2.1 Case One: Urban data exploration
The first case generates charts of taxi trajectories in an urban area. Exploring temporal distribution of mobility. First, a calendar diagram is generated to show the overview pattern of taxi trajectories. Weekly traffic flow is visualized (Figure 1 (b)), depicting the temporal distribution of mobility in city H. It indicates that traffic in July and August is heavier than in other months, which confirms that the city attracts a large number of tourists in the summer vacation.

Exploring spatial pattern of mobility. Next, we query the traffic flow from May 1st to October 31st. The heatmap depicts the traffic flow. The more intensive the location is, the redder it will be on the heatmap. heatmap and ODMap.

5.2.2 Case Two: Evaluating privacy risks
The privacy risks mainly arise from the availability of external information about database records and the background knowledge of the attackers. For example, when a private database is joined with a publicly available database, attackers may re-identify sensitive attributes through quasi-identifiers (like age, gender, zip code, etc.).

Federated visualization localizes the underlying data and visualization task in each client, and supports the creation of visualization charts that can be generated through data aggregation. The privacy risks of our approach are evaluated from two aspects below.

Reducing privacy risks through aggregation. The dataset in each client is processed into aggregated values for visual mapping, and then the server performs a global aggregation of local computations for
a global visualization. Aggregation is a common privacy-preserving operation, where sensitive data values are either hidden or generalized. Many applications [8, 27, 35] have proven that it can effectively reduce the risk of privacy leakage. While aggregation-based visualizations can be considered harmless from the perspective of a privacy attack, this assumption may not always be true [13]. Considering the background knowledge and external information held by a malicious attacker, some patterns may be disclosed when sensitive visualization charts are explored. For example, in a histogram, if the height of a certain bar is 1, the attacker is likely to infer sensitive attributes or reveal individual identities with knowledge of quasi-identifiers. Our approach shows the visualization of aggregated data in a global view. Even if an attacker can identify some sensitive information from the global view, inferring which client the data came from is intractable. This reduces the risk of data distribution leakage for a single client to a certain extent.

Privacy preservation in a decentralized environment. By using federated visualization, decentralized datasets do not need to leave the client. Each client can leverage the local storage and computational resources to facilitate the generation of a global visualization, thereby reducing the risk of data exposure and reducing communication overhead. For the query-based scheme, we use the secure aggregation scheme to protect the privacy of aggregated values generated in each client. In essence, it implements privacy preservation of data through perturbation [13], the server aggregates the encrypted parameters and clears the perturbations, resulting in accurate aggregation information. For the prediction-based scheme, the encryption of parameters in model training is done with SMC [4], which has proven to be effective for protecting data leakage against the semi-honest server.

5.2.3 Case Three: Creating different charts with federated visualization

Representative charts generated with our approach are shown in Figure 11. When loading the gallery page, the response time of different charts varies. Charts with the same granularity of data partitioning have similar response time, and the data type and chart type have little effect. This is because that the granularity of the data partitioning determines the size of the model training set. A finer granularity means that the model training set will be larger, resulting in more training time. The time spent on data preprocessing of different data types and the rendering of different charts has relatively little influence on the response time. More charts can be seen in the supplementary video.

5.3 Expert Interview

To evaluate the effectiveness of federated visualization, we conducted one-on-one interviews with three experts.

Background. All experts have experiences in visual analysis of urban data. In a prior discussion, they agreed that privacy is a big concern in visually analyzing decentralized data. A comprehensive analysis of a problem is not possible if data can not be shared. They have no knowledge about federated visualization.

Process. In each interview, we first introduced the background of federated visualization, our visual interface and chart examples in gallery, and demonstrated how the system works with the first case. Then we asked them to freely explore urban data, answered their questions and observed their behaviors. Finally, we collected their feedback. This process took approximately 60 minutes.

