PrivNN: A Private and Efficient Framework for Spatial Nearest Neighbor Query Processing

Zechun Cao*, Brian Kishiyama and Jeong Yang

Texas A&M University-San Antonio, One University Way, San Antonio, 78224, Texas, USA

ARTICLE INFO

Keywords:
Privacy
Security
Privacy-preserving computation
Location-based services
Nearest neighbor search
Spatial query processing

ABSTRACT

A common query type in location-based services (LBS) is finding the nearest neighbor (NN) of a given query object. However, the exact location of the query object is often sensitive information, posing significant privacy risks if the LBS server is untrusted or compromised. In this paper, we propose PrivNN, a novel spatial NN query processing framework that allows users to perform exact NN queries without revealing their location. Our framework introduces a novel spatial NN search algorithm, Dynamic Hierarchical Voronoi Overlay (DHVO), which efficiently finds the nearest neighbor by iteratively refining the search region using multi-granular Voronoi diagrams. We also present a client–server communication protocol that enables the server to respond to encrypted spatial NN queries by employing homomorphic encryption. We rigorously prove the correctness of our algorithm, analyze the theoretical properties of our framework, and demonstrate its strong security and robust privacy bounds. We implement and evaluate PrivNN on real-world spatial datasets, showing that it substantially reduces computational and communication overhead while remaining practical for private NN search in LBS applications.

1. Introduction

The widespread adoption of GPS-enabled devices, such as mobile phones and smartwatches, coupled with the prevalence of cloud computing, has significantly driven the growth of location-based services (LBSs), making them integral to our daily lives. LBS providers own and host large spatial databases in the cloud, utilizing users' geospatial locations at various levels of granularity to deliver personalized query results. For instance, Google only needs the user's city or region to provide weather updates or regional news, whereas finer granularity at the neighborhood or zip code level is necessary for services like finding nearby restaurants or attractions. While these services offer convenience, they also raise significant privacy concerns regarding users' location data [1–4].

Privacy concerns over users' location data are directly related to data granularity because the level of detail in location data can significantly impact the potential for misuse and the risk to user privacy. While some users may be comfortable sharing their location at a coarse granularity, certain types of LBSs require finer-grained locations, such as GPS coordinates, to be effective. For instance, when Alice searches for the nearest coffee shop, she needs to share her precise location to receive accurate results. However, fine granule location data, such as Alice's GPS coordinates, can reveal intimate details about individuals, including their home addresses, workplaces, and frequently visited locations [5]. By correlating this data with publicly accessible information, such as telephone directories, adversaries can deduce individuals' identities, behavior patterns, and personal lives, leading to identity theft, stalking, or unauthorized surveillance [6, 7].

Moreover, users often inadvertently or are compelled to authorize LBS providers to share their data with third parties. These third parties, who may pose even greater risks, further jeopardize user privacy. Recent high-profile incidents involving user location data at Uber [8–12] and unethical data sharing by the fitness app Runkeeper [13] underscore the urgent need for robust privacy protections for users' location data from untrusted LBS providers.

1.1. Motivation

The query sent by Alice is an example of a spatial nearest neighbor (NN) query, which identifies the object that minimizes a distance-based function relative to a query object. Spatial NN queries are the backbone of LBSs with numerous applications, such as proximity search, route planning, navigation, and emergency response (e.g., E-911 service). Accurate distance function evaluation in spatial NN queries necessitates the user's precise location to ensure the correctness of the query results. Consequently, this requirement places user privacy at significant risk when dealing with untrusted LBS providers.

Most existing work on preserving user location privacy in spatial nearest neighbor (NN) queries can be categorized into cooperation- and cryptography-based methods [5,6,14]. Location obfuscation techniques, such as differential privacy [15], cloaking regions [16], and dummy locations [17], obfuscate a user's precise location to preserve privacy. However, these methods, which rely on altered user locations, are only applicable to process aggregate queries, such as group spatial NN queries, and cannot provide exact results

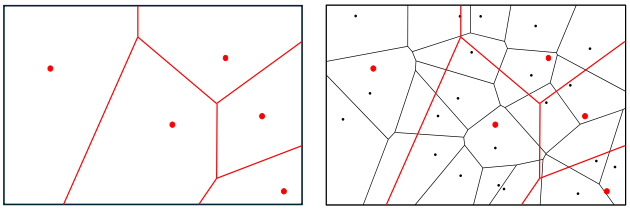
*This material is based upon work supported by the National Science Foundation under Grant Nos. CNS-2131193 and CNS-2219588.

*Corresponding author

zcao@tamusa.edu (Z. Cao); bkish01@jaguar.tamu.edu (B. Kishiyama); jeong.yang@tamusa.edu (J. Yang)

ORCID(s): 0000-0002-4542-7791 (Z. Cao); 0009-0007-7245-3796 (B. Kishiyama); 0000-0002-3819-3544 (J. Yang)

[2]https://doi.org/10.1016/j.jisa.2025.104244



(a) Voronoi diagram with coarse granularity (red), consisting of 5 generator points.

(b) Overlay of two Voronoi diagrams. The Voronoi diagram (black) represents the finer granularity with 21 generator points.

Figure 1: On the left is a coarse-granularity Voronoi diagram (red) with 5 generator points, while on the right, a finer-granularity Voronoi diagram (black) with 21 generator points is overlaid on the coarse-granularity diagram.

for individual spatial NN queries. Additionally, the privacy level is weakly quantified by metrics like location indistinguishability and k-anonymity, making them susceptible to sophisticated attacks, such as side-channel attacks.

On the other hand, cryptography-based methods encrypt user locations to hide them from LBS servers. These approaches can provide exact spatial NN query results for individual queries with rigorous security guarantees. However, the computational and communication overheads associated with encryption schemes make spatial NN searches inefficient and impractical in real-world applications. Moreover, some cryptography-based methods require a middle layer, such as a trusted third party, for encryption key management to ensure secure communication between the client and server [14]. This additional layer can become a network bottleneck and introduce security and privacy vulnerabilities if it is compromised.

In summary, while existing methods can preserve the user's private location in spatial NN queries with various trade-offs, to the best of our knowledge, there are still no practical solutions for privacy-preserving spatial NN query processing with strong security guarantees. This is due to three main challenges: (i) the results of proximity evaluation without the user's precise location are inaccurate, (ii) performing spatial NN search against a large spatial dataset with encrypted user locations is inefficient, and (iii) relying on third parties for secure client-server communication is difficult to ensure in practice.

1.2. Our Solution

To address these challenges, we propose PrivNN, a novel two-party computation framework that allows servers to efficiently process spatial NN queries without knowing users' precise locations. PrivNN comprises two main components: a novel spatial NN search algorithm, named Dynamic Hierarchical Voronoi Overlay (DHVO), which efficiently performs exact spatial NN searches on large spatial datasets, and a secure client-server communication protocol implemented with the homomorphic encryption (HE) scheme [18].

Upon registering with PrivNN, the user generates a pair of public and private keys, storing the public key on the server. The user then encrypts her query location and sends the encrypted query to the server along with a user-specified *privacy profile*, which includes two parameters: the partition factor k, which controls the search refinement granularity and consequently optimizes network traffic load for better communication efficiency, and the initial search region C, which defines a user-customized initial search region to speed up query processing while maintaining security and privacy guarantees.

The main idea behind DHVO is to refine the search space by iteratively overlaying multi-granular Voronoi diagrams. Initially, DHVO computes and saves a Voronoi diagram over the entire search space as its reference layer. It then iteratively overlays multiple Voronoi diagrams, generated through random sampling, onto this reference layer to refine the search space. This refinement process is dynamic, with the granularity of the generated Voronoi diagrams controlled by the user's privacy profile. DHVO returns the NN search result when the search space is refined to contain at most k Voronoi polygons in the base layer. As an example, Figure 1 illustrates the refinement of the search space by overlaying two Voronoi diagrams with different granularities.

To demonstrate PrivNN's performance, we conduct a case study performing spatial NN search on a real-world dataset within a client-server architecture. We systematically evaluate PrivNN with various parameter settings and provide heuristics for optimizing these parameters to enhance PrivNN's performance. The results show that PrivNN can efficiently process exact spatial NN queries under various network conditions.

In summary, the contributions of this paper can be summarized as follows:

1. We propose a novel spatial NN search algorithm, DHVO, to efficiently solve the exact NN spatial search problem. DHVO iteratively refines the search space by overlaying multi-granular Voronoi diagrams under parameter control. We demonstrate that our algorithm consistently produces correct and exact NN search results, with sublinear time and space complexities in the worst-case scenario.
2. We introduce PrivNN, a privacy-preserving two-party computation framework that enables the server to process spatial NN queries with encrypted user locations. PrivNN comprises DHVO and a client-server communication protocol that is parameterized to optimize network traffic load, thereby enhancing communication efficiency.
3. We provide formal analyses of the security and privacy guarantees of PrivNN and prove that PrivNN is secure against semi-honest adversaries under the known background model, along with establishing the privacy bound.
4. We present experimental evidence from real-world datasets demonstrating that PrivNN is efficient in

terms of query processing time and client-server network communication.

We organize this paper as follows: Section 2 reviews prior research related to our work. In Section 3, we formulate the problem and outline our system model, threat models, design goals, and primary notations. Section 4 presents the preliminaries, including the Voronoi diagram and homomorphic encryption. Section 5 describes DHVO, our novel spatial NN search algorithm, while Section 6 details the framework PrivNN for privacy-preserving client and server query processing. Section 7 presents our experimental results. Finally, we conclude in Section 8.

2. Related Work

Our work is related to previous studies on privacy-preserving techniques for the spatial NN search problem. Since the spatial NN search problem can be reduced from the general k-nearest neighbor (k-NN) search problem, in this section, we review prior research on the privacy-preserving k-NN search, also known as the secure k-NN search.

Existing privacy-preserving k-NN search methods can be classified into two categories: exact k-NN and approximate k-NN (AkNN) searches. AkNN methods [19] aim to provide approximate results rather than exact k-NN search results. These methods excel in terms of searching efficiency, especially in large and high-dimensional datasets. However, in this paper, we focus solely on exact k-NN search methods, as our problem setting requires precise search results. Researchers have extensively studied methods to preserve privacy in the exact k-NN search. Based on the mechanisms of these methods, we present an overview of the existing approaches as follows.

2.1. Distortion-based Methods

The core principle of these methods is data distortion through algorithms such as cloaking, spoofing, and perturbation. Cloaking methods [14, 20–27] utilize the k-anonymity property [28] as a privacy guarantee and are widely adopted in the geospatial domain. These methods conceal the precise location of the user by sending a generalized region, known as a cloaking region, instead of an exact point to the server. For example, Mokbel et al. [21] proposed submitting NN queries that include $k-1$ false locations or k queries together with $k-1$ pseudo-users within a cloaking region to anonymize the user using k-anonymity. Furthermore, Gedik et al. [22] extended the cloaking regions to the temporal dimension by allowing a delay until $k-1$ users are present in the cloaking region. However, cloaking methods are primarily applied to process aggregate and group queries and thus are not suitable for our objective of protecting the privacy of individual users.

Another class of distortion-based methods is spoofing. For example, dummy data methods [29–33] use fake locations instead of cloaking regions to conceal user locations. Essentially, these methods select fake locations that are close to the real location and generate an obfuscation set that is then forwarded to the server. The server processes the queries for all locations in the obfuscation set and returns the results. The user subsequently selects the result corresponding to the real location, thereby hiding it from the server. Despite their effectiveness, these methods generate a large amount of dummy query data, resulting in prohibitively high communication and computational overheads. Furthermore, both cloaking and dummy data methods rely on the k-anonymity property [28], which lacks a formal privacy guarantee [34].

Differential privacy (DP) methods [35–41], which distort data through perturbation, work by injecting controlled random noise to obscure sensitive information while still allowing useful aggregate insights to be derived. Unlike approaches based on k-anonymity, DP methods provide provable formal privacy assurances [42]. Research in [43,44] has explored the application of DP to the k-NN search problem in computer vision and other classification tasks, demonstrating that private models can achieve performance comparable to their non-private counterparts. However, despite these advancements, the privacy guarantees of DP come at the cost of introducing random perturbation into the results [45], leading to inaccuracies in individual query results [44]. This limitation makes DP methods unsuitable for our model, which requires precise results for each query.

