Approximate k-Nearest Neighbor Query over Spatial Data Federation

Kaining Zhang[1], Yongxin Tong[1(✉)], Yexuan Shi[1], Yuxiang Zeng[1,2], Yi Xu[1], Lei Chen[2,3], Zimu Zhou[4], Ke Xu[1], Weifeng Lv[1], and Zhiming Zheng[1]

[1] State Key Laboratory of Software Development Environment, Beijing Advanced Innovation Center for Future Blockchain and Privacy Computing, School of Computer Science and Engineering and Institute of Artificial Intelligence, Beihang University, Beijing, China
{zhangkaining,yxtong,skyxuan,turf1013,xuy,kexu,lwf}@buaa.edu.cn, zzheng@pku.edu.cn

[2] Department of Computer Science and Engineering, HKUST, Hong Kong, China
leichen@cse.ust.hk

[3] Data Science and Analytics Thrust, HKUST (GZ), Guangzhou, China

[4] City University of Hong Kong, Hong Kong, China
zimuzhou@cityu.edu.hk

Abstract. Approximate nearest neighbor query is a fundamental spatial query widely applied in many real-world applications. In the big data era, there is an increasing demand to scale these queries over a spatial data federation, which consists of multiple data owners, each holding a private, disjoint partition of the entire spatial dataset. However, it is non-trivial to enable approximate k-nearest neighbor query over a spatial data federation. This is because stringent security constraints are often imposed to protect the sensitive, privately owned data partitions, whereas naively extending prior secure query processing solutions leads to high inefficiency (*e.g.*, 100 s per query). In this paper, we propose two novel algorithms for efficient and secure approximate k-nearest neighbor query over a spatial data federation. We theoretically analyze their communication cost and time complexity, and further prove their security guarantees and approximation bounds. Extensive experiments show that our algorithms outperform the state-of-the-art solutions with respect to the query efficiency and often yield a higher accuracy.

Keywords: Approximate nearest neighbor · Spatial data federation

1 Introduction

k-Nearest Neighbor (kNN) query is one of the most fundamental queries in spatial databases, which aims to find k spatial objects that are closest to a given location. The approximate solutions to kNN queries (*a.k.a.*, approximate kNN or ANN) are of particular research interest since they are better suited for real-time response over large-scale spatial data than the exact counterparts [6,7,12,13,25].

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2023
X. Wang et al. (Eds.): DASFAA 2023, LNCS 13943, pp. 351–368, 2023.
https://doi.org/10.1007/978-3-031-30637-2_23

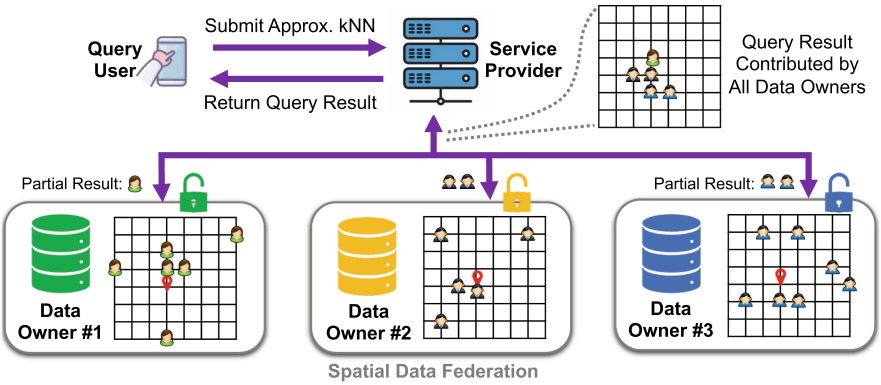


Fig. 1. Example of approximate kNN query over spatial data federation

There has been widespread adoption of approximate kNN in various application domains, such as transportation and map service, to name a few [12,13,21].

As the scale of real-world spatial applications continues to grow from region- to city- or even nation-wide, there has been a sharp urge to support approximate kNN queries over a *spatial data federation* [2,18,24]. A spatial data federation consists of multiple data owners who agree on the same schema and manage their own partition of the entire spatial dataset autonomously. Direct access to each raw data partition is prohibited, and secure queries over the federation are compulsory due to data sensitivity or commercial reasons. Take Amap [1], a major map service company in China, as an example. It offers a taxi-calling service via an integrated platform uniting dozens of taxi companies including Caocao, Shouqi, Yidao, *etc.* Prior to dispatching taxi orders, Amap may want to retrieve the k nearest drivers of a given passenger over the entire dataset of these companies via an approximate kNN query (see Fig. 1). The query should deliver high accuracy within a short time for satisfactory user experiences. It should also provide security guarantees for data partitions of the taxi companies by not leaking sensitive information such as locations during the query processing.

A naive solution is to extend general-purpose secure query schemes over data federations [2,24] to approximate kNN queries over a spatial data federation. However, they can be highly inefficient when processing approximate kNN queries. For instance, on the OpenStreetMap benchmark dataset [20,27] with 10k spatial objects and 6 data owners, the query delay of Conclave [24] is 100 s per query, which can hardly support real-time responses in taxi-calling applications. Other proposals (*e.g.*, SAQE [4] and Shrinkwrap [3]) are dedicated to specific relational queries and cannot be easily extended to spatial queries such as approximate kNN.

In this paper, we focus on efficient and secure solutions to approximate kNN query over spatial data federation ("*federated approximate kNN query*" for short). Specifically, we first design an approximation algorithm called *one-round*. The main idea is to (1) retrieve kNN over each data owner's local dataset in plain-

text, (2) securely estimate how much each local kNN will contribute to the final result, and (3) securely collect the final result from all data owners based on the estimated contribution ratio. Due to the possible errors introduced in *one-round*, we further propose an improved algorithm called *multi-round* which optimizes the contribution ratio estimation procedure for a better trade-off between query accuracy and efficiency. To summarize, we have made following contributions.