Feedback. Overall, all experts felt that our system has no difference from conventional visual analysis systems and can be easily analyzed. They confirmed this convenience to various chart types and some basic interactions. An expert liked the provision of different accuracies in the system, which makes it flexible for different privacy scenarios. Surely, a low accuracy has little influence on visual distributions but can shorten the system response time. An expert tried to re-identify an arbitrary individual, but failed. This verifies the effectiveness of our system.

All experts commented, “It takes a long time to refresh the heatmap and ODMap interactively, while the histogram can be refreshed in real-time.” To handle complex charts and large datasets, schemes for optimizing query performance like the pre-computation scheme, could be incorporated in the future. An expert reported, “I cannot feel the results are inaccurate, and I am not confident in the results of the analysis.” And he further suggested us to present the inaccuracy of the result. For example, visual encodings of uncertainty can be strengthened to clarify the inaccuracy. They also discussed the possibility of using federated visualization to create complicated visualizations. They hoped that our approach could be extended to support complex scenarios, such as graph data or multi-source datasets from different domains.
works in this scenario. If some attribute values are missing in a client, visualization, the overall accuracy decreases, because the client does not visualize is feasible. For example, creating a pie chart needs to information to the server, such as zero. Consequently, the composed the five clients collude with each other, these four clients send invalid privacy of transported parameters. In our implementation, if four of leakage. Our approach employs secure aggregation to protect the different ways. However, collusion among clients can result in privacy honestly, but the server may try to learn additional information in server can lie to client privacy under the threat conditions of server is honest-but-curious becomes even longer, resulting in unfriendly user experience. First, the number of clients should be larger than 3. Otherwise, the collected information in the server can be used to infer secured information. Second, clients should make a strict protocol agreement and be honest because our approach can not identify the collusion among clients. Third, our prediction-based scheme re-trains the model after each user interaction. When the number of visual features to be trained reaches tens of thousands, the response time lead to privacy exposure.

Performance. Several factors, such as response time and accuracy, may influence the performance of federated visualization. Compared with traditional centralized visualization, our approach has extra time overhead to achieve privacy protection. The query-based scheme integrates encrypted visual features on the server side, and computes the sum of visual features of each client without loss. Data encrypted transmission costs extra response time. The prediction-based scheme uses the data on each client to train a global model to fit global visual features. Ideally, its accuracy can reach 100%. However, in real scenarios, models are often unable to fit completely, because of time constraints, data distributions (non-i.i.d. or i.i.d.), or federated learning characteristics. Quantitative results show that the time increases with the accuracy. A balance between the accuracy and the time can be achieved. In addition, as the amount of data required to create visual charts increases, obtaining more accurate results is more time-consuming. There is much capacity for further performance optimization. For example, the level-of-details scheme can be employed in visualizing a heatmap.

The query-based scheme can generate more accurate results than the prediction-based scheme, and has a better running performance. Thus, it is more feasible for time-critical situations.

Limitations. First, the number of clients should be larger than 3. Otherwise, the collected information in the server can be used to infer secured information. Second, clients should make a strict protocol agreement and be honest because our approach can not identify the collusion among clients. Third, our prediction-based scheme re-trains the model after each user interaction. When the number of visual features to be trained reaches tens of thousands, the response time becomes even longer, resulting in unfriendly user experience.

6 DISCUSSION

Below, we discuss our work in terms of generality, privacy-preserving and performance. We also summarize the limitations and suggest directions for further work.

Generality. Case studies on heatmap, ODMap, and calendar heatmap verify the effectiveness of our approach. Actually, our approach is applicable to all visualization charts that can be generated through data aggregation (average and addition). Essentially, our approach aggregates visual features from each client without obtaining the specific raw data of each client. This means that if visual features of a chart can be obtained by summing data attributes, privacy-preserving visualization is feasible. For example, creating a pie chart needs to count the amount of data items for corresponding sectors from clients. The prediction-based scheme has a better generality than the query-based scheme. In particular, the query-based scheme may yield less accuracy when there are missed attribute values. For example, the server needs to use data from three clients (C1,C2,C3) to visualize the probability of two events E_A and E_B. Both the data held by C1 and C2 can be used to calculate the probability of two events, while C3 can only obtain the probability of E_A and does not know the probability of E_B. In this case, the query-based scheme with which each client needs to send all visual features does not work. If the probability of E_B is set to be zero by C3 to complement all visual features to participate in federated visualization, the overall accuracy decreases, because the client does not know the probability of E_B. In contrast, the prediction-based scheme works in this scenario. If some attribute values are missing in a client, the client only employs existing data for local training. The missing data does not affect accuracy.