2.2. Cryptography-based Methods

Cryptography-based methods utilize cryptographic techniques to conceal sensitive data without altering its integrity. Existing methods in this category include space transformation, private information retrieval (PIR), and secure multiparty computation (SMC).

Space transformation techniques (e.g., [46–48]) encrypt location and query data for secure transmission to servers, ensuring that only mobile users can decrypt the data. However, these techniques do not consider the original data distribution, which limits hierarchical access control and results in inaccurate k-NN results. Consequently, these methods are unsuitable for addressing the problem discussed in our paper. PIR is a technique that allows clients to retrieve data from a server without revealing their access patterns to the server [49]. Research in [50, 51] introduced a PIR-based framework for privacy-preserving NN queries in LBSs. Nevertheless, a major limitation of PIR-based methods is the significant storage, network, and computational overheads they incur, making PIR-based methods infeasible for handling user queries efficiently in our framework.

SMC methods employ encryption schemes to allow multiple parties to jointly compute the output of a function without revealing their individual inputs to each other. Prior research has explored the application of SMC methods in secure k-NN queries. For instance, Wong et al. [52] presented an asymmetric scalar-product preserving encryption (ASPE) scheme to perform secure k-NN computations on encrypted databases. The scalar product between encrypted queries and data points is preserved, and the encryption equations built by ASPE enable the user to identify the k closest points among all the encrypted data. Hu et al. [53] discussed the general problem of k-NN query processing over untrusted

data clouds. They leverage a privacy homomorphism encryption method to enable the secure execution of a k-NN best-first search (BFS) algorithm over the R-tree. However, these methods address the problem of protecting the server's dataset against the user, which is the inverse of what we aim to achieve in this paper.

Elmehdwi et al. [54] introduced a fully secure k-NN search protocol over encrypted datasets in an outsourced environment. They developed several basic security protocols, such as secure multiplication, secure squared Euclidean distance, secure minimum, and secure bit-or using the Paillier cryptosystem to construct a comprehensive secure scheme. This scheme not only preserves data privacy and query privacy but also hides the data access pattern from the cloud server. Xu et al. [55] addressed the privacy-preserving k-NN problem by using garbled circuits to simulate Oblivious RAM (ORAM) for accessing data in the kd-tree [56] structure. Similarly, Hsu [57] employed the Paillier cryptosystem with an R-tree [58] structure to protect users' query data from the server. Although these methods are effective in safeguarding users' privacy, they assume a trusted third party between the user and the server, which is not only impractical in reality but also introduces additional communication overhead and potential vulnerabilities to the system.

Other studies have explored methods without assuming a trusted third party. Qi and Atallah [59] proposed a privacy-preserving k-NN search protocol that allows users to query the server without revealing their data. The protocol uses secure two-party computation and is provably secure with linear computation and communication complexities. However, this protocol assumes the dataset is partitioned across all querying users rather than stored on the server. In contrast, our work focuses on the scenario where the server holds the entire dataset, and the user queries the server for k-NN search results. Wang et al. [60] proposed a practical method using ideal secure order-preserving encryption and the R-tree structure to solve a similar problem on large-scale data. In their work, the search procedure includes two interactions between the user and the server. In the first round of interaction, the server narrows down the candidate results by identifying which minimum bounding rectangle (MBR) contains the query and sends the points within the MBR back to the user. The user then creates a search box according to the nearest point in the returned set and sends it to the server. The server outputs all points in that box as the resulting set. The main drawback of this approach is that it does not return accurate results, and additional distance calculations must be performed by the user.

3. Problem Formulation

Our work in this paper addresses a fundamental question within the domain of location-based services: *Is it possible to design a framework that allows users to perform exact spatial NN queries $Q(u_i)$ without revealing their actual locations u_i to the server S?*

Table 1
List of Key Notations

Notation	Description
P	the set of 2-dimensional spatial objects.
U	the set of system users.
$VD(P)$	the Voronoi diagram generated by the set of points P.
$VP(p_i)$	the Voronoi polygon that contains the generator point p_i.
$d(,)$	the Euclidean distance function.
k	the partition factor parameter in DHVO.
C	the initial search region parameter in DHVO.
l	the level of DHVO layer.
S_l	the set of generator points at the l-th DHVO layer.
W_l	the search space at the l-th DHVO layer.
N, q_L	CKKS parameters.

Central to our investigation is the requirement that the query results $R(u_i)$ must not only be accurate but also be returned within an operationally feasible timeframe. By addressing these criteria, we aim to ensure both the privacy of the user's location and the practical usability of the framework. In this section, we first formulate the problem statement with formal definitions, then illustrate the system architecture and explain the threat model, followed by stating our design goals.

For convenience, we list some key notations in Table 1.

3.1. Problem Statement

Consider a set of spatial objects in a 2-dimensional space, represented as $P = \{p \mid p \in \mathbb{R}^2\}$. The dataset P is hosted by a server S (for example, an LBS provider) to provide query service. Each object p_i in P is defined by a tuple $p_i = (p_{i,1}, p_{i,2})$, which denotes the coordinates in the space, i.e., latitude and longitude. Concurrently, a set of users $U = \{u \mid u \in \mathbb{R}^2\}$ exists within the same space. A user u_i in U, determined by their location $u_i = (u_{i,1}, u_{i,2})$, initiates a NN query $Q(u_i)$ to the server S using its dataset P. The server, applying a distance function d (such as Euclidean distance), returns the query result $R(u_i) = \{p \in P \mid \forall p' \in P, d(u_i, p) \leq d(u_i, p')\}$, which includes the spatial object nearest to the user's location.

3.2. System Architecture

We illustrate the architecture of PrivNN in Figure 2, which depicts a two-party computation framework consisting of two principal entities: the query service provider (referred to as the server) and system-authorized users. The server manages a database containing spatial objects, which are considered proprietary assets. Each user can mutually authenticate with the server and possesses a unique pair of public and private cryptographic keys. In contrast, the server has access only to the user's public keys.

Users are authorized to submit NN queries with encrypted locations to the server. Upon receiving a query, the server processes it and returns the results without decrypting

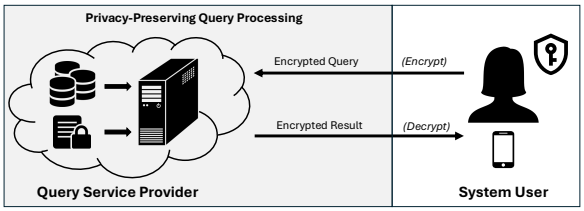


Figure 2: The architecture of the PrivNN framework. PrivNN does not assume any trusted third party. The server hosts the database, along with encryption parameters associated with each user. The private keys are known only to the users, ensuring that the server cannot decrypt the users' queries. The region shaded in light grey represents the user's query in its encrypted state.

the query. This mechanism ensures that the user's precise location remains confidential, effectively concealing it from the server.

3.3. Threat Model

We model the server as *semi-honest* (also known as "honest-but-curious" [61–65]): it correctly follows the protocol but may attempt to glean extra information from the messages it processes. We assume the server trusts all clients, while the server itself could be compromised or collude with external adversaries in an attempt to break privacy. Finally, we protect every client–server exchange using standard network-layer encryption (e.g. TLS/SSL).

There are two known threat models: the *known ciphertext model* and the *known background model*, each based on different assumptions about the server's capabilities. Both models are examined in existing works [66, 67]. We provide formal proofs for the security of our framework under the *known background model* in Section 6.

3.4. Design Goals

We observe that an effective framework must preserve user privacy against a *semi-honest* server with the following properties:

1. *Query privacy.* The authorized user submits her encrypted location to the server. The latter executes the query and sends the nearest neighbor from its database to the user. Except for the nearest neighbor, the server does not learn the precise location of the user.
2. *Database privacy.* The server's database may contain proprietary information that should remain private. At the end of the query process, the user learns nothing about the database beyond the nearest neighbor to her query.
3. *Accuracy.* The server accurately computes the nearest neighbor of the user's query.
4. *Efficiency.* The framework incurs low computational overhead on the user, providing practical and scalable query processing.

We note that our framework does not protect against the case of a server corrupting the database. Additionally, we assume that the server is motivated to provide a useful service and, therefore, has no incentive to provide inaccurate or deceptive query results.

4. Preliminaries

Our proposed approach to address privacy-preserving NN queries leverages the Voronoi diagram and homomorphic encryption (HE). In this section, we begin by reviewing the definition of Voronoi diagrams, focusing on the 2-dimensional Euclidean space and highlighting the properties relevant to our approach. We then introduce the preliminaries of HE, which serve as the foundational elements of our proposed communication protocol.

4.1. Voronoi Diagram

Voronoi diagram [68] is a data structure that is extremely efficient in exploring a local neighborhood in a geometric space. Consider the set of spatial objects $P = \{p_1, \dots, p_m\} \subset \mathbb{R}^2$, in the Euclidean plane, the Voronoi diagram of the given set of points P partitions the space of $\mathbb{R}^n$ into m regions. The set of points P is called generator points. Each region includes all points in $\mathbb{R}^2$ with a common closest point in the given set P according to the Euclidean distance function d [69]. More formally, the *Voronoi polygon* and Voronoi diagram can be defined as: Assume the generators set $P = \{p_1, \dots, p_m\} \subset \mathbb{R}^n$, where $2 < m < \infty$ and $p_i \neq p_j$ for $i \neq j,\ i, j \in I_n = \{1, \dots, m\}$. The region is given by:

$$VP(p_i) = \left\{ p \mid d\left(p, p_i\right) \leq d\left(p, p_j\right) \right\} \text{ for } j \neq i,\ j \in I_n \quad (1)$$

where $d(p, p_i)$ specifies the Euclidean distance between p and p_i, i.e., length of the straight line connecting p and p_i in Euclidean space, is called the *Voronoi polygon* associated with p_i, and the set given by:

$$VD(P) = \left\{ VP\left(p_1\right), \dots, VP\left(p_m\right) \right\} \quad (2)$$

is called the Voronoi diagram generated by P with respect to the Euclidean distance function d. The Voronoi polygons of a Voronoi diagram are collectively exhaustive because every location in the plane is associated with at least one generator. The polygons are also mutually exclusive except for their boundaries. The boundaries of the polygons called *Voronoi edges*, are the set of locations that can be assigned to more than one generator. The Voronoi polygons that share the same edges are called adjacent polygons, and their generators are called adjacent generators. Figure 1(a) shows the Voronoi diagram created by 5 generator points in $\mathbb{R}^2$ with the Euclidean distance function. Throughout this paper, we use the Euclidean distance function in the space of $\mathbb{R}^2$.

Next, we review two basic geometric properties of the Voronoi diagrams. The proofs for these properties are presented in [69]. These properties are the basis for proving the correctness of our algorithm introduced in Section 5.

Property 1: The Voronoi diagram of a certain set P of points, $VD(P)$, is unique.

Property 2: Given the Voronoi diagram of P and a query point $q \notin P$, a point p is the nearest point of P to q, if and only if $q \in VP(p)$.

4.2. Sqaured Euclidean Distance

The Squared Euclidean distance (SED) is extensively utilized in computational applications, notably in algorithms such as k-means clustering [70], due to its operational simplicity compared to the traditional Euclidean distance. Consider two points in an n-dimensional space $\mathbb{R}^n$: $u = (u_1, u_2, \dots, u_n)$, representing a user's location, and $p = (p_1, p_2, \dots, p_n)$. As $d(u, p)$ denotes the Euclidean distance between u and p, then we denote SED as $d^2(u, p)$, is calculated as:

$$d^2(u, p) = \|u - p\|_2^2 = \sum_{i=1}^{n} (u_i - p_i)^2 \tag{3}$$

Here, $d^2(u, p)$ corresponds to the square of the ℓ^2-norm of the vector difference between the points, effectively eliminating the square root operation typically involved in computing Euclidean distances.

Given that the squaring function is strictly increasing for non-negative values, SED inherently preserves the ordering of distances as determined by the Euclidean metric. Consequently, the relationship:

$$\|u - p\|_2 \text{ is minimal} \iff \|u - p\|_2^2 \text{ is minimal} \tag{4}$$

holds true. This preservation of ordering renders the Squared Euclidean Distance (SED) particularly advantageous in contexts where relative distance comparisons are crucial. By circumventing the computational demands of the square root calculation, SED facilitates efficient and accurate ranking and proximity evaluations. Therefore, we adopt SED for distance computations in this paper.