- To the best of our knowledge, we are the first to study approximate kNN queries over a spatial data federation.
- We design two novel algorithms for federated approximate kNN queries. We theoretically analyze their communication cost and time complexity, and prove their security guarantees and approximation bounds.
- We conduct extensive experiments on both synthetic and real datasets and compare our algorithms with the state-of-the-art solutions. Results show that our method is always more efficient and often achieves higher accuracy. For instance, our solution can be 2–4 orders of magnitude faster than SMCQL [2] and Conclave [24] on the real dataset.

The rest of this paper is organized as follows. We first formally define the federated approximate kNN query in Sect. 2. Next, we introduce our algorithms and present our theoretical analysis in Sect. 3. Finally, we conduct experiments in Sect. 4, review related studies in Sect. 5, and conclude in Sect. 6.

2 Problem Statement

In this section, we first introduce some basic concepts and then present the formal definition of the approximate kNN query over spatial data federation.

Definition 1 (Spatial Object). *A spatial object d_i is denoted by a location $l_{d_i} = (x_{d_i}, y_{d_i})$ on a 2-dimensional Euclidean space.*

Based on Definition 1, the distance between spatial objects d_1 and d_2 is computed by the Euclidean distance function $dis(l_{d_1}, l_{d_2}) = \sqrt{(x_{d_2} - x_{d_1})^2 + (y_{d_2} - y_{d_1})^2}$.

Definition 2 (Data Owner). *A data owner S_i owns a set D_i of spatial objects $d_1, d_2, \cdots, d_{|D_i|}$, where $|D_i|$ is the number of spatial objects in D_i.*

In real-world applications, a data owner is often a company that owns a certain amount of spatial data [19,22,26]. Moreover, these data owners may want to run spatial analytics jointly (over the union of their datasets) but are unwilling/unable to share their raw data directly. As a result, a *spatial data federation* can be established to provide unified analytic services with security guarantees.

Definition 3 (Spatial Data Federation). *A spatial data federation $F = \{S_1, S_2, ..., S_n\}$ units n data owners to provide spatial query services. A set D is used to represent all the spatial objects, i.e., $D_1 \bigcup D_2 \bigcup \ldots \bigcup D_n$, where a subset D_i of spatial objects is held by the data owner S_i.*

The query processing over a spatial data federation usually requires a joint computation across the data owners. Therefore, this federation also needs to guarantee the security of each data owner. As a result, existing work [2–4,24] usually assumes that each data owner can be a *semi-honest* attacker, which is defined by the following threat model.

Definition 4 (Semi-honest Threat Model). *Referring to existing work [2, 24], the data owners over spatial data federation are presumed to be semi-honest. A semi-honest attacker follows the query execution plan (e.g., correctly executing spatial queries over their owned dataset), but is also curious about the sensitive data (e.g., the locations of spatial objects) of the other data owners.*

Our problem is defined upon the (exact) k nearest neighbor (kNN) query.

Definition 5 ((Exact) kNN [13]). *Given a set D of spatial objects, a query location $l_q = (x_q, y_q)$, and a positive integer k, kNN retrieves a set $res^* \subseteq D$ of k spatial objects that are closest to the query location l_q, i.e., $\forall d \in res^*$ and $d' \in D - res^*$, $dis(l_d, l_q) \leq dis(l_{d'}, l_q)$.*

Based on these concepts, we introduce the approximate kNN query over spatial data federation ("*federated approximate kNN query*" for short) as follows.

Definition 6 (Federated Approximate kNN Query). *Given a spatial data federation $F = \{S_1, S_2, ..., S_n\}$, a query location $l_q = (x_q, y_q)$, and a positive integer k, a federated approximate kNN query $q(F, l_q, k)$ aims to find a set res of k spatial objects over the whole dataset D such that the **result accuracy** δ of this approximate answer can be maximized as much as possible*

$$\delta = \frac{|res \cap res^*|}{k}, \text{ where } res^* \text{ is the exact kNN} \tag{1}$$

while satisfying the following security constraint.

- ***Security constraint.** Under the semi-honest threat model, the query processing algorithm should ensure a data owner cannot infer any sensitive information about other data owners except for the query result.*

In Definition 6, the result accuracy defined in Eq. (1) is a widely-used metric [13,25] to assess the quality of the retrieved results. In the security requirement, the extra sensitive information can be the locations of any other owner's spatial objects, the ownership of the spatial objects, and the cardinality of query results over another owner's local dataset ("local result" as short).

Example 1. Three data owners S_1, S_2, and S_3 constitute a spatial data federation F. The locations of the spatial objects owned by S_1, S_2, and S_3 are listed in Table 1. A query user submits a kNN query $q(F, l_q, k)$ to this federation F, where the query location $l_q = (3, 3)$ and $k = 5$. Suppose the exact kNN is res^*= {(2,3),(3,2),(3,2),(3,3),(3,4)}, and the result found by an approximation algorithm is $res = \{(3, 3), (3, 2), (2, 3), (0, 4), (6, 4)\}$. The accuracy δ of this approximate result is $|res \cap res^*|/k = 3/5 = 60\%$.

3 Our Approximation Algorithms

In this section, we first introduce an overview of our solution in Sect. 3.1. Then, two basic and secure computation operations are presented in Sect. 3.2. Based on these basic operations, we propose two algorithms with different trade-offs for federated approximate kNN queries, *i.e.*, one-round algorithm (Sect. 3.3) and multi-round algorithm (Sect. 3.4). The former is faster than the latter, while the latter is more accurate than the former. Finally, we prove the approximation guarantees of both algorithms in Sect. 3.5.