Privacy preservation. Secure aggregation is widely used in federate learning to encrypt the model gradients of each client and avoid backtracking the original data. Secure aggregation can guarantee privacy under the threat conditions of server is honest-but-curious and server can lie to client, that is, all clients follow the protocol honestly, but the server may try to learn additional information in different ways. However, collusion among clients can result in privacy leakage. Our approach employs secure aggregation to protect the privacy of transported parameters. In our implementation, if four of the five clients collude with each other, these four clients send invalid information to the server, such as zero. Consequently, the composed visualization only shows the data of the innocent client, which may lead to privacy exposure.

Performance. Several factors, such as response time and accuracy, may influence the performance of federated visualization. Compared with traditional centralized visualization, our approach has extra time overhead to achieve privacy protection. The query-based scheme integrates encrypted visual features on the server side, and computes the sum of visual features of each client without loss. Data encrypted transmission costs extra response time. The prediction-based scheme uses the data on each client to train a global model to fit global visual features. Ideally, its accuracy can reach 100%. However, in real scenarios, models are often unable to fit completely, because of time constraints, data distributions (non-i.i.d. or i.i.d.), or federated learning characteristics. Quantitative results show that the time increases with the accuracy. A balance between the accuracy and the time can be achieved. In addition, as the amount of data required to create visual charts increases, obtaining more accurate results is more time-consuming. There is much capacity for further performance optimization. For example, the level-of-details scheme can be employed in visualizing a heatmap.

The query-based scheme can generate more accurate results than the prediction-based scheme, and has a better running performance. Thus, it is more feasible for time-critical situations.

7 CONCLUSION

This paper addresses an important aspect of decentralized visualization: privacy. The fundamental idea is to mimic the process of federated learning, and reformulate the visualization process with a new federated model. We propose two implementations: a query-based scheme, and a prediction-based scheme.

The query-based scheme directly encrypts accurate results, resulting in relative short response time than the prediction-based scheme. The
prediction-based scheme can be inefficient when the parameter number of the prediction model is large, i.e., more than ten thousands. We plan to reduce its computational complexity. Our work is the first attempt to tackle data privacy issues in a decentralized visualization framework. We hope that this work will inspire other researchers to further study the privacy preservation solution in distributed environment, which should be a meaningful direction for sensitive data collaborative analysis.