4.3. Homomorphic Encryption (HE)

HE is a special type of encryption that supports arithmetic operations between ciphertexts without decryption. It has been successfully applied in various fields [71–74] involving private data as it allows the analysis of encrypted data without information leakage from messages. HE was first proposed in a blueprint by Gentry and Boneh [18], and a number of HE schemes have been suggested [75–84]. The term "homomorphic" derives from the Greek words "homos" and "morphe," which together mean "same shape" [85]. In the field of algebra, the term homomorphism is used for "a structure-preserving mapping between two algebraic structures" [86]. In terms of cryptography, this means that the input clear text data (or plaintexts $\mathcal{P}$) is mapped to encrypted data (or ciphertexts $\mathcal{C}$) such that the algebraic structure between $\mathcal{P}$ and $\mathcal{C}$ is preserved over addition and multiplication. More concretely, let $a, b \in P$ and ENC denotes the HE encryption procedure, then $ENC(a) \oplus ENC(b) = ENC(a + b)$ and $ENC(a) \odot ENC(b) = ENC(a \cdot b)$, where $\oplus$ and $\odot$ are homomorphic addition and multiplication, respectively, and where equality is achieved after decryption.

The possible operations of HE methods are limited to addition and multiplication in known methods. Nevertheless, HE methods are able to perform arbitrary operations, as proven by Gentry [87]. He demonstrated that circuit functions that are based on only additions and multiplications enable modeling arbitrary operations [87]. This allows one to evaluate arbitrary computations (modeled as circuits) on encrypted data by only manipulating the ciphertexts.

HE is typically classified into three categories based on the type and number of operations it supports: Partially Homomorphic Encryption (PHE), Somewhat Homomorphic Encryption (SHE), and Fully Homomorphic Encryption (FHE) [85, 87]. PHE schemes are limited to supporting either addition or multiplication, but not both. In contrast, both SHE and FHE are capable of performing unrestricted addition and multiplication. A key distinction between SHE and FHE lies in their error-correction mechanisms [18]. In HE schemes, ciphertexts are intentionally infused with a small degree of noise to enhance security [88]. This noise accumulates with each computational operation (high rate for multiplication and low rate for addition), potentially leading to decryption failure if it surpasses a predefined threshold. To address this challenge, FHE employs a bootstrapping technique, which effectively reduces a high-noise ciphertext to a low-noise one, thereby facilitating the evaluation of functions at arbitrary computational depths without decryption. However, bootstrapping is notably more computationally demanding than other operations due to its reliance on complex, non-polynomial decryption processes. Conversely, SHE avoids the use of bootstrapping, thus requiring less computational power. Nonetheless, this limitation restricts the number of permissible operations before the noise level reaches a critical error threshold, beyond which computational errors may become prohibitively large [18, 89].

Modern FHE schemes exhibit a diversity in their underlying mathematical structures, capabilities, and performance metrics. In this paper, we utilize the squared Euclidean distance function over real numbers to measure spatial proximity. As our method has an indeterministic number of homomorphic operations, we have used the CKKS scheme [77] to encrypt the user's queries as it can achieve fully homomorphic capabilities without depth limitations. Moreover, the CKKS scheme is particularly suited for handling computations on encrypted floating-point numbers, thereby aligning well with the needs of our proposed method.

4.4. The CKKS Scheme

The CKKS scheme, as described in [77], is a leveled approximate FHE scheme. During encryption in CKKS, a minimal amount of noise is introduced into the plaintext to secure the ciphertext, and this noise remains even after decryption. The impact of this noise is minimal relative to the decryption output, which is why the output is considered an "approximate" decryption result. The efficiency of CKKS has been proven for many real-world applications, such as machine learning [74, 90] and cyber physical system [91],

and is still being improved with better bootstrapping algorithms [92,93]. Next, we briefly describe the CKKS scheme used in our work as defined in [77,94,95], and omit details, such as *Relinearization* and *Rescaling* for simplicity.

Consider N as a power of 2, and define $R = \mathbb{Z}[X]/(X^N + 1)$ as the ring of polynomials with integer coefficients, each having a degree less than N and taken modulo $X^N + 1$. For a given positive integer q, the ring $R_q = \mathbb{Z}_q[X]/(X^N + 1)$ includes identical polynomials but with coefficients modulo q. Suppose $q_L > q_{L-1} > \cdots > q_1$ represent L increasing positive integers where each $q_l = \prod_{i=1}^{l} p_i$ and the p_i are small prime numbers. In the CKKS scheme, L indicates the highest level of multiplicative depth available. At any specific level l from 1 to L, all operations are executed within $R_{q_l} = \mathbb{Z}_{q_l}[X]/\langle X^N + 1\rangle$. Initially, encrypted data involves ciphertext polynomials from R_{q_L} at the highest level L, which may later be adjusted to $R_{q_{L-1}}$. This process in CKKS suggests a framework with L computational levels, starting from the top and potentially moving downward as necessary. The terms DFT and IDFT refer to the Discrete Fourier Transform and its inverse. Given input data represented as real or complex numbers, we describe the main algorithms and holomorphic operations [95] used in our work as follows:

- `Setup`: given a desired security level λ, and maximum computation levels L, initialize CKKS by setting N, two uniform random distributions: $\mathcal{X}_{key}$ over R_2 and $\mathcal{X}_{q_L}$ over R_{q_L}, and a zero-mean discrete Gaussian distribution $\mathcal{X}_{err}$ with standard deviation σ over R_{q_L}.
- `KeyGen`: let $s \leftarrow \chi_{key}$ and $e \leftarrow \chi_{err}$. Sample uniformly random $a \in R_{q_L}$. Output the secret key $\mathtt{sk} = (1, s)$, the public key $\mathtt{pk} = (b, a)$, where $b = [-a \cdot s + e]_{q_L}$.
- $\mathtt{Encode}_{\mathtt{pk}}(v, \rho)$: given a vector of complex numbers $v \in \mathbb{C}^{N/2}$ and precision ρ, return a polynomial $\mu = \lfloor \mathrm{IDFT}(2^\rho v) \rceil \in R$.
- $\mathtt{Enc}_{\mathtt{pk}}(\mu)$: given a plaintext message μ, sample $v \leftarrow \mathcal{X}_{q_L}$ and $e_0, e_1 \leftarrow \mathcal{X}_{err}$. Return ciphertext $ct = \big(c_0, c_1\big) = \big(av + \mu + e_0, bv + e_1\big) \in R^2_{q_L}$.
- $\mathtt{Dec}_{\mathtt{sk}}(\mathtt{ct})$: given a ciphertext $ct \in R^2_{q_l}$, return $\mu = c_0 + sc_1 \in R_{q_l}$.
- $\mathtt{Decode}_{\mathtt{sk}}(\mu, \rho)$: given $\mu \in R$ and precision ρ, return $v = \mathrm{DFT}(\mu/2^\rho) \in \mathbb{C}^{N/2}$.

Homomorphic operations can be carried out via:

- $\mathtt{HAdd}(ct_0, ct_1)$: For ciphertexts ct_0 and ct_1 at the same level l, output the ciphertext $ct^+ = ct_0 + ct_1 \in R^2_{q_l}$.
- $\mathtt{HMult}(ct_0, ct_1)$: For ciphertexts ct_0 and ct_1, output the ciphertext $ct^\times = (c_0, c_1, c_2) \in R^3_{q_l}$, where
  $c_0 = [ct_0[0] \cdot ct_1[0]]_{q_l}$,
  $c_1 = [ct_0[0] \cdot ct_1[1] + ct_0[1] \cdot ct_1[0]]_{q_l}$,
  $c_2 = [ct_0[1] \cdot ct_1[1]]_{q_l}$.
  Note that the *relinearization* procedure can reduce $ct^\times$ back to two elements $\in R^2_{q_l}$.
- $\mathtt{HAddPlain}(ct, pt)$: homomorphic addition of a ciphertext $ct = (c_0, c_1) \in R^2_{q_l}$ and plaintext $pt \in R$, output the ciphertext $ct^+ = (c_0 + pt, c_1) \in R^2_{q_l}$.
- $\mathtt{HMultPlain}(ct, pt)$: For ciphertexts $ct_0 = (c_0, c_1) \in R^2_{q_l}$ and plaintext $pt \in R$, output the ciphertext $ct^\times = (c_0 \cdot pt, c_1 \cdot pt) \in R^2_{q_l}$.

We adopt the CKKS encryption scheme as it supports both addition and multiplication on encrypted floating point data, which is well-suited for distance function evaluation in our work. However, note that the ciphertext size and noise may increase during operations. To manage these, we apply the relinearization procedure to control ciphertext size and maintain accuracy in multiplicative operations.

5. Dynamic Hierarchical Voronoi Overlay (DHVO)

In this section, we introduce DHVO, a dynamic spatial NN search algorithm. We begin by detailing the construction process of DHVO layers with increasing granularity. Following this, we present the DHVO algorithm along with its correctness proof. Finally, we conduct a complexity analysis and compare DHVO with existing methods.

5.1. DHVO Layer Construction

We consider a 2D spatial dataset P containing 21 points in Euclidean space. Our objective is to identify the point in P closest to a given query point q using the DHVO structure. The key parameter to build the DHVO layers is the partition factor k. For this illustration, we set the $k = 5$ and use the Euclidean distance function to evaluate proximity.

Firstly, we construct $VD(P)$, the Voronoi diagram of P, and save it as the reference layer, as illustrated in Figure 3. This reference layer is used for building subsequent layers. Next, starting from layer 1 on the left, we randomly sample 5 points, defined by k, as set S_1 from a uniform distribution across the entire search space, and construct layer 1 by building the Voronoi diagram of S_1, $VD(S_1)$. Using the Euclidean distance function, we then identify the Voronoi polygon, $VP(s_{1,2})$ in layer 1 containing the query point q.

Next, we perform a geometric intersection operation between the Voronoi diagram $VD(P)$ and $VP(s_{1,2})$. We compute the number of intersecting Voronoi polygons in $VD(P)$ with $VP(s_{1,2})$. Since the number of intersecting polygons exceeds 5, we proceed to build layer 2 in DHVO to form a reduced search space that includes the union of all intersecting Voronoi polygons in the reference layer. This process of constructing subsequent layers continues until the number of intersecting Voronoi polygons is less than or equal to 5. As in the example shown in Figure 3, we reach the base layer of the DHVO structure, where there are 4 intersecting Voronoi polygons in the reference layer, which is less than $k = 5$. At the base layer, DHVO reduced the search space to $VD(P')$, where $P' \subseteq P$ and $|P'| \leq k$. Finally, we compute the distances and identify $p_7 \in P$ as the nearest data point to q.

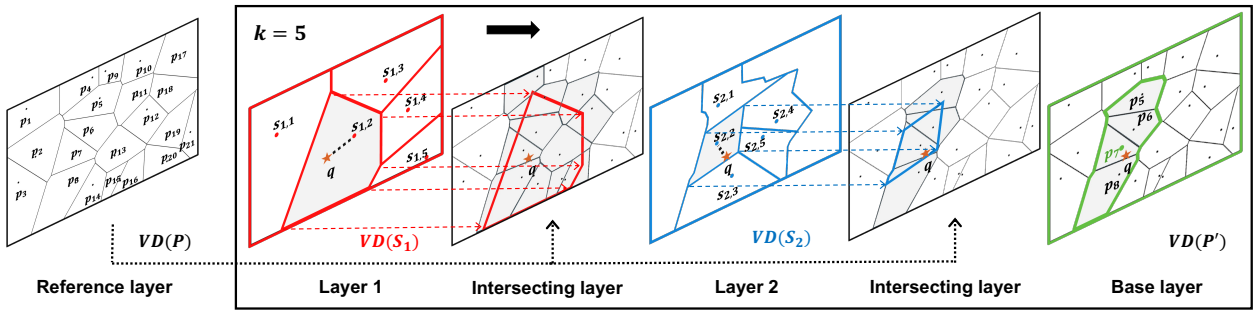


Figure 3: A geometrical illustration of building the DHVO layers to identify nearest neighbor in P to query point q, with $|P| = 21$ and $k = 5$. In this example, the DHVO construction reaches the base layer after constructing 2 intermediate layers. At the base layer, the point $p_7 \in P$ is identified as the nearest point to q.