Table 1. Locations of spatial objects owned by S_1, S_2, and S_3

S_1

ID	Location
1	(0,4)
2	(3,0)
3	(3,4)
4	(3,5)
5	(4,4)
6	(7,6)

S_2

ID	Location
1	(1,6)
2	(3,2)
3	(3,3)
4	(3,6)
5	(4,5)

S_3

ID	Location
1	(1,4)
2	(2,3)
3	(3,2)
4	(4,2)
5	(4,6)
6	(6,4)
7	(7,3)

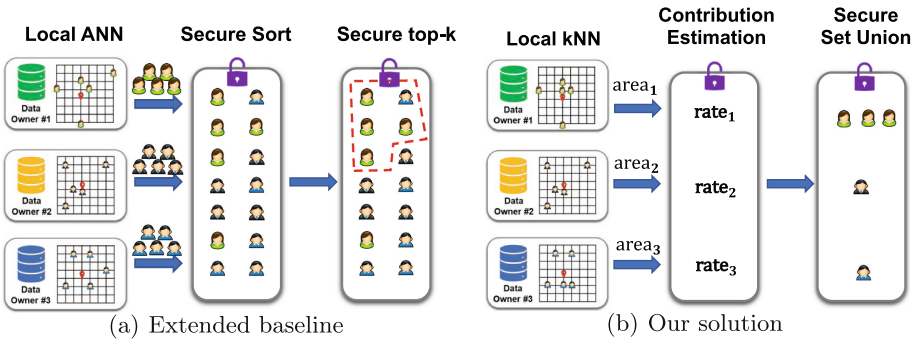


Fig. 2. Comparison of the extended baseline and our solution

3.1 Overview

Limitation of Extended Baseline. As shown in Fig. 2(a), existing general-purpose solutions (*e.g.*, Conclave [24]) can be extended as a baseline. In general, it first asks each data owner to perform an approximate kNN (*i.e.*, ANN) over its local dataset, and then uses secure sort and secure top-k to compute the final

result. From experiments in Sect. 4, we have two observations on the limitation of baselines. (1) The *inefficiency* is mainly caused by the secure operations, which are complicated and time-consuming. (2) The errors of totally $O(n)$ ANNs may exacerbate the error of the final answer. Thus, it motivates us to design a new query processing solution that requires *more light-weight* secure operations and *fewer inaccurate approximations*.

Overview of Our Solution. To overcome the limitation, the *main idea* of our solution is illustrated in Fig. 2(b). Specifically, each silo first executes exact kNN (instead of ANN) to produce k candidate objects. Next, an accurate method is devised to approximately estimate the contribution ratio to the final answer in each silo. This method relies on secure summation, which is one of the simplest secure operations. Based on this ratio, a secure set union is used to collect the final answer from all silos. By contrast, our solution only involves *light-weight secure operations* and *one approximation operation*.

3.2 Preliminary

Our algorithm is designed based on two primitive operations as follows, *i.e.*, secure summation and secure set union.

Secure Summation Operation [5]. The secure summation operation gets the sum of the private values held by multiple data owners while satisfying the security constraint. For example, when data owners S_1-S_3 hold $value_1$-$value_3$ respectively, a secure summation operation works as follows. Each pair of S_i and S_j $(i < j)$ negotiates a random number $sc_{i,j}$ secretly in advance. When a secure summation request is submitted, each data owner S_i perturbs $value_i$ as $value_i^{'} = value_i + \sum_{j\in[1,3],i<j} sc_{i,j} - \sum_{j\in[1,3],i>j} sc_{j,i}$. Next, S_i sends $value_i^{'}$ to the spatial data federation. Finally, the spatial data federation adds up $value_1^{'}, value_2^{'}$, and $value_3^{'}$ in plain text as the final result, which equals to $value_1 + value_2 + value_3$.

Secure Set Union Operation [10]. The secure set union operation gets the union of the private sets held by multiple data owners while satisfying the security constraint. For example, suppose 3 data owners S_1-S_3 hold datasets $set_1 - set_3$, respectively. First, S_1, S_2 and S_3 generate random sets $rset_1, rset_2$, and $rset_3$, respectively. Then, S_1 sends $tset_1 = set_1 \bigcup rset_1$ to S_2, S_2 sends $tset_2 = tset_1 \bigcup set_2 \bigcup rset_2$ to S_3, and S_3 sends $tset_3 = tset_2 \bigcup set_3 \bigcup rset_3$ to S_1. Next, S_1 sends $tset_1 = tset_3 - rset_1$ to S_2, S_2 sends $tset_2 = tset_1 - rset_2$ to S_3, and S_3 sends $tset_3 = tset_2 - rset_3$ to S_1. Finally, S_1 submits $tset_3$ to the spatial data federation, which equals the union of set_1, set_2, and set_3.

3.3 One-Round Algorithm

Main Idea. Each data owner first performs an exact kNN query over its local dataset to get the local kNN set. Then the spatial data federation estimates the contribution ratio of each data owner's local kNN to the final answer. The larger

the ratio is, the more objects the data owner's local kNN will contribute to the final answer.

Algorithm Details. Algorithm 1 illustrates the detailed procedure. In lines 1–2, each data owner executes its local kNN. In lines 3–5, we use a radius r_i to denote the kth nearest distance to the query location l_q and $area_i$ to denote the area of a circle with the radius r_i to the center l_q. Then, the contribution ratio $rate_i$ of the data owner S_i's local kNN to the final result is inversely proportional to $area_i$. To compute $rate_i$, we first compute $sum = \sum_{i=1}^{n} \frac{1}{area_i}$ by the secure summation operation $SecureSum()$, and then each data owner S_i can calculate $rate_i$ as $\frac{1/area_i}{sum}$. Accordingly, S_i provides the top $(rate_i \times k)$NN as the partial result res_i. Finally, the spatial data federation performs the secure union $SecureUnion()$ among these partial results $res_1, res_2, \cdots, res_n$ to collect the final result.

Example 2. Back to Example 1. The values of some intermediate variables are shown in Table 2. Data owners calculate the kth nearest distance to l_q based on their local kNN first. For example, since the location of the kth nearest neighbor of S_1 is (0,4), the kth nearest distance to l_q can be calculated as $r_1 = \sqrt{(0-3)^2 + (4-3)^2} = \sqrt{10}$. Similarly, we have $r_2 = \sqrt{13}$ and $r_3 = \sqrt{10}$. Therefore, $area_1 = 10\pi, area_2 = 13\pi$, and $area_3 = 10\pi$. By the secure summation, we have $sum = 18/65\pi$. Then, $rate_1$ can be computed as $area_1/sum = 13/36$. $rate_2$ and $rate_3$ can be calculated similarly. It indicates the data owners S_1-S_3 would offer their 2NN, NN and 2NN as the partial result. By using a secure union, the final result by Algorithm 1 is $res = \{(3,4),(4,4),(3,3),(3,2),(2,3)\}$.