References

[1] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu. Hippocampal Databases. In Proceedings of the 28th International Conference on Very Large Databases, pp. 143–154, 2002.
[2] A. Arleo, O.-H. Kwon, and K.-L. Ma. GraphRay: Distributed Pathfinder Network Scaling. In Proceedings of the IEEE 7th Symposium on Large Data Analysis and Visualization, pp. 74–83, 2017.
[3] 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 preprint arXiv:1611.04482, 2016.
[4] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth. Practical Secure Aggregation for Privacy-Preserving Machine Learning. In Proceedings of the ACM SIGSAC Conference on Computer and Communications Security, pp. 1175–1191, 2017.
[5] K. W. Brodlie, D. A. Duce, J. R. Gallop, J. P. Walton, and J. D. Wood. Distributed and Collaborative Visualization. Computer Graphics Forum, 23(2):223–251, 2004.
[6] W. Chen, Z. Huang, F. Wu, M. Zhu, H. Guan, and R. Maciejewski. VAUD: A Visual Analysis Approach for Exploring Spatio-Temporal Urban Data. IEEE Transactions on Visualization and Computer Graphics, 24(9):2636–2648, 2017.
[7] J.-K. Chou, C. Bryan, J. Li, and K.-L. Ma. An Empirical Study on Perceptually Masking Privacy in Graph Visualizations. In Proceedings of the IEEE Symposium on Visualization for Cyber Security, pp. 1–8, 2018.
[8] J.-K. Chou, C. Bryan, and K.-L. Ma. Privacy Preserving Visualization for Social Network Data with Ontology Information. In Proceedings of the IEEE Pacific Visualization Symposium, pp. 11–20, 2017.
[9] J.-K. Chou, Y. Wang, and K.-L. Ma. Privacy Preserving Event Sequence Data Visualization using a Sankey Diagram-like Representation. In Proceedings of the SIGGRAPH ASIA Symposium on Visualization, pp. 1–8, 2016.
[10] J.-K. Chou, Y. Wang, and K.-L. Ma. Privacy Preserving Visualization: A Study on Event Sequence Data. Computer Graphics Forum, 38(1):340–355, 2019.
[11] A. Dasgupta and R. Kosara. Adaptive Privacy-Preserving Visualization Using Parallel Coordinates. IEEE Transactions on Visualization and Computer Graphics, 17(12):2241–2248, 2011.
[12] A. Dasgupta, R. Kosara, and M. Chen. Guess Me If You Can: A Visual Uncertainty Model for Transparent Evaluation of Disclosure Risks in Privacy-Preserving Data Visualization. In Proceedings of the IEEE Symposium on Visualization for Cyber Security, 2019.
[13] A. Dasgupta, E. Maguire, A. Abdul-Rahman, and M. Chen. Opportunities and Challenges for Privacy-Preserving Visualization of Electronic Health Record Data. In Proceedings of the IEEE VIS Workshop on Visualization of Electronic Health Records, 2014.
[14] R. C. Geyer, T. Klein, and M. Nabi. Differentially Private Federated Learning: A Client Level Perspective. arXiv preprint arXiv:1712.07557, 2017.
[15] L. Huang, A. L. Shea, H. Qian, A. Masurkar, H. Deng, and D. Liu. Patient clustering improves efficiency of federated machine learning to predict mortality and hospital stay time using distributed electronic medical records. Journal of Biomedical Informatics, 99:103291, 2019.
[16] L. Huang, Y. Yin, Z. Fu, S. Zhang, H. Deng, and D. Liu. LoAdaBoost: Loss-Based AdaBoost Federated Machine Learning on medical data. arXiv preprint arXiv:1811.12629, 2018.
[17] J. Konečný, H. B. McMahan, D. Ramage, and P. Richtárik. Federated Optimization: Distributed Machine Learning for On-Device Intelligence. arXiv preprint arXiv:1610.02527, 2016.
[18] J. Konečný, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon. Federated Learning: Strategies for Improving Communication Efficiency. arXiv preprint arXiv:1610.05492, 2016.
[19] S. Lefebvre and H. Hoppe. Perfect Spatial Hashing. ACM Transactions on Graphics, 25(3):579–588, 2006.
[20] N. Li, T. Li, and S. Venkatasubramanian. t-Closeness: Privacy Beyond k-Anonymity and l-Diversity. In Proceedings of the 33rd International Conference on Data Engineering, pp. 106–115, 2007.