In summary, DHVO iteratively partitions the search space by constructing Voronoi diagrams as subsequent layers, each progressively narrowing the search space containing the query point q until the nearest neighbor can be identified with fewer than k distance computations. The search space partitioning rate is controlled by the partition factor k. It is important to note that the sampling process at each intermediate layer, such as layer 1 and layer 2, draws k points randomly and may not belong to P. However, the Voronoi polygons at the base layer are generated by points in P, ensuring that the nearest neighbor result is strictly from P.

5.2. Algorithm

Building on the construction process of the DHVO layers described in Section 5.1, we propose our NN search algorithm detailed in Algorithm 1. This algorithm takes as inputs a spatial dataset P as the candidate set, a query point u, and a partitioning factor k, outputting the nearest neighbor in P to u. Initially, the Voronoi diagram $VD(P)$ is computed for all candidates in P as the reference layer (line 1). This reference layer is then used to initialize the Voronoi diagram $VD(P')$ in the initial search space (line 2), along with the initialization of the layer variable l (line 3). At each intermediate layer, partition candidates are sampled within the search space from the previous layer (lines 6-7). The search process progresses through successive DHVO layer constructions, iteratively reducing the candidate set P' via set intersection operations (line 10). The search loop terminates when there are at most k candidates left in P'.

The correctness of our algorithm is based on the properties of the Voronoi diagram (Properties 1 and 2 in Section 4), and the condition that the Voronoi polygon $VP(p)$ containing the query point q is contained within the search spaces at each DHVO layer. By applying mathematical induction, we establish the correctness of our algorithm.

Lemma 1. *Given a query point q and a spatial dataset P of points in $\mathbb{R}^2$, the Voronoi polygon $VP(p)$ containing the query point q is contained within the search space at each DHVO layer (i.e., $VP(p) \subseteq W_l$ for each layer l).*

Algorithm 1: DHVO Algorithm

Input: Spatial dataset $P = \{p_1, \ldots, p_m\}$; Query point u;
Parameter: Partitioning factor k
Output: The nearest point $p \in P$ to u;

1 Compute Voronoi diagram $VD(P)$ for P as generator points
2 $VD(P') \leftarrow VD(P)$ `// initialize VD(P')`
3 $l \leftarrow 0$ `// layer 0 is reference layer`
4 **while** $|P'| > k$ **do**
5 $\quad l \leftarrow l + 1$
6 $\quad W_l \leftarrow \bigcup_{p_i \in P'} VP(p_i)$ `// current search space`
7 $\quad S_l \leftarrow \text{Sample}(W_l, k)$
8 $\quad$ Compute Voronoi diagram $VD(S_l)$ for generator points S_l
9 $\quad s \leftarrow$ the point in S_l nearest to u
10 $\quad VD(P') \leftarrow VD(P) \cap VP(s)$
11 $p \leftarrow$ the point in P' nearest to u
12 **return** $p \in P'$ closest to u `// P' ⊆ P`

Proof. We construct our proof using mathematical induction. Consider $p \in P$ as the nearest neighbor to the query point q, i.e., $q \in VP(p)$. At layer 1, the search space is the same as the Voronoi diagram $VD(P)$ generated by the points in P. Hence, $VP(p)$ is evidently contained in the search space.

Assume that at layer l, $VP(p)$ is contained within the search space W_l. Let point $s_{(l,1)}$ be the nearest neighbor to q at layer l. The search space W_{l+1} at layer $l+1$ is formed by the geometric intersection of $VD(P)$ with $VP(s_{(l,1)})$ (lines 6 and 10 in Algorithm 1). Since $q \in VP(s_{(l,1)})$ and $q \in VP(p)$, it follows that $VP(p) \subseteq W_{l+1}$.

Therefore, by induction, $VP(p)$ is contained within the search space at each DHVO layer, *Lemma 1* holds. □

5.3. Complexity Analysis

DHVO finds the NN in a spatial dataset P by iteratively partitioning the search space using Voronoi diagrams. The algorithm reduces the search space by a factor of k at each layer, resulting in a logarithmic reduction in the search space

size. The number of layers needed to reduce the search space to the base layer is $\log_k m$, where m is the number of elements in the search space.

The time complexity for each layer involves constructing a Voronoi diagram for k points and performing set intersections. Constructing a Voronoi diagram for k points by Fortune's algorithm [96] has a time complexity of $O(k \log k)$. Therefore, the total time complexity of the DHVO algorithm is:

$$O(k \log k \cdot \log_k m) = O(k \log m) \tag{5}$$

Since k is a constant, DHVO achieves a sublinear time complexity for finding the NN in a spatial dataset.

The space complexity is determined by the storage requirements for the partition points at each layer, which is $O(k)$ for each of the $\log_k m$ layers, leading to a total space complexity of:

$$O(k \cdot \log_k m) = O(k \log m / \log k) \tag{6}$$

6. PrivNN

6.1. Protocol of PrivNN

We outline the communication protocol of PrivNN in Algorithm 2, which facilitates a private nearest neighbor (NN) search between the client and the server. Initially, the client encrypts the query point u using the CKKS public key, resulting in the ciphertext c_u. The client then selects the partition factor k and the initial search region C, and sends both parameters along with c_u to the server (lines 1-2). The server, which already has a pre-computed Voronoi diagram $VD(P)$ of its dataset P, initializes a working Voronoi diagram $VD(P')$ as a copy of $VD(P)$ (line 3). We introduce the initial search region C as a region enclosing the query point u to speed up the query processing without sacrificing privacy. A formal analysis of the impact of the size of C on the privacy bound is provided in Section 6.4.

As the protocol progresses, the server iteratively refines P' to focus on potential nearest neighbors. In each iteration, if the size of P' exceeds k, the server samples k generator points from P', constructs a temporary Voronoi diagram, and calculates the squared Euclidean distance (SED) between u and each generator point using homomorphic operations (line 6). These SED values, still encrypted, are sent back to the client. The client decrypts these values, identifies the minimum SED, and informs the server of the index of this minimum (line 8). The server then performs a set intersection operation between the reference layer $VD(P)$ and the polygon containing the generator point with the minimum SED, updating $VD(P')$ with the intersecting polygons in $VD(P)$ (line 9). This iterative process continues until there are at most k points in P'. The server then calculates the SEDs for all points in P' and sends these encrypted values to the client, who decrypts them to find and confirm the nearest neighbor p. This protocol ensures that the DHVO algorithm is executed in a privacy-preserving manner, with the server unable to learn the user's location.

Algorithm 2: Protocol of PrivNN

Input: The point set $P = \{p_1, \ldots, p_m\}$, pre-computed Voronoi diagram $VD(P)$, user query point u;
Parameters: k, C, CKKS parameters;
Output: The nearest point $p \in P$ to u from the server's dataset;

```
// At client:
```
1 Creates ciphertext c_u by encrypting the query point u with CKKS public key.
2 Chooses parameters k and C, and sends c_u, k and C to the server.

```
// At server:
```
3 Server initializes $VD(P')$ by performing a set intersection between C and $VD(P)$
4 **while** $|P'| > k$ **do**
5 Server create a Voronoi diagram by sampling k generator points as S within the search region $VD(P')$.
6 Server homomorphically computes SED values between c_u and S by using homomorphic operations `AddPlain`(c_u, `S`) and `MultPlain`(c_u, `S`).
7 Server sends SED values in ciphertext to the client.

```
// At client:
```
8 Client decrypts SED values and sends the index of the nearest generator point in S to the server.

```
// At server:
```
9 Server updates $VD(P')$ by performing a set intersection between the polygon containing the nearest generator point and $VD(P)$.

```
// At server:
```
10 Server homomorphically computes $|P'|$ SED values in cyphertext by using homomorphic operations `AddPlain`(c_u, `P'`) and `MultPlain`(c_u, `P'`).
11 Server sends $|P'|$ SED values in ciphertext with P' to the client.

```
// At client:
```
12 Client decrypts $|P'|$ SED values and finds the $p \in P$ with minimum SED value from P'.
13 **return** p

6.2. Security Analysis

This section presents security proof for the PrivNN protocol against semi-honest adversaries within the known background model. We first use the security games theory [97] to prove that the CKKS FHE scheme is semantically secure, as formalized in the following Lemma.

Lemma 2. *The CKKS scheme is semantically secure if the encrypted messages are indistinguishable.*

Proof. We aim to demonstrate that the probability of a probabilistic polynomial-time (PPT) adversary $\mathcal{A}$ successfully breaking the encrypted messages of the CKKS scheme is negligible. Suppose the challenger executes `Setup` and `KeyGen` to initialize the CKKS scheme and generate a key pair consisting of a secret key `sk` and a public key `pk` by setting the security parameter N. The adversary $\mathcal{A}$ submits two messages m_0 and m_1. The challenger randomly selects $b \in \{0, 1\}$, encrypts m_b using `pk`, and sends the resulting ciphertext to $\mathcal{A}$. Given that CKKS incorporates random noise in the encryption process, this transforms the plaintext into varied ciphertexts even when using the same key, ensuring that $\mathcal{A}$ cannot guess b correctly with a probability greater than $1/2$. Consequently, the advantage in the security game is defined as $\mathrm{Adv}_{CKKS,\mathcal{A}}(N) = |\Pr[b = b'] - \frac{1}{2}| < \mathrm{negl}(N)$. Here, $\mathrm{negl}(N)$ is a negligible function of N that is asymptotically smaller than any polynomial inverse as N grows, implying that the adversary's advantage is too small to be practically significant. Additionally, the CKKS scheme relies on the foundational hardness of the Ring Learning With Errors (Ring-LWE) problem, which is known to be difficult to solve efficiently with polynomial-time algorithms [98]. Therefore, *Lemma 2* holds. □

The indistinguishability of encrypted messages in the CKKS scheme is based on the added random noise and the computational difficulty of the underlying Ring-LWE problem, ensuring robust privacy protection.

Based on Lemma 2, we adopt the simulation-based security model [99–101] to prove that PrivNN is secure under the known background model.

Theorem 1. *PrivNN is secure against semi-honest adversaries under the known background model.*

We first introduce some notions used in [101] and adapt them for our proof.

- *History*: an interaction between the user and server, determined by a dataset $\mathcal{D}$, and a set of queries $Q = \left(q_1, \ldots, q_\tau\right)$ submitted by users, denoted as the knowledge $H = (\mathcal{D}, Q)$.
- *View*: the encrypted form of H under the shared public key `pk`, denoted as $V(H) = (\mathcal{D}, TK(Q))$, where $TK(Q)$ is the encrypted queries. Note that the server can only see the views.
- *Trace*: given a history H, the trace $Tr(H)$ is the information that can be learned by the server. It contains the access pattern $\alpha(H)$, search pattern $\sigma(H)$, and the returned identifiers $ID(Q)$. Let $ID(q)$ be the identifier of the matched data record, and $\alpha(H) = \{ID(q_1)), \ldots, ID(q_\tau)\}$. The search pattern $\sigma(H)$ is an $n \times \tau$ binary matrix, where n is the size of $\mathcal{D}$, $\sigma(H)_{i,j}$ is 1 if the $\mathcal{D}_i$ is returned by a query q_j, and 0 otherwise. Then we have $Tr(H) = \{ID(Q), \alpha(H), \sigma(H)\}$.

Under the known background model, we assume the server obtains the $Tr(H)$, and a certain number of queries and its probability pairs (q_i, p_i). In simpler terms, if two histories produce the same trace and the server, given the query distribution, cannot tell which one was created by the simulator, then it gains no extra information beyond what is intended to be disclosed (namely, the trace). Therefore, our approach is considered secure.

Proof. Let $\mathcal{S}$ be a simulator that can simulate a view V' indistinguishable from the server's view $V(H) = (\mathcal{D}, TK(Q))$, with the same trace $Tr(H) = \{ID(Q), \alpha(H), \sigma(H)\}$. To achieve this, the $\mathcal{S}$ does the following:

- $\mathcal{S}$ constructs the query Q' and the encrypted queries $TK(Q')$ as follows. For each $q'_i \in Q', 1 \le i \le \tau$,
  1. Generate a vector of 2 elements, denoted as $u' = (u'_1, u'_2)$, where u'_1 and u'_2 are a pair of random numbers as 2D coordinates within the Voronoi polygon identified by $ID(q'_i)$.
  2. Generate the encrypted query for each q'_i by encrypting u' with the shared public key `pk`. Then the $\mathcal{S}$ obtains $\mathsf{Enc}_{\mathsf{pk}}(Q') = \{\mathsf{Enc}_{\mathsf{pk}}(q'_1), \ldots, \mathsf{Enc}_{\mathsf{pk}}(q'_\tau)\}$.
- $\mathcal{S}$ outputs the view $V' = (\mathcal{D}, TK(Q'))$.