Algorithm 1: One-round algorithm

Input: spatial object sets $D_1, D_2, \cdots, D_n$, the query request $q(F, l_q, k)$
Output: query result

1 **for** $i \in [1, n]$ **do**
2 $nn_i \leftarrow S_i$'s local kNN over D_i
3 **for** $i \in [1, n]$ **do**
4 $r_i \leftarrow \max_{j \in [1,k]} dis(l_{nn_i[j]}, l_q)$
5 $area_i \leftarrow \pi(r_i)^2$
6 $sum \leftarrow SecureSum(\frac{1}{area_1}, \frac{1}{area_2}, \cdots, \frac{1}{area_n})$
7 **for** $i \in [1, n]$ **do**
8 $rate_i \leftarrow \frac{1/area_i}{sum}$
9 $num_i \leftarrow rate_i \times k$
10 $res_i \leftarrow$ the top num_i-NN in nn_i
11 **return** $res \leftarrow SecureSetUnion(res_1, res_2, \cdots, res_n)$

Table 2. Values of some intermediate variables in Example 2

	S_1	S_2	S_3
ID of local kNN	{3,5,4,2,1}	{3,2,5,4,1}	{2,3,4,1,5}
r	$\sqrt{10}$	$\sqrt{13}$	$\sqrt{10}$
$area$	10π	13π	10π
$rate$	13/36	5/18	13/36
num	2	1	2
ID of res	{3,5}	{3}	{2,3}

Communication Complexity. In Algorithm 1, the communication mainly occurs in the secure summation and secure set union. Since each data owner needs to send information to other $n-1$ data owners, the communication complexity of secure summation is $O(n^2)$. In the secure set union, the communication complexity of interactions among these n data owners is $O(n)$. Thus, the communication complexity of Algorithm 1 is $O(n^2)$.

Time Complexity. The time complexity of lines 1–5 is $O(\log m)$, where m is the maximum number of objects owned by a data owner. In line 6, the time complexity of the secure summation is $O(n)$, where n is the number of data owners. The time complexity of lines 7–10 is $O(1)$. Note that although the total communication cost is $O(n^2)$, the time complexity of communication is still bounded by $O(n + logm)$ for each data owner. In line 11, the time complexity of the secure set union is $O(n)$. Thus, the time complexity of Algorithm 1 is $O(n + \log m)$.

Security Proof. We prove the security of Algorithm 1 in Lemma 1 by the composition lemma [8] in cryptography theory.

Fact 1 (Composition Lemma [8]). *Given a secure protocol $\phi(y|x)$ that can securely compute y based on plain-text query x, the operation y can be securely computed by executing protocol $\phi(y|x)$ but substitutes every plain-text query x with a secure protocol $\phi(x)$.*

Lemma 1. *Algorithm 1 is secure against the semi-honest threat model in Definition 4.*

Proof. Let $\phi(x)$ be the secure summation (line 6) and $\phi(y|x)$ be the calculation procedure in lines 1–10 based on the plain-text summation. It can be observed that lines 1–5 and lines 7–10 do not involve interactions across data owners, which means each data owner can execute lines 1–10 independently when the summation result is known. Based on the composition lemma [8], the computations in lines 1–10 are also secure. As the secure set union (line 11) is secure, the whole calculation procedure of Algorithm 1 is secure.

3.4 Multi-round Algorithm

Motivation. The one-round algorithm roughly estimates the contribution ratio of data owners' local kNN to the query result, which may result in unsatisfied accuracy (see our experiments in Sect. 4). Thus, we present a multi-round algorithm that slightly sacrifices efficiency for an improvement in the accuracy.

Main Idea. To further improve the query *accuracy*, we use a more fine-grained approach to estimate the contribution of each data owner to the final query result. Specifically, we divide the query processing procedure into multiple rounds and each round contributes a part of the final result. For example, when the number of rounds is W, k/W nearest neighbors (k/WNN) of kNN will be determined in each round. In this way, the estimation of the contribution ratio in each round can be more accurate than that in Algorithm 1. Besides, those objects, which have been determined, will not appear in subsequent rounds.

Algorithm Details. Algorithm 2 illustrates the detailed procedure. In lines 1–3, each data owner S_i maintains a set $UnadSet_i$ that contains candidate objects and a set $AdSet_i$ that contains selected objects. The local kNN of S_i is put into $UnadSet_i$ as initialization. Different from Algorithm 1, Algorithm 2 takes W ($W > 1$) rounds to compute the final results in lines 4–8 and each round decides k/W nearest neighbors (k/WNN). Specifically, for each round w, we estimate the contribution ratio of $UnadSet_1, UnadSet_2, \cdots, UnadSet_n$ to the k/WNN in this round, which is similar to the one-round algorithm. Note that in the w ($w \geq 2$)th round, we fine-tune the $area_i$ as $\pi[r_i^2 - (r_i^{'})^2]$, where r_i is defined as line 4 of Algorithm 1 and r_i' is the cached value of r_i in the $(w-1)$th round. The selected objects of S_i are removed from $UnadSet_i$ to $AdSet_i$. In line 9, the final result is calculated by a secure set union over $AdSet_1, AdSet_2, \cdots, AdSet_n$.

Example 3. Back to Example 1. We aim to find 2NN and 3NN in the 1st and 2nd rounds, respectively. The values of the intermediate variables of the 1st and 2nd rounds are listed in Table 3 and Table 4, respectively. By merging $AdSet_1, \cdots, AdSet_n$, we have res={(3,4),(3,3),(3,2),(2,3),(3,2)}, whose query accuracy is 100%.