[21] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith. Federated Learning: Challenges, Methods, and Future Directions. arXiv preprint arXiv:1908.07873, 2019.
[22] Y. Liu, A. Huang, Y. Luo, H. Huang, Y. Liu, Y. Chen, T. Chen, H. Yu, and Q. Yang. FedVision: An Online Vision Object Detection Platform Powered by Federated Learning. arXiv preprint arXiv:2001.06202, 2020.
[23] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramanian. l-Diversity: Privacy Beyond k-Anonymity. ACM Transactions on Knowledge Discovery from Data, 1(1):3, 2007.
[24] H. B. McMahan, D. Ramage, K. Talwar, and L. Zhang. Learning Differentially Private Recurrent Language Models. arXiv preprint arXiv:1710.06967, 2017.
[25] H. Mei, W. Chen, Y. Wei, Y. Hu, S. Zhou, B. Lin, Y. Zhao, and J. Xia. RSAtree: Distribution-Aware Data Representation of Large-Scale Tabular Datasets for Flexible Visual Query. IEEE Transactions on Visualization and Computer Graphics, 26(1):1161–1171, 2019.
[26] A. Meidiana, S.-H. Hong, J. Huang, P. Eades, and K.-L. Ma. Topology-Based Spectral Sparsification. In Proceedings of the IEEE 9th Symposium on Large Data Analysis and Visualization, pp. 73–82, 2019.
[27] J. Oksanen, C. Bergman, J. Sainio, and J. Westerholm. Methods for deriving and calibrating privacy-preserving heat maps from mobile sports tracking application data. Journal of Transport Geography, 48:135–144, 2015.
[28] S. M. Plis, A. D. Sarwate, D. Wood, C. Dieringer, D. Landis, C. Reed, S. R. Panta, J. A. Turner, J. M. Shoemaker, K. W. Carter, et al. COINSTAC: A Privacy Enabled Model and Prototype for Leveraging and Processing Decentralized Brain Imaging Data. Frontiers in Neuroscience, 10:365, 2016.
[29] B. Ramsundar and R. B. Zadeh. TensorFlow for Deep Learning: From Linear Regression to Reinforcement Learning, 2018.
[30] D. K. Saha, V. D. Calhoun, S. R. Panta, and S. M. Plis. See Without Looking: Joint Visualization of Sensitive Multi-site Datasets. In Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, pp. 2672–2679, 2017.
[31] M. Shih, S. Rizzi, J. Insley, T. Uram, V. Vishwanath, M. Herold, M. E. Papka, and K.-L. Ma. Parallel Distributed, GPU-Accelerated, Advanced Lighting Calculations for Large-Scale Volume Visualization. In Proceedings of the IEEE 6th Symposium on Large Data Analysis and Visualization, pp. 47–55, 2016.
[32] L. Sweeney. k-anonymity: A model for protecting privacy. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 10(5):557–570, 2002.
[33] X. Wang, W. Chen, J.-K. Chou, C. Bryan, H. Guan, W. Chen, R. Pan, and K.-L. Ma. GraphProtector: A Visual Interface for Employing and Assessing Multiple Privacy Preserving Graph Algorithms. IEEE Transactions on Visualization and Computer Graphics, 25(1):193–203, 2018.
[34] X. Wang, J.-K. Chou, W. Chen, H. Guan, W. Chen, T. Lao, and K.-L. Ma. A Utility-aware Visual Approach for Anonymizing Multi-attribute Tabular Data. IEEE Transactions on Visualization and Computer Graphics, 24(1):351–360, 2017.
[35] X. Wang, T. Gu, X. Luo, X. Cai, T. Lao, W. Chen, Y. Wu, J. Yu, and W. Chen. A User Study on the Capability of Three Geo-based Features in Analyzing and Locating Trajectories. IEEE Transactions on Intelligent Transportation Systems, 20(9):3375–3385, 2018.
[36] J. Xie, H. Yu, and K.-L. Ma. Visualizing Large 3D Geodesic Grid Data with Massively Distributed GPUs. In Proceedings of the IEEE 4th Symposium on Large Data Analysis and Visualization, pp. 3–10, 2014.
[37] Q. Yang, Y. Liu, T. Chen, and Y. Tong. Federated Machine Learning: Concept and Applications. ACM Transactions on Intelligent Systems and Technology, 10(2):1–19, 2019.
[38] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra. Federated Learning with Non-IID Data. arXiv preprint arXiv:1806.00582, 2018.
[39] Z. Zhou, L. Meng, C. Tang, Y. Zhao, Z. Guo, M. Hu, and W. Chen. Visual Abstraction of Large Scale Geospatial Origin-Destination Movement Data. IEEE Transactions on Visualization and Computer Graphics, 25(1):43–53, 2018.