The correctness of such construction is easy to demonstrate by querying $TK(Q')$ over $\mathcal{D}$. The dataset $\mathcal{D}$ and the encrypted queries $TK(Q')$ generate the same trace as the one that the server has. We claim that no probabilistic polynomial-time (PPT) adversary can distinguish the view V' from $V(H)$. Specifically, due to the semantic security of CKKS, no PPT adversary can distinguish the $TK(Q')$ from $TK(Q)$. Moreover, the PPT adversary with the query and probability pairs cannot distinguish which encrypted queries originate from the same underlying query because of the added random noise during the CKKS encryption process, so it cannot exploit the distributions of plain and encrypted queries to infer any additional information. Consequently, *Theorem 1* holds. □

6.3. Robustness Against Advanced Threats

We extend our security analysis beyond the semi-honest server assumption to address more sophisticated adversarial scenarios. PrivNN performs spatial nearest neighbor queries entirely within the encrypted domain using fully homomorphic encryption (FHE), ensuring that a semi-honest server cannot decrypt or infer any information about user queries or database entries. Even in the presence of collusion between a semi-honest server and malicious users, PrivNN remains secure, as plaintext data is never exposed to any party. To defend against active attacks—such as data manipulation or computation tampering—PrivNN can be enhanced with lightweight verifiable computation mechanisms, including SNARKs [102, 103] or homomorphic message authentication codes (MACs) [104], allowing clients to detect unauthorized modifications with minimal overhead. Furthermore, unlike certain FHE-based schemes that require a trusted third party for key generation or parameter setup, PrivNN operates strictly within a two-party model, eliminating external trust assumptions. In summary, PrivNN offers strong

resilience against a broad range of adversarial threats, delivering a practical and comprehensive security framework.

6.4. Privacy Analysis

In this section, we analyze the privacy metric of PrivNN by measuring the disclosure risk [105] of the user's location to the adversary. Disclosure risk represents the probability that an attacker may know about the user's location and other sensitive information according to released data and background knowledge [105]. Typically, the more background knowledge available to the adversary, the greater the risk of disclosure. We use $\mathcal{L}$ to denote the user's location and $\mathcal{L}_{\mathcal{K}}$ to denote disclosing $\mathcal{L}$ using background knowledge $\mathcal{K}$. Thus, $r(\mathcal{L}, \mathcal{K})$, representing the disclosure risk as the probability of $\mathcal{L}_{\mathcal{K}}$, as $r(\mathcal{L}, \mathcal{K}) = P_r(\mathcal{L}_{\mathcal{K}})$.

According to the protocol, the user sends an NN query, which consists of an initial search region denoted as $\mathcal{C}$, along with their encrypted location, to the server. We begin by dividing the entire search area into grid cells, each of which is small enough to potentially identify individual users; we denote the area of such a grid cell as S_0. The area of the initial search region $\mathcal{C}$ is represented as S_C. We also specify the areas of all the Voronoi polygons as S_{VP} and the smallest Voronoi polygon as $\min(S_{VP})$. Intuitively, the server's ability to determine the user's exact location is given by the ratio $S_0/\min(S_C, \min(S_{VP}))$. It is important to note that users should not set their initial search region to be smaller than the smallest Voronoi polygon, i.e., $S_C \gg \min(S_{VP})$. This condition can be ensured through a pre-negotiation between the client and server prior to the search process. Consequently, we define the disclosure risk as follows:

$$r(L, K) = P_r(L_K) \leq \frac{S_0}{min(S_{VP})} \tag{7}$$

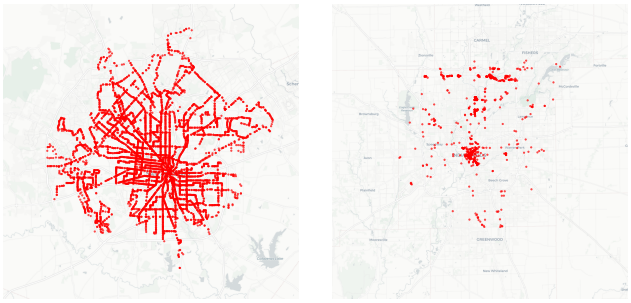
It is clear that the worst-case degree of privacy offered by PrivNN is contingent on the area of the smallest Voronoi polygon. This observation is logical, as in densely populated urban areas, the results of nearest neighbor searches are likely to disclose more about a user's location compared to those conducted in sparsely populated suburban areas. The dataset density in the search area thus directly impacts the amount of location information potentially exposed.

7. Experimental Study

In this section, we first present the performance of PrivNN in comparison with baseline methods for privacy-preserving NN queries, evaluated on two publicly available datasets. We then analyze the impact of parameter tuning on the performance of PrivNN in greater detail.

7.1. Experimental Setup and Dataset

We implemented PrivNN using Ubuntu 22.04 and Python 3.9. For Voronoi diagram computation, we employed *Qhull* via the *SciPy* library. Networking was handled using Python's *socket* and *ssl* libraries. The CKKS scheme was implemented using the *SEAL* [106] library, accessed through the Pyfhel [107] Python wrapper. For the CKKS scheme parameters, we selected the security parameter $N = 8192$ and a total coefficient-modulus size of $log_2 q_L = 218$. We note that our CKKS parameters satisfy a 128-bit security level [108]. We deployed the client and server on Google Cloud Platform (GCP) instances, each with 4 virtual cores and 16 GB of RAM.



(a) High-density POIs in the VIA dataset.

(b) Low-density POIs in the Yelp dataset.

Figure 4: Overview of POI densities from two datasets.

Prior to query processing, the client generates a CKKS key pair, storing the public key on the server while retaining the private key locally. The server associates each client's public key with their respective client ID. Notably, clients only need to generate their key pair once during the initial registration process with the server. Additionally, the server pre-computes the Voronoi diagram using all data points as generators, storing this as the reference layer. This reference layer is pre-loaded into the server's memory before processing queries, ensuring efficient query responses.

We conducted experiments on two publicly available datasets to assess the practicality of the PrivNN framework in supporting spatial NN queries across diverse real-world scenarios.

VIA dataset. The VIA dataset [109] is a public transportation dataset provided by VIA Metropolitan Transit in San Antonio, Texas. It consists of 76,707 bus stops within the city limits, each represented by a pair of geographic coordinates. We used this dataset to simulate a high-density POI scenario, in which a bus rider (the client) queries the service provider (the server) for the nearest bus stop in a densely populated urban environment.

Yelp dataset. The Yelp dataset[1] contains a large collection of geotagged businesses and user reviews across multiple cities. For our analysis, we filtered the data to include only businesses located in the Indianapolis metropolitan area with more than 100 reviews, treating these as points of interest (POIs). This filtering results in a dataset of 810 businesses. The low spatial density of the resulting dataset simulates a realistic scenario in which a user queries for the nearest POI in a sparsely populated or rural area.

[1]The dataset is available at: https://business.yelp.com/data/resources/open-dataset/

Figure 4 illustrates both high-density and low-density POI distributions, as represented by the two datasets.

7.2. Baseline Comparison

We conducted two experiments to evaluate PrivNN against state-of-the-art baselines in terms of accuracy and server response time (CPU time) under varying privacy guarantees. Specifically, we compared PrivNN with CGP [39], an enhanced geo-privacy framework based on differential privacy (DP), and with a recent k-anonymity cloaking method [23]. For PrivNN, we set the partition factor $k = 10$ and the initial search region $C = \min(S_{VP})$.

Privacy vs. Accuracy. This experiment compares PrivNN and CGP in terms of the privacy–accuracy trade-off. We calibrate the differential privacy budget ϵ using the odds-ratio interpretation of geo-indistinguishability: within a specified protection radius, the output probabilities for any two locations differ by at most a fixed multiplicative factor. To align this with PrivNN's guarantee, we convert the smallest Voronoi cell area $\min(S_{VP})$ into an equivalent circular radius and select ϵ accordingly. This anchors the privacy parameter to intuitive spatial metrics (radius/area) while maintaining consistency with geo-indistinguishability [110].

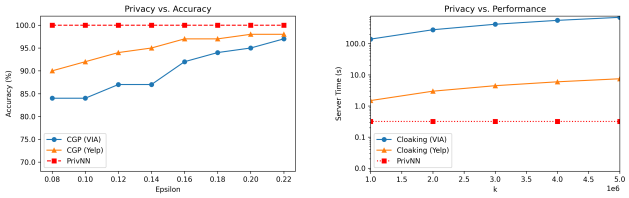
For each ϵ, we issue 100 queries and report the mean accuracy. As shown in Figure 5a, at the matched privacy level of $\epsilon = 0.08$ for the VIA dataset, CGP achieves only around 85% query accuracy, while PrivNN maintains perfect accuracy. For the Yelp dataset, which has lower location density and thus a matched privacy level of $\epsilon = 1.0$, PrivNN continues to outperform CGP, demonstrating significantly higher accuracy under equivalent privacy constraints.

Privacy vs. Performance. The second experiment compares PrivNN to a k-anonymity-based cloaking baseline to evaluate the privacy–performance trade-off. We measure the server's wall-clock time required to process spatial queries. To align the privacy levels of both approaches, we match the adversary's identification probability: specifically, we select k such that the worst-case probability of identifying a user over the entire search region equals $1/\min(S_{VP})$—the inverse of the smallest Voronoi cell area used by PrivNN. For each k, we issue 100 queries and report the mean server-side processing time.

As shown in Figure 5b, matching privacy corresponds to $k = 5 \times 10^6$ for the VIA dataset. At this level, the cloaking method must transmit the locations of $k - 1$ pseudo-users to the server per query, resulting in significantly increased computation time. In contrast, PrivNN processes the same number of queries with an average latency of only 0.32 seconds, demonstrating substantial efficiency under equivalent privacy guarantees.

7.3. Parameter Tuning Analysis

To evaluate performance under various network conditions, our experiments were divided into two configurations: short and long connections. For the short connection configuration, we allocated both instances within the US continent—one in the "US South" region (Texas) as the client, and the other in the "US West" region (Oregon) as the server. For the long connection configuration, we ran one instance in the "US South" region (Texas) as the client and the other in the "Europe West" region (Germany) as the server.



(a) Accuracy comparison with various privacy budget ϵ for CGP.

(b) Performance comparison with various k for cloaking method.

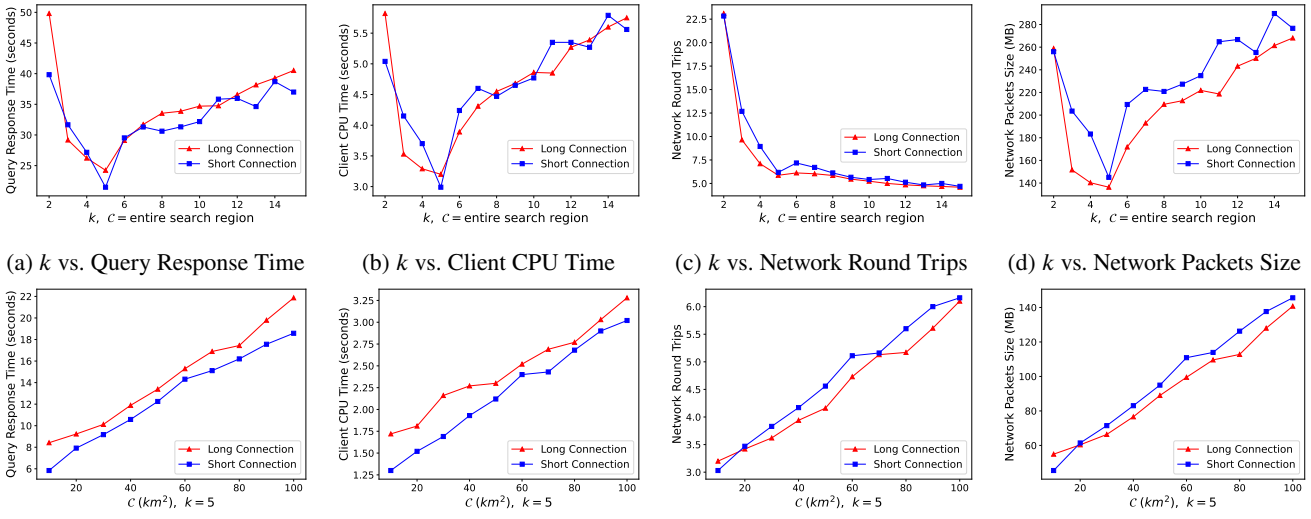
Figure 5: Accuracy and performance comparison between baseline models and PrivNN for two datasets.