Communication and Time Complexity. Algorithm 2 involves W contribution estimation and 1 secure set union operation. Since the number W of rounds is a constant parameter, the communication complexity and time complexity of Algorithm 2 are same as those of Algorithm 1, *i.e.*, $O(n^2)$ and $O(n + \log m)$, respectively.

Algorithm 2: Multi-round algorithm

Input: spatial object sets $D_1, D_2, \cdots, D_n$, the query request $q(F, l_q, k)$, the query round W
Output: query result

1 **for** $i \in [1, n]$ **do**
2 $\quad UnadSet_i \leftarrow S_i$'s local kNN over D_i
3 $\quad AdSet_i \leftarrow \phi$
4 **for** $w \in [1, W]$ **do**
5 $\quad$ **for** $i \in [1, n]$ **do**
6 $\quad\quad NewadSet_i \leftarrow res_i$ found by Algorithm 1 whose input is $\{UnadSet_1, UnadSet_2, \cdots, UnadSet_n, q(F, l_q, k/W)\}$
7 $\quad\quad AdSet_i \leftarrow AdSet_i \cup NewadSet_i$
8 $\quad\quad UnadSet_i \leftarrow UnadSet_i - NewadSet_i$
9 **return** $res \leftarrow SecureUnion(AdSet_1, AdSet_2, \cdots, AdSet_n)$

Table 3. Values of some intermediate variables of the 1st round in Example 3

	S_1	S_2	S_3
ID of objects in $NewadSet$	$\emptyset$	{3}	{2}
ID of objects in $AdSet$	$\emptyset$	{3}	{2}
ID of objects in $UnadSet$	{1,2,3,4,5,6}	{1,2,4,5}	{1,3,4,5,6,7}

Security Proof. We prove the security of Algorithm 2 in Lemma 1 by the composition lemma [8] (*i.e.*, Fact 1 in Sect. 3.3).

Lemma 2. *Algorithm 2 is secure against the semi-honest threat model in Definition 4.*

Proof. Let $\phi(x)$ be the calculation procedure in line 6 and $\phi(y|x)$ be the calculation procedure in lines 1–8 based on the plain-text calculation procedure in line 6. It can be observed that lines 1–5 and lines 7–8 do not involve interactions across data owners, which means each data owner can execute lines 1–8 independently when the result of line 6 is known. Since the calculation procedure in line 6 is proved to be secure in Sect. 3.3, the computations in lines 1–8 are also secure based on the composition lemma [8]. As the secure set union (line 9) is secure, the whole calculation procedure of Algorithm 2 is secure.

3.5 Approximation Guarantees of both Algorithms

The following theorem proves the approximation guarantees of our one-round algorithm (when $W = 1$) and multi-round algorithm (when $W > 1$).

Theorem 1. *The accuracy δ of the approximate kNN query result satisfies*

$$\Pr(\delta < 1 - \varepsilon) \leq 2\exp(\frac{-2W\varepsilon^2}{n}),$$

Table 4. Values of some intermediate variables of the 2nd round in Example 3

	S_1	S_2	S_3
ID of $NewadSet$	{3}	{2}	{3}
ID of $AdSet$	{3}	{3,2}	{2,3}
ID of $UnadoptedSet$	{1,2,4,5,6}	{1,4,5}	{1,4,5,6,7}

where W,n, and k represent the query round, the number of data owners, and the number of objects to be queried respectively.

Proof. Assume that the spatial objects owned by data owners $S_1, S_2, \cdots, S_n$ roughly follow the uniform distribution in each round. Denote the density of objects in S_i as ρ_i. We have $\mathbb{E}[\rho_i] = \frac{k/W}{area_i}$, where $area_i$ is the circle's area of local $\frac{k}{W}$NN in data owner S_i. Thus, the expected area of the circle of global $\frac{k}{W}$NN is

$$\mathbb{E}[area] = \frac{k/W}{\sum_{i=1}^{n} \mathbb{E}[\rho_i]} = \frac{1}{\sum_{i=1}^{n} \frac{1}{area_i}}.$$

Denote the contribution ratio of the data owner S_i as $rate_i$. We have

$$\mathbb{E}[rate_i] = \frac{\frac{1}{area_i}}{\sum_{i=1}^{n} \frac{1}{area_i}}.$$

Then, by applying the Hoeffding's inequality [23], we can derive that for the data federation F,

$$\Pr(\sum_{j=1}^{W}\sum_{i=1}^{n} |rate_i^j - \mathbb{E}[rate_i^j]| > \varepsilon) \leq 2\exp(\frac{-2\epsilon^2}{Wn}).$$

And the accuracy δ of the approximate kNN query result is

$$\delta = 1 - \frac{(\sum_{j=1}^{W}\sum_{i=1}^{n} |rate_i^j - \mathbb{E}[rate_i^j]|) \cdot \frac{k}{W}}{k}.$$

Therefore, the accuracy δ of the approximate result satisfies

$$\Pr(\delta < 1 - \varepsilon) \leq 2\exp(\frac{-2W\varepsilon^2}{n}).$$

4 Experimental Evaluation

This section presents the experimental setup in Sect. 4.1 and results in Sect. 4.2.

4.1 Experimental Setup

Datasets. Both real dataset (MBJ) and synthetic dataset (OSM) are used in our experimental evaluation.

– **Multi-company Spatial Data in Beijing (MBJ).** We randomly select 10^6 pieces of spatial data records from the original dataset comprising 1,029,081 pieces of records from 10 companies in Beijing. Each company is regarded as a data owner.

Table 5. Parameter settings

Parameter	Setting
#(Nearest neighbors) k	4, 8, **16**, 32, 64
#(Data onwers) n	2, 4, **6**, 8, 10
Data size $\|D\|$	10^4, 10^5, $\mathbf{10^6}$, 10^7, 10^8

– **OpenStreetMap (OSM).** This dataset is widely used in large-scale spatial data systems [20,27]. We randomly select 10^4, 10^5, 10^6, 10^7, and 10^8 pieces of location records and assign each record a random data owner number so that data owners have the same number of objects.