We conducted several experiments to evaluate the overhead of PrivNN under various network environments and parameter settings. In the first set of experiments, we analyzed the impact of the partition factor k by varying k values while keeping the initial search region C as the entire search region. In the second set of experiments, we assessed the impact of the initial search region C on the performance of the PrivNN framework, setting $k = 5$ and evaluating performance with different sizes of C.

We measured query response time, client CPU time, total network round trips, and total exchanged network packet size as performance metrics to evaluate time and space overhead in network communication. Each experiment was performed for both short and long connection configurations to allow for comparative analysis. Given the network condition fluctuations and the random sampling employed by the DHVO algorithm in its partitioning process, we simulated 50 queries for each parameter setting and reported the mean values for each evaluation metric.

Figure 6 illustrates the performance of the PrivNN framework with varying values of the partition factor k and the initial search region C. In the top row of Figure 6, for each value of k (x-axis), we performed 50 queries where the location of the query point was randomly selected, and we averaged the results. Note that the initial search region C is set to the maximum, i.e., the entire search region, to evaluate the impact of k independently of C.

Two key observations can be made from the top row in Figure 6. First, the short and long connections exhibit consistency in relative performance across all metrics, indicating that network conditions, such as latency, do not significantly impact performance. The second observation is that the query response time, client CPU time, and network packet size reach their lowest values at $k = 5$, then increase as k continues to rise. This occurs because, as k increases, more points are sampled at each DHVO layer, resulting in fewer round trips needed to reach the base layer. However, a higher k value also increases the size of network packets



(a) k vs. Query Response Time (b) k vs. Client CPU Time (c) k vs. Network Round Trips (d) k vs. Network Packets Size

(e) $\mathcal{C}$ vs. Query Response Time (f) $\mathcal{C}$ vs. Client CPU Time (g) $\mathcal{C}$ vs. Network Round Trips (h) $\mathcal{C}$ vs. Network Packets Size

Figure 6: Performance metrics of PrivNN measured for short and long connections, varying parameters k and initial search region $\mathcal{C}$. The top row demonstrates the impact of different values of k on query response time, client CPU time, total network round trips, and total exchanged network packet size, with $\mathcal{C}$ encompassing the entire search region to provide the highest level of privacy protection. The bottom row shows the effect of varying $\mathcal{C}$ while keeping $k = 5$, illustrating the practicality of PrivNN with reduced but still sufficient privacy protection.

and the computing time required for the client to decrypt and find the shortest distance for each round. This observation is important as it suggests that setting $k = 5$ is a good heuristic for minimizing PrivNN's overhead, independent of network conditions.

We adopted the heuristic of setting $k = 5$ for the subsequent experiments to evaluate the impact of the initial search region $\mathcal{C}$ on the performance of the PrivNN framework. In the lower row of Figure 6, we set $k = 5$ and varied the size of the initial search region $\mathcal{C}$ from 10 km^2 to 100 km^2 to assess the impact of $\mathcal{C}$ while keeping k constant. The mean values of the performance metrics for different $\mathcal{C}$ values are listed in Table 2. The results show that all metrics linearly increase as the size of the initial search region $\mathcal{C}$ increases. This is expected, as a larger initial search region $\mathcal{C}$ results in more DHVO layers being constructed, requiring more rounds of communication between the client and server. The results indicate that the initial search region $\mathcal{C}$ significantly affects the performance of the PrivNN framework. As shown in Equation (3), the size of the smallest Voronoi polygon dictates PrivNN's privacy bound. In practice, even a region of 10 km^2 is much larger than the size of the smallest Voronoi polygon, thus it does not reduce the theoretical level of privacy protection. In the context of the evaluated dataset, smaller-sized polygons are within the downtown area of San Antonio, where we argue that a lower privacy protection level is within tolerance. From Table 2, we observed that the performance metrics remain practical even for larger $\mathcal{C}$ values, indicating that PrivNN can provide an excellent tradeoff between privacy and performance in real-world scenarios.

Table 2
Metrics Mean for Various $\mathcal{C}$ Values ($k = 5$)

$\mathcal{C}$ (km^2)	Query Response Time (s)		Client CPU Time (s)		Network Round Trips		Network Packet Size (MB)	
	Short	Long	Short	Long	Short	Long	Short	Long
10	5.84	8.42	1.30	1.72	3.03	3.20	45.45	54.93
20	7.92	9.23	1.52	1.81	3.47	3.42	61.43	60.35
30	9.17	10.12	1.69	2.16	3.83	3.62	71.47	66.37
40	10.58	11.89	1.93	2.27	4.17	3.94	83.03	76.60
50	12.25	13.39	2.12	2.30	4.56	4.16	94.97	88.97
60	14.32	15.29	2.40	2.52	5.11	4.73	110.88	99.51
70	15.11	16.89	2.43	2.69	5.16	5.13	113.93	109.59
80	16.20	17.44	2.68	2.77	5.60	5.17	126.27	112.77
90	17.56	19.79	2.90	3.03	6.00	5.61	137.64	128.01
100	18.58	21.87	3.02	3.28	6.16	6.10	145.60	140.75

7.4. Operational Overhead Analysis

To provide a fine-grained analysis of PrivNN's end-to-end overhead, we measured both the latencies of the CKKS operations used in our implementation and the precomputation time for the Voronoi diagram.

CKKS Operations Overhead. Table 3 reports the per-operation latencies and the amortized per-query costs derived from these measurements. All measurements were performed on the GCP instance described in Section 7.1. We executed 100 queries on the VIA dataset with partition factor $k = 10$ and initial search region $\mathcal{C} = \min(S_{VP})$.

Voronoi Precomputation Overhead. The Voronoi precomputation required by PrivNN is modest in practice. On the GCP instance described in Section 7.1, constructing the full Voronoi diagram took approximately 5 minutes for the Yelp dataset and about 18 minutes for the VIA dataset. Theoretically, planar Voronoi/Delaunay construction admits a tight bound of $\Theta(n \log n)$ in two dimensions, where n

Table 3
CKKS Operations Overhead ($N = 8192,\ \log_2 q_L = 218$)

(A) One-time client operation (ms)	
Operation	Time/Op
KeyGen	190

(B) Per-query CKKS operations (ms)		
Operation	Time/Op	Amortized Per-Query
Encode	0.51	15.81
Enc	4.63	4.63
Dec	0.85	20.22
HAdd	0.58	17.98
HMultPlain	0.63	19.53

is the number of generating points. In practice, SciPy's implementation typically runs in $O(n \log n)$ time but can degrade to $O(n^2)$ [111–113]. Because the global Voronoi diagram is computed offline and reused for many queries, this one-time cost is easily amortized. For frequent POI updates, we recommend either periodic rebuilds or localized recomputation: update the Voronoi diagram only within an expanded bounding box around the modified points, rather than recomputing the entire diagram.

8. Conclusion

In this paper, we proposed PrivNN, a privacy-preserving two-party computation framework that enables users to perform spatial NN queries from a server without disclosing their exact location. To efficiently search for NNs in spatial datasets, we introduced a novel spatial NN search algorithm, DHVO, which iteratively overlays multi-granular Voronoi diagrams to refine the search region. Our complexity analysis showed that DHVO offers several advantages over existing methods used for spatial NN search.

We also proposed a client-server communication protocol that enables DHVO to be executed in a privacy-preserving manner using encrypted query data. A rigorous security and privacy analysis of PrivNN was presented, demonstrating its robustness against semi-honest adversaries under the known background knowledge model, and establishing its formal privacy guarantees. Our comparison experiment results show that PrivNN consistently outperforms existing approaches in both accuracy and performance under equivalent privacy settings. We further conducted parameter tuning experiments, which revealed that the empirical parameter k can be strategically optimized, and that PrivNN remains efficient even when operating over large initial search regions. We believe that PrivNN can be effectively extended to a wide range of LBS applications, such as location-based recommendations, geofencing, and emergency services, thereby protecting user privacy while maintaining the functionality and practicality of these applications.

We note two primary, unresolved challenges in our work. First, privacy can degrade in very dense spatial datasets because ambiguity is reduced and anonymity is compromised—an intrinsic limitation of nearest-neighbor queries and current practical HE deployments. Second, the interactive pruning loop leaks access-pattern signals (the indices the client returns each round); although distance values remain encrypted, these index replies can be recorded and correlated by a curious server and so aid re-identification. Possible mitigations range from lightweight, low-cost approaches—returning a small set of dummy indices or using randomized index reporting (which trades some utility for reduced leakage)—to stronger but costlier defenses such as PIR, secure MPC for arg-min function, or trusted execution environments. We hope these challenges will be addressed in future work.

References

[1] Electronic Frontier Foundation, "Locational privacy," 2024, accessed: 2025-03-30. [Online]. Available: https://www.eff.org/issues/location-privacy

[2] Viterbi Conversations in Ethics, "The future of location-based services and the implications of user privacy," *Viterbi Conversations in Ethics*, 2024, accessed: 2025-03-30. [Online]. Available: https://vce.usc.edu/volume-3-issue-1/the-future-of-location-based-services-and-the-implications-of-user-privacy/

[3] S. Gray, "A closer look at location data: Privacy and pandemics," 2024, accessed: 2025-03-30. [Online]. Available: https://fpf.org/2024/05/24/a-closer-look-at-location-data-privacy-and-pandemics/

[4] M. Herrmann, M. Hildebrandt, L. Tielemans, and C. Diaz, "Privacy in location-based services: An interdisciplinary approach," *SCRIPTed*, 2024, accessed: 2025-03-30. [Online]. Available: https://script-ed.org/2024/05/31/privacy-in-location-based-services-an-interdisciplinary-approach/

[5] K. Cohen, "Location, health, and other sensitive information: Ftc committed to fully enforcing the law against illegal use and sharing of highly sensitive data," 2022, accessed: 2025-03-30. [Online]. Available: https://www.ftc.gov/business-guidance/blog/2022/07/location-health-and-other-sensitive-information-ftc-committed-fully-enforcing-law-against-illegal

[6] G. Drakonakis, N. Laoutaris, E. P. Markatos, P. Papadopoulos, and S. Ioannidis, "Please forget where i was last summer: The privacy risks of public location (meta)data," in *Proceedings of the Network and Distributed System Security Symposium (NDSS)*, 2019, accessed: 2025-03-30. [Online]. Available: https://www.ndss-symposium.org/wp-content/uploads/2019/02/ndss2019_01A-6_Drakonakis_paper.pdf

[7] C. Boyd, "Strava heatmap loophole may reveal users' home addresses," 2023, accessed: 2025-03-30. [Online]. Available: https://www.malwarebytes.com/blog/news/2023/06/strava-heatmap-loophole-may-reveal-users-home-addresses

[8] K. Conger and K. Roose, "Uber Investigating Breach of Its Computer Systems," *The New York Times*, Sep. 2022. [Online]. Available: https://www.nytimes.com/2022/09/15/technology/uber-hacking-breach.html

[9] T. Brewster, "FTC: Uber Failed To Protect 100,000 Drivers In 2014 Hack," *Forbes*, Aug. 2017. [Online]. Available: https://www.forbes.com/sites/thomasbrewster/2017/08/15/uber-settles-ftc-complaint-over-secuirty-and-privacy/?sh=39812c3c88da

[10] E. Newcomer, "Uber Paid Hackers to Delete Stolen Data on 57 Million People," *Bloomberg*, Nov. 2017. [Online]. Available: https://www.bloomberg.com/news/articles/2017-11-21/uber-concealed-cyberattack-that-exposed-57-million-people-s-data#xj4y7vzkg

[11] Federal Trade Commission, "Uber settles FTC allegations that it made deceptive privacy and data security claims," *Federal Trade Commission Press Release*, Aug. 2017, accessed: 2025-03-30. [Online]. Available: https://www.ftc.gov/news-events/news/press-releases/2017/08/uber-settles-ftc-allegations-it-made-deceptive-privacy-data-security-claims

[12] Z. Khan, "Hundreds of Uber Eats User records leaked on Dark Web," *Hack Read*, Aug. 2020. [Online]. Available: https://www.hackread.com/uber-eats-user-records-leaked-dark-web/

[13] K. Carlon, "Runkeeper is secretly tracking you around the clock and sending your data to advertisers," *Android Authority*, May 2016. [Online]. Available: https://www.androidauthority.com/runkeeper-user-location-tracking-data-advertisers-692346/

[14] C.-Y. Chow, M. F. Mokbel, and W. G. Aref, "Casper*: Query processing for location services without compromising privacy," *ACM Trans. Database Syst.*, vol. 34, no. 4, dec 2009. [Online]. Available: https://doi.org/10.1145/1620585.1620591

[15] C. Dwork, "Differential privacy," in *Proceedings of the 33rd International Colloquium on Automata, Languages and Programming (ICALP)*. Springer, 2006, pp. 1–12.

[16] M. Gruteser and D. Grunwald, "Anonymous usage of location-based services through spatial and temporal cloaking," in *Proceedings of the 1st International Conference on Mobile Systems, Applications and Services*, ser. MobiSys '03. New York, NY, USA: Association for Computing Machinery, 2003, p. 31–42. [Online]. Available: https://doi.org/10.1145/1066116.1189037

[17] H. Kido, Y. Yanagisawa, and T. Satoh, "Protection of location privacy using dummies for location-based services," in *21st International Conference on Data Engineering Workshops (ICDEW'05)*, 2005, pp. 1248–1248.

[18] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *Proceedings of the forty-first annual ACM symposium on Theory of computing*. ACM, 2009, pp. 169–178.

[19] A. Boldyreva and T. Tang, "Privacy-preserving approximate k-nearest-neighbors search that hides access, query and volume patterns," *IACR Cryptol. ePrint Arch.*, p. 816, 2021. [Online]. Available: https://eprint.iacr.org/2021/816

[20] M. Gruteser and D. Grunwald, "Anonymous usage of Location-Based services through spatial and temporal cloaking," in *First International Conference on Mobile Systems, Applications, and Services (MobiSys2003)*. San Francisco, CA: USENIX Association, May 2003. [Online]. Available: https://www.usenix.org/conference/mobisys2003/anonymous-usage-location-based-services-through-spatial-and-temporal-cloaking

[21] M. F. Mokbel, C.-Y. Chow, and W. G. Aref, "The new casper: query processing for location services without compromising privacy," in *Proceedings of the 32nd International Conference on Very Large Data Bases*, ser. VLDB '06. VLDB Endowment, 2006, p. 763–774.

[22] B. Gedik and L. Liu, "Location privacy in mobile systems: A personalized anonymization model," in *25th IEEE International Conference on Distributed Computing Systems (ICDCS'05)*, 2005, pp. 620–629.

[23] H. N. S. S. Jagarlapudi, S. Lim, J. Chae, G. S. Choi, and C. Pu, "Drone helps privacy: Sky caching assisted k-anonymity in spatial querying," *IEEE Systems Journal*, vol. 16, no. 4, pp. 6360–6371, 2022.

[24] L. Xing, D. Zhang, H. Wu, H. Ma, and X. Zhang, "Distributed k-anonymous location privacy protection algorithm based on interest points and user social behavior," *Electronics*, vol. 12, no. 11, p. 2446, 2023.

[25] C. Gutiérrez-Soto, P. Galdames, C. Faúndez, and C. Durán-Faúndez, "Location-query-privacy and safety cloaking schemes for continuous location-based services," *Mobile Information Systems*, vol. 2022, p. 5191041, 2022.

[26] M. Zhang, X. Li, Y. Miao, B. Luo, Y. Ren, and S. Ma, "Peak: Privacy-enhanced incentive mechanism for distributed k-anonymity in lbs," *IEEE Transactions on Knowledge and Data Engineering*, 2024, early Access.

[27] Z. Liu, D. Miao, R. Li, Y. Liu, and X. Li, "Cache-based privacy protection scheme for continuous location query," *Entropy*, vol. 25, no. 2, p. 201, 2023.

[28] L. Sweeney, "k-anonymity: a model for protecting privacy," *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, vol. 10, no. 5, p. 557–570, oct 2002. [Online]. Available: https://doi.org/10.1142/S0218488502001648

[29] H. Liu, X. Li, H. Li, J. Ma, and X. Ma, "Spatiotemporal correlation-aware dummy-based privacy protection scheme for location-based services," in *IEEE INFOCOM 2017 - IEEE Conference on Computer Communications*, 2017, pp. 1–9.

[30] X. Zhang, J. Wang, M. Shu, Y. Wang, M. Pan, and Z. Han, "Tpp: Trajectory privacy preservation against tensor voting based inference attacks," *IEEE Access*, vol. 6, pp. 77 975–77 985, 2018.

[31] B. Niu, Q. Li, X. Zhu, G. Cao, and H. Li, "Achieving k-anonymity in privacy-aware location-based services," in *IEEE INFOCOM 2014 - IEEE Conference on Computer Communications*, 2014, pp. 754–762.

[32] G. Sun, V. Chang, M. Ramachandran, Z. Sun, G. Li, H. Yu, and D. Liao, "Efficient location privacy algorithm for internet of things (iot) services and applications," *Journal of Network and Computer Applications*, vol. 89, pp. 3–13, 2017, emerging Services for Internet of Things (IoT). [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1084804516302429

[33] K. C. Lee, W.-C. Lee, H. V. Leong, and B. Zheng, "Navigational path privacy protection: navigational path privacy protection," in *Proceedings of the 18th ACM Conference on Information and Knowledge Management*, ser. CIKM '09. New York, NY, USA: Association for Computing Machinery, 2009, p. 691–700. [Online]. Available: https://doi.org/10.1145/1645953.1646041

[34] C. Dwork and A. Roth, "The algorithmic foundations of differential privacy," *Found. Trends Theor. Comput. Sci.*, vol. 9, no. 3–4, p. 211–407, aug 2014. [Online]. Available: https://doi.org/10.1561/0400000042

[35] L. Zhou, L. Yu, S. Du, H. Zhu, and C. Chen, "Achieving differentially private location privacy in edge-assistant connected vehicles," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4472–4481, 2019.

[36] L. Luo, Z. Han, C. Xu, and G. Zhao, "A geo-indistinguishable location privacy preservation scheme for location-based services in vehicular networks," in *Algorithms and Architectures for Parallel Processing*, S. Wen, A. Zomaya, and L. T. Yang, Eds. Cham: Springer International Publishing, 2020, pp. 610–623.

[37] C. Qiu, A. Squicciarini, C. Pang, N. Wang, and B. Wu, "Location privacy protection in vehicle-based spatial crowdsourcing via geo-indistinguishability," *IEEE Transactions on Mobile Computing*, vol. 21, no. 7, pp. 2436–2450, 2022.

[38] T. Cunningham, G. Cormode, H. Ferhatosmanoglu, and D. Srivastava, "Real-world trajectory sharing with local differential privacy," *Proc. VLDB Endow.*, vol. 14, no. 11, p. 2283–2295, jul 2021. [Online]. Available: https://doi.org/10.14778/3476249.3476280

[39] Y. Liang and K. Yi, "Concentrated geo-privacy," in *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security (CCS '23)*. Copenhagen, Denmark: Association for Computing Machinery, 2023, pp. 1934–1948. [Online]. Available: https://doi.org/10.1145/3576915.3623068

[40] H. Wang, H. Hong, L. Xiong, Z. Qin, and Y. Hong, "L-srr: Local differential privacy for location-based services with staircase randomized response," in *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security (CCS '22)*. Los Angeles, CA, USA: Association for Computing Machinery, 2022, pp. 2809–2823. [Online]. Available: https://doi.org/10.1145/3548606.3560636

[41] Y. Han, Z. Li, J. Zhang, Z. Wu, and Y. Ding, "Location nearest neighbor query scheme in edge computing based on differential privacy," in *Web Information Systems Engineering – WISE 2024*, ser. Lecture Notes in Computer Science, vol. 15440. Singapore: Springer, 2025, pp. 95–106. [Online]. Available: https://doi.org/10.1007/978-981-96-0576-7_8

[42] C. Dwork, "Differential privacy," in *Automata, Languages and Programming*, M. Bugliesi, B. Preneel, V. Sassone, and I. Wegener, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 1–12.

[43] Y. Zhu, X. Yu, M. Chandraker, and Y.-X. Wang, "Private-knn: Practical differential privacy for computer vision," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 11 851–11 859.

[44] M. E. Gursoy, A. Inan, M. E. Nergiz, and Y. Saygin, "Differentially private nearest neighbor classification," *Data Min. Knowl. Discov.*, vol. 31,

no. 5, p. 1544–1575, sep 2017. [Online]. Available: https://doi.org/10.1007/s10618-017-0532-z

[45] S. Canard, B. Olivier, and T. Quertier, "Differentially private instance-based noise mechanisms in practice," in *2017 15th Annual Conference on Privacy, Security and Trust (PST)*, 2017, pp. 105–10 509.

[46] A. Khoshgozaran and C. Shahabi, "Blind evaluation of nearest neighbor queries using space transformation to preserve location privacy," in *Advances in Spatial and Temporal Databases*, D. Papadias, D. Zhang, and G. Kollios, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 239–257.

[47] D. Lin, E. Bertino, R. Cheng, and S. Prabhakar, "Position transformation: a location privacy protection method for moving objects," in *Proceedings of the SIGSPATIAL ACM GIS 2008 International Workshop on Security and Privacy in GIS and LBS*, ser. SPRINGL '08. New York, NY, USA: Association for Computing Machinery, 2008, p. 62–71. [Online]. Available: https://doi.org/10.1145/1503402.1503414

[48] A. Khoshgozaran, H. Shirani-Mehr, and C. Shahabi, "Blind evaluation of location based queries using space transformation to preserve location privacy," *Geoinformatica*, vol. 17, no. 4, p. 599–634, oct 2013. [Online]. Available: https://doi.org/10.1007/s10707-012-0172-9

[49] B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan, "Private information retrieval," in *Proceedings of IEEE 36th Annual Foundations of Computer Science*, 1995, pp. 41–50.

[50] G. Ghinita, P. Kalnis, A. Khoshgozaran, C. Shahabi, and K.-L. Tan, "Private queries in location based services: anonymizers are not necessary," in *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '08. New York, NY, USA: Association for Computing Machinery, 2008, p. 121–132. [Online]. Available: https://doi.org/10.1145/1376616.1376631

[51] A. Khoshgozaran, C. Shahabi, and H. Shirani-Mehr, "Location privacy: going beyond k-anonymity, cloaking and anonymizers," *Knowledge and Information Systems*, vol. 26, no. 3, pp. 435–465, 2011. [Online]. Available: https://doi.org/10.1007/s10115-010-0286-z

[52] W. K. Wong, D. W.-l. Cheung, B. Kao, and N. Mamoulis, "Secure knn computation on encrypted databases," in *Proceedings of the 2009 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '09. New York, NY, USA: Association for Computing Machinery, 2009, p. 139–152. [Online]. Available: https://doi.org/10.1145/1559845.1559862

[53] H. Hu, J. Xu, C. Ren, and B. Choi, "Processing private queries over untrusted data cloud through privacy homomorphism," in *2011 IEEE 27th International Conference on Data Engineering*, 2011, pp. 601–612.

[54] Y. Elmehdwi, B. K. Samanthula, and W. Jiang, "Secure k-nearest neighbor query over encrypted data in outsourced environments," in *2014 IEEE 30th International Conference on Data Engineering*, 2014, pp. 664–675.

[55] R. Xu, K. Morozov, Y. Yang, J. Zhou, and T. Takagi, "Privacy-preserving k-nearest neighbour query on outsourced database," in *Information Security and Privacy*, J. K. Liu and R. Steinfeld, Eds. Cham: Springer International Publishing, 2016, pp. 181–197.