Parameter Settings. Referring to the parameter settings in [9,20], we vary k, $|D|$, and n from 4 to 64, 10^4 to 10^8, and 2 to 10, respectively. The parameter settings are summarized in Table 5, in which default parameters are in bold.

Compared Algorithms. We compare the performance of our algorithms and the following baselines (*i.e.*, SMCQL [2], which is developed by ObliVM [15], and Conclave [24], which is developed by MP-SPDZ [11]).

For our proposed algorithms, we use OR and MR to denote the one-round algorithm (*i.e.*, Algorithm 1) and multi-round algorithm (*i.e.*, Algorithm 2), respectively.

For the compared baselines, we make extensions on supporting approximate kNN queries as follows. Each data owner calculates its local approximate kNN by a seminal indexing approach, the ANN library [16]. Then the final kNN is derived by a secure sorting over the union of local kNN computed by data owners. Notice that SMCQL only supports queries over a federation of two data owners.

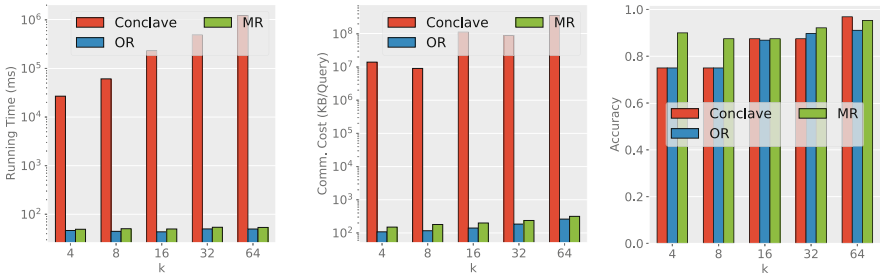
Evaluation Metrics. We use *running time* and *communication cost* to test the efficiency and result accuracy ("*accuracy*" as short) to test the effectiveness.

Experimental Environment. The experiments are conducted on 10 docker containers (*i.e.*, up to 10 data owners), each with 16 AMD Ryzen 3.4GHz CPU cores. Each docker container can be regarded as a data owner. The experimental results are the average of 50 repetitions.

4.2 Experimental Results

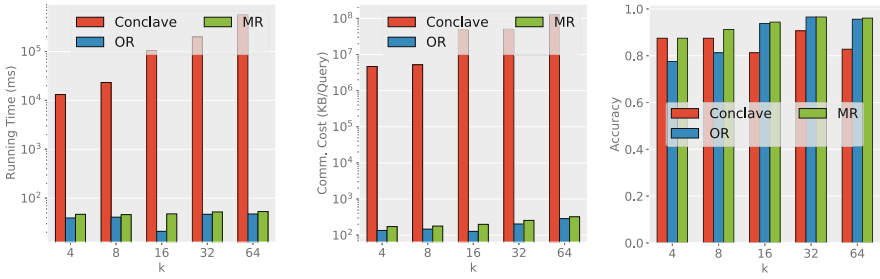
Note that the results of SMCQL are only available when n is 2, since its adopted security technique, ObliVM [15], is only applicable for 2 data owners.

Effect of k. Figure 3-4 illustrate the results when varying k on MBJ and OSM, respectively. We can first observe that our solution always achieves a better efficiency than Conclave. For example, on the real dataset (MBJ), Conclave can be 4 orders of magnitude slower than our one-round (OR) and multi-round (MR) algorithms, and its communication cost is at least 5 orders of magnitude higher than that of our solution. In terms of result accuracy, our solution is often better than Conclave. For instance, the accuracy of MR is up to 13% higher than that of Conclave. The error of Conclave is mainly caused by the accumulative errors of its local approximate kNN queries ($O(n)$ in total). By contrast, the error of our solution is only produced in one operation, *i.e.*, contribution estimation. Moreover, the results also show that MR can effectively improve the accuracy with only marginal sacrifice in the efficiency.



(a) Running time (b) Communication cost (c) Accuracy

Fig. 3. Results when varying k on the MBJ dataset



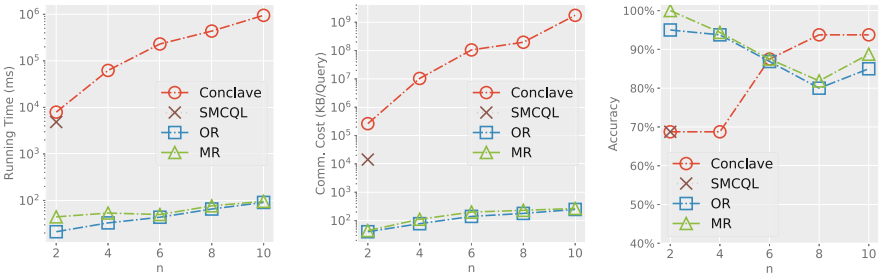
(a) Running time (b) Communication cost (c) Accuracy

Fig. 4. Results when varying k on the OSM dataset

Effect of n. Figure 5-6 present the results when varying n on MBJ and OSM, respectively. In terms of efficiency, the running time and communication cost of

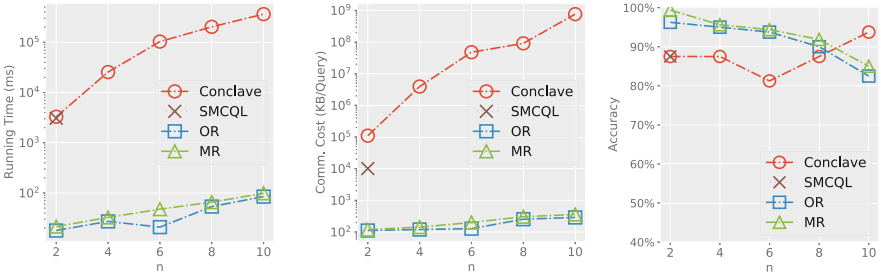
our algorithms are still significantly lower than those of SMCQL and Conclave. For instance, OR and MR are up to 4 orders of magnitude faster than the baselines. The communication cost of Conclave is up to 1.6 TB per query, while that of ours is below 0.27 MB. Besides, the running time and communication cost of all algorithms show an upward trend, when the number n of data owners is increasing. This is because (1) the secure operations are efficiency bottleneck of the baselines and (2) the efficiency of secure operations is sensitive to n. In terms of effectiveness, the result accuracy of our algorithms is higher than that of baselines in most cases. For example, the accuracy of OR and MR is up to 26.25% and 31.25% higher than that of SMCQL and Conclave, respectively.

Effect of $|D|$. Figure 7 shows the results when varying the data size $|D|$ on OSM. When varying the data size, the running time and communication cost of our algorithms are constantly lower than those of Conclave. For instance, OR is usually 3 orders of magnitude faster than Conclave. Between our algorithms, MR is slightly less efficient than OR. Overall, the results show that our solution is much more scalable than the state-of-the-art. As for the result accuracy, our algorithms are more stable than Conclave. Specifically, the accuracy of OR and MR is usually above 83% and 90%, respectively. By contrast, the accuracy of the baseline can be as low as 75%.



(a) Running time (b) Communication cost (c) Accuracy

Fig. 5. Results when varying n on the MBJ dataset



(a) Running time (b) Communication cost (c) Accuracy

Fig. 6. Results when varying n on the OSM dataset

Summary of results. The experimental findings are summarized as follows.

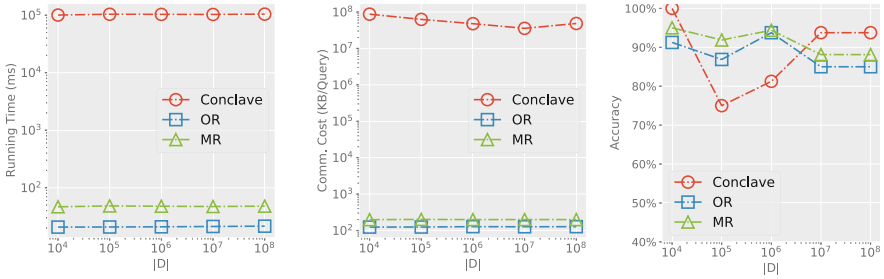
(1) Our solution is notably more efficient than the baselines in query efficiency. It can be 2–4 orders of magnitude faster and take 2–6 orders of magnitude lower communication cost than that of SMCQL and Conclave.
(2) The result accuracy of our solution is often higher than that of baselines. For example, the result accuracy of our solution can be up to 31.25% higher than the accuracy of SMCQL and Conclave.
(3) Our multi-round algorithm can effectively improve the accuracy of one-round algorithm by sacrificing only a little efficiency.

5 Related Work

Our work is related to the domains of *Querying over Data Federation* and *Approximate kNN Query*.

Querying over Data Federation. Existing algorithms perform queries over a data federation by utilizing the Secure Multiparty Computation (SMC) [8] technique, which protects the input and intermediate data from leaking. After receiving a query request submitted by the user, the data federation parses the query into a series of secure and plain-text operations and coordinates multiple data owners to get query results based on the union of their local datasets.

Hu-Fu [17,20] is a spatial data federation system that supports spatial queries with exact results. However, unlike SMCQL [2] and Conclave [24], it is non-trivial to extend Hu-Fu to support approximate kNN queries. This is because (1) the query rewriter of Hu-Fu decomposes a kNN query into a series of range counting queries and one range query (instead of several kNN queries on each local dataset) and (2) the results of range counting and range query here must be accurate to ensure the final answer has exactly k objects. Other data federation systems, *SAQE* [4] and *Shrinkwrap* [3], consider the trade-offs among result accuracy, query efficiency, and differential privacy. For example, *SAQE* improves the query efficiency by sampling and protects the query result by differential



(a) Running time (b) Communication cost (c) Accuracy

Fig. 7. Results when varying $|D|$ on the OSM dataset

privacy, while *Shrinkwrap* protects the intermediate results by using differential privacy. However, since differential privacy is not the main concern of our work, their solution cannot be extended to our problem setting.

Some recent work also studies graph (data) federation [28]. A typical application of the graph federation is multi-modal route planning [14] over multiple transportation networks. To solve this problem, Li *et al.* [14] proposed a novel solution to achieve very high efficiency and low communication overhead.

Approximate kNN Query. Existing solutions to approximate kNN queries can be divided into three categories: *tree-based*, *graph-based*, and *hash-based* solutions. In *tree-based solutions*, the main idea is to divide the entire space into multiple disjoint areas, and kNN often falls in the area that contains or is adjacent to the query location [13]. For *graph-based solutions*, the core idea is to construct a proximity graph based on the neighbor relationship between objects and search for the "nearest neighbors" first [25]. As for *hash-based solutions*, the basic idea is to map data objects to lower-dimensional hash values. Then, the closer the distance between two objects is, the higher the probability of being mapped to the same hash value [6]. These algorithms do not consider data security/privacy and hence cannot be used as the solutions to our federated approximate kNN query problem. However, these algorithms can be used in querying the local dataset in each data owner.

Some existing work [7,12], which considers data privacy, is mostly studied under the scenario of outsourced databases. Although these solutions [7,12] can protect security by searchable asymmetric encryption schemes, their application scenario is quite different from ours (*i.e.*, a data federation). For example, an outsourced database assumes the raw data of one data owner is encrypted and stored in the third party (*e.g.*, a cloud server). By contrast, in a data federation, the local dataset of each data owner does not need to be encrypted and stored in a third party. Thus, queries can be securely executed in plaintext over each local dataset, which is quite different from the scenario of outsourced databases.

6 Conclusion

This paper investigates efficient and secure approximate kNN query over a spatial data federation. We propose two secure algorithms with low communication cost and time complexity. We also prove that their approximation guarantees (*i.e.*, the accuracy of the query results) have non-trivial bounds. The experimental results on both synthetic and real datasets show that the query efficiency of our algorithms is significantly better than that of state-of-the-arts by a large margin. Results also show that our algorithms can often achieve a higher result accuracy.