[56] J. L. Bentley, "Multidimensional binary search trees used for associative searching," *Commun. ACM*, vol. 18, no. 9, p. 509–517, sep 1975. [Online]. Available: https://doi.org/10.1145/361002.361007

[57] Y.-C. Hsu, C.-H. Hsueh, and J.-L. Wu, "A privacy preserving cloud-based k-nn search scheme with lightweight user loads," *Computers*, vol. 9, no. 1, 2020. [Online]. Available: https://www.mdpi.com/2073-431X/9/1/1

[58] A. Guttman, "R-trees: a dynamic index structure for spatial searching," in *Proceedings of the 1984 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '84. New York, NY, USA: Association for Computing Machinery, 1984, p. 47–57. [Online]. Available: https://doi.org/10.1145/602259.602266

[59] Y. Qi and M. J. Atallah, "Efficient privacy-preserving k-nearest neighbor search," in *Proceedings of the 2008 The 28th International Conference on Distributed Computing Systems*, ser. ICDCS '08. USA: IEEE Computer Society, 2008, p. 311–319. [Online]. Available: https://doi.org/10.1109/ICDCS.2008.79

[60] B. Wang, Y. Hou, and M. Li, "Practical and secure nearest neighbor search on encrypted large-scale data," in *IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications*, 2016, pp. 1–9.

[61] N. Cao, C. Wang, M. Li, K. Ren, and W. Lou, "Privacy-Preserving Multi-Keyword Ranked Search over Encrypted Cloud Data," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 1, pp. 222–233, Jan. 2014. [Online]. Available: http://ieeexplore.ieee.org/document/6674958/

[62] Peng Wang and C. V. Ravishankar, "Secure and efficient range queries on outsourced databases using Rp-trees," in *2013 IEEE 29th International Conference on Data Engineering (ICDE)*. Brisbane, QLD: IEEE, Apr. 2013, pp. 314–325. [Online]. Available: http://ieeexplore.ieee.org/document/6544835/

[63] B. Wang, Y. Hou, M. Li, H. Wang, and H. Li, "Maple: scalable multi-dimensional range search over encrypted cloud data with tree-based index," in *Proceedings of the 9th ACM symposium on Information, computer and communications security*. Kyoto Japan: ACM, Jun. 2014, pp. 111–122. [Online]. Available: https://dl.acm.org/doi/10.1145/2590296.2590305

[64] R. Li, A. X. Liu, A. L. Wang, and B. Bruhadeshwar, "Fast and Scalable Range Query Processing With Strong Privacy Protection for Cloud Computing," *IEEE/ACM Transactions on Networking*, vol. 24, no. 4, pp. 2305–2318, Aug. 2016. [Online]. Available: https://ieeexplore.ieee.org/document/7222488/

[65] M. Barhamgi, D. Benslimane, Y. Amghar, N. Cuppens-Boulahia, and F. Cuppens, "PrivComp: a privacy-aware data service composition system," in *Proceedings of the 16th International Conference on Extending Database Technology*. Genoa Italy: ACM, Mar. 2013, pp. 757–760. [Online]. Available: https://dl.acm.org/doi/10.1145/2452376.2452473

[66] G. Xu, H. Li, Y. Dai, K. Yang, and X. Lin, "Enabling Efficient and Geometric Range Query With Access Control Over Encrypted Spatial Data," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 4, pp. 870–885, Apr. 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8452984/

[67] W. Yang, Y. Geng, L. Li, X. Xie, and L. Huang, "Achieving Secure and Dynamic Range Queries Over Encrypted Cloud Data," *IEEE Transactions on Knowledge and Data Engineering*, pp. 1–1, 2020. [Online]. Available: https://ieeexplore.ieee.org/document/9050504/

[68] F. Aurenhammer, "Voronoi diagrams—a survey of a fundamental geometric data structure," *ACM Comput. Surv.*, vol. 23, no. 3, p. 345–405, sep 1991. [Online]. Available: https://doi.org/10.1145/116873.116880

[69] A. Okabe, B. Boots, K. Sugihara, and S. N. Chiu, *Spatial Tessellations, Concepts and Applications of Voronoi Diagrams*, 2nd ed., ser. Wiley Series in Probability and Statistics. Chichester: John Wiley and Sons Ltd, 2000.

[70] J. B. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, vol. 1. Berkeley: University of California Press, 1967, pp. 281–297.

[71] C. Boura, N. Gama, and M. Georgieva, "Chimera: a unified framework for B/FV, TFHE and HEAAN fully homomorphic encryption and predictions for deep learning," Cryptology ePrint Archive, Report 2018/758, Tech. Rep., 2018.

[72] J. H. Cheon, K. Han, S. Hong, H. J. Kim, J. Kim, S. Kim, H. Seo, H. Shim, and Y. Song, "Toward a secure drone system: Flying with real-time homomorphic authenticated encryption," *IEEE Access*, vol. 6, pp. 24 325–24 339, 2018.

[73] R. Gilad-Bachrach, N. Dowlin, K. Laine, K. Lauter, M. Naehrig, and J. Wernsing, "Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy," in *International Conference on Machine Learning*, 2016, pp. 201–210.

[74] A. Kim, Y. Song, M. Kim, K. Lee, and J. H. Cheon, "Logistic regression model training based on the approximate homomorphic encryption," *BMC medical genomics*, vol. 11, no. 4, pp. 23–31, 2018.

[75] J. W. Bos, K. Lauter, J. Loftus, and M. Naehrig, "Improved security for a ring-based fully homomorphic encryption scheme," in *IMA International Conference on Cryptography and Coding*. Springer, 2013, pp. 45–64.

[76] Z. Brakerski, "Fully homomorphic encryption without modulus switching from classical GapSVP," in *Annual Cryptology Conference*. Springer, 2012, pp. 868–886.

[77] J. H. Cheon, A. Kim, M. Kim, and Y. Song, "Homomorphic encryption for arithmetic of approximate numbers," in *Advances in Cryptology - ASIACRYPT 2017 - 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part I*, ser. Lecture Notes in Computer Science, T. Takagi and T. Peyrin, Eds., vol. 10624. Springer, 2017, pp. 409–437.

[78] I. Chillotti, N. Gama, M. Georgieva, and M. Izabachene, "Faster fully homomorphic encryption: Bootstrapping in less than 0.1 seconds," in *International Conference on the Theory and Application of Cryptology and Information Security*. Springer, 2016, pp. 3–33.

[79] A. Costache and N. P. Smart, "Which ring based somewhat homomorphic encryption scheme is best?" in *Cryptographers' Track at the RSA Conference*. Springer, 2016, pp. 325–340.

[80] L. Ducas and D. Micciancio, "Fhew: Bootstrapping homomorphic encryption in less than a second," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2015, pp. 617–640.

[81] M. Van Dijk, C. Gentry, S. Halevi, and V. Vaikuntanathan, "Fully homomorphic encryption over the integers," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2010, pp. 24–43.

[82] J. Fan and F. Vercauteren, "Somewhat practical fully homomorphic encryption," IACR Cryptology ePrint Archive, Tech. Rep., 2012.

[83] C. Gentry, S. Halevi, and N. P. Smart, "Homomorphic evaluation of the aes circuit," in *Annual Cryptology Conference*. Springer, 2012, pp. 850–867.

[84] C. Gentry, A. Sahai, and B. Waters, "Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based," in *Annual Cryptology Conference*. Springer, 2013, pp. 75–92.

[85] M. Ogburn, C. Turner, and P. Dahal, "Homomorphic encryption," *Procedia Computer Science*, vol. 20, pp. 502–509, 2013.

[86] F. Modler and M. Kreh, *Tutorium Analysis 1 und Lineare Algebra 1*. Berlin/Heidelberg, Germany: Springer, 2011.

[87] C. Gentry, "Computing arbitrary functions of encrypted data," *Communications of the ACM*, vol. 53, no. 3, pp. 97–105, 2010.

[88] Z. Brakerski and V. Vaikuntanathan, "Fully homomorphic encryption from ring-lwe and security for key dependent messages," in *Annual Cryptology Conference*. Berlin, Heidelberg: Springer, 2011, pp. 505–524.

[89] J. Müller-Quade, M. Huber, and T. Nilges, "Daten verschlüsselt speichern und verarbeiten in der cloud," *Datenschutz und Datensicherheit - DuD*, vol. 39, no. 8, pp. 531–535, 2015.

[90] M. Kim, Y. Song, S. Wang, Y. Xia, and X. Jiang, "Secure logistic regression based on homomorphic encryption: Design and evaluation," *JMIR Medical Informatics*, vol. 6, no. 2, p. e19, 2018.

[91] J. H. Cheon, K. Han, S. Hong, H. J. Kim, J. Kim, S. Kim, H. Seo, H. Shim, and Y. Song, "Toward a secure drone system: Flying with real-time homomorphic authenticated encryption," *IEEE Access*, vol. 6, pp. 24 325–24 339, 2018. [Online]. Available: https://doi.org/10.1109/ACCESS.2018.2819189

[92] H. Chen, I. Chillotti, and Y. Song, "Improved bootstrapping for approximate homomorphic encryption," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2019, pp. 34–54.

[93] J. H. Cheon, K. Han, A. Kim, M. Kim, and Y. Song, "Bootstrapping for approximate homomorphic encryption," in *Advances in Cryptology - EUROCRYPT 2018 - 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Proceedings, Part I*. Springer, 2018, pp. 360–384. [Online]. Available: https://doi.org/10.1007/978-3-319-78381-9_14

[94] H. Chen, I. Iliashenko, and K. Laine, "When heaan meets fv: A new somewhat homomorphic encryption with reduced memory overhead," in *Cryptography and Coding - 18th IMA International Conference, IMACC 2021, Virtual Event, December 14-15, 2021, Proceedings*, ser. Lecture Notes in Computer Science, vol. 13129. Springer, 2021, pp. 265–285.

[95] J. H. Cheon, K. Han, A. Kim, M. Kim, and Y. Song, "A full rns variant of approximate homomorphic encryption," in *Selected Areas in Cryptography - SAC 2018*. Springer, 2018, pp. 347–368. [Online]. Available: https://dblp.org/rec/conf/sacrypt/CheonHKKS18

[96] S. Fortune, "A sweepline algorithm for voronoi diagrams," *Algorithmica*, vol. 2, no. 1-4, pp. 153–174, 1987.

[97] S. Goldwasser and S. Micali, "Probabilistic encryption," *Journal of Computer and System Sciences*, vol. 28, no. 2, pp. 270–299, April 1984. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0022000084900709

[98] V. Lyubashevsky, C. Peikert, and O. Regev, "On ideal lattices and learning with errors over rings," in *EUROCRYPT*, 2010, pp. 1–23.

[99] O. Goldreich, *Foundations of Cryptography: Volume 2, Basic Applications*. New York, NY, USA: Cambridge University Press, 2009.

[100] M. Chase and S. Kamara, "Structured encryption and controlled disclosure," in *Proceedings of the 16th International Conference on Theory and Application of Cryptology and Information Security (ASIACRYPT)*. Springer, 2010, pp. 577–594.

[101] R. Curtmola, J. Garay, S. Kamara, and R. Ostrovsky, "Searchable symmetric encryption: Improved definitions and efficient constructions," in *Proceedings of the 13th ACM Conference on Computer and Communications Security (CCS)*. ACM, 2006, pp. 79–88.

[102] E. Ben-Sasson, A. Chiesa, E. Tromer, and M. Virza, "libsnark: a c++ library for zksnark proofs," 2019, https://github.com/scipr-lab/libsnark.

[103] J. Groth, "On the size of pairing-based non-interactive arguments," in *Annual international conference on the theory and applications of cryptographic techniques*. Springer, 2016, pp. 305–326.

[104] R. Gennaro and D. Wichs, "Homomorphic macs: Mac-based integrity for network coding," in *International Conference on the Theory and Application of Cryptology and Information Security*. Springer, 2013, pp. 539–557.

[105] C. Dwork, "Re-identification in the natural experiment setting," *Journal of Privacy and Confidentiality*, vol. 2, no. 1, 2010. [Online]. Available: https://journalprivacyconfidentiality.org/index.php/jpc/article/view/584

[106] "Microsoft SEAL (release 4.0)," https://github.com/Microsoft/SEAL, Mar. 2022, microsoft