Acknowledgements. We are grateful to anonymous reviewers for their constructive comments. This work is partially supported by the National Science Foundation of China (NSFC) under Grant No. U21A20516 and 62076017, the Beihang University Basic Research Funding No. YWF-22-L-531, the Funding No. 22-TQ23-14-ZD-01-001 and WeBank Scholars Program. Lei Chen's work is partially supported by National

Science Foundation of China (NSFC) under Grant No. U22B2060, the Hong Kong RGC GRF Project 16213620, RIF Project R6020-19, AOE Project AoE/E-603/18, Theme-based project TRS T41-603/20R, China NSFC No. 61729201, Guangdong Basic and Applied Basic Research Foundation 2019B151530001, Hong Kong ITC ITF grants MHX/078/21 and PRP/004/22FX, Microsoft Research Asia Collaborative Research Grant and HKUST-Webank joint research lab grants.

References

1. Amap: https://lbs.amap.com/. Accessed 30 Jan 2023
2. Bater, J., Elliott, G., Eggen, C., Goel, S., Kho, A.N., Rogers, J.: SMCQL: secure query processing for private data networks. PVLDB **10**(6), 673–684 (2017)
3. Bater, J., He, X., Ehrich, W., Machanavajjhala, A., Rogers, J.: Shrinkwrap: efficient SQL query processing in differentially private data federations. PVLDB **12**(3), 307–320 (2018)
4. Bater, J., Park, Y., He, X., Wang, X., Rogers, J.: SAQE: practical privacy-preserving approximate query processing for data federations. PVLDB **13**(11), 2691–2705 (2020)
5. Bonawitz, K., et al.: Practical secure aggregation for privacy-preserving machine learning. In: CCS, pp. 1175–1191 (2017)
6. Cai, D.: A revisit of hashing algorithms for approximate nearest neighbor search. IEEE Trans. Knowl. Data Eng. **33**(6), 2337–2348 (2021)
7. Choi, S., Ghinita, G., Lim, H., Bertino, E.: Secure knn query processing in untrusted cloud environments. IEEE Trans. Knowl. Data Eng. **26**(11), 2818–2831 (2014)
8. Evans, D., Kolesnikov, V., Rosulek, M.: A pragmatic introduction to secure multi-party computation. Found. Trends Priv. Secur. **2**(2–3), 70–246 (2018)
9. Gao, D., Tong, Y., She, J., Song, T., Chen, L., Xu, K.: Top-k team recommendation in spatial crowdsourcing. In: WAIM, pp. 191–204 (2016)
10. Jurczyk, P., Xiong, L.: Information sharing across private databases: secure union revisited. In: SocialCom/PASSAT, pp. 996–1003 (2011)
11. Keller, M.: MP-SPDZ: a versatile framework for multi-party computation. In: CCS, pp. 1575–1590 (2020)
12. Lei, X., Liu, A.X., Li, R.: Secure KNN queries over encrypted data: dimensionality is not always a curse. In: ICDE, pp. 231–234 (2017)
13. Li, W., Zhang, Y., Sun, Y., Wang, W., Li, M., Zhang, W., Lin, X.: Approximate nearest neighbor search on high dimensional data - experiments, analyses, and improvement. IEEE Trans. Knowl. Data Eng. **32**(8), 1475–1488 (2020)
14. Li, Y., Yuan, Y., Wang, Y., Lian, X., Ma, Y., Wang, G.: Distributed multimodal path queries. IEEE Trans. Knowl. Data Eng. **34**(7), 3196–3210 (2022)
15. Liu, C., Wang, X.S., Nayak, K., Huang, Y., Shi, E.: Oblivm: a programming framework for secure computation. In: S & P, pp. 359–376 (2015)
16. Mount, D.M., Arya, S.: Ann library. http://www.cs.umd.edu/mount/ANN/. Accessed 30 Jan 2023
17. Pan, X., et al.: Hu-fu: a data federation system for secure spatial queries. PVLDB **15**(12), 3582–3585 (2022)
18. Shi, Y., Tong, Y., Zeng, Y., Zhou, Z., Ding, B., Chen, L.: Efficient approximate range aggregation over large-scale spatial data federation. IEEE Trans. Knowl. Data Eng. **35**(1), 418–430 (2023)

19. Tao, Q., Zeng, Y., Zhou, Z., Tong, Y., Chen, L., Xu, K.: Multi-worker-aware task planning in real-time spatial crowdsourcing. In: DASFAA, pp. 301–317 (2018)
20. Tong, Y., et al.: Hu-fu: efficient and secure spatial queries over data federation. PVLDB **15**(6), 1159–1172 (2022)
21. Tong, Y., Zeng, Y., Zhou, Z., Chen, L., Xu, K.: Unified route planning for shared mobility: an insertion-based framework. ACM Trans. Database Syst. **47**(1), 2:1-2:48 (2022)
22. Tong, Y., Zhou, Z., Zeng, Y., Chen, L., Shahabi, C.: Spatial crowdsourcing: a survey. VLDB J. **29**(1), 217–250 (2020)
23. Vershynin, R.: High-Dimensional Probability: An Introduction with Applications in Data Science. Cambridge University Press, Cambridge (2018)
24. Volgushev, N., Schwarzkopf, M., Getchell, B., Varia, M., Lapets, A., Bestavros, A.: Conclave: secure multi-party computation on big data. In: EuroSys, pp. 3:1–3:18 (2019)
25. Wang, M., Xu, X., Yue, Q., Wang, Y.: A comprehensive survey and experimental comparison of graph-based approximate nearest neighbor search. PVLDB **14**(11), 1964–1978 (2021)
26. Wang, Y., et al.: Fed-LTD: towards cross-platform ride hailing via federated learning to dispatch. In: KDD, pp. 4079–4089 (2022)
27. Xie, D., Li, F., Yao, B., Li, G., Zhou, L., Guo, M.: Simba: efficient in-memory spatial analytics. In: SIGMOD, pp. 1071–1085 (2016)
28. Yuan, Y., Ma, D., Wen, Z., Zhang, Z., Wang, G.: Subgraph matching over graph federation. PVLDB **15**(3), 437–450 (2021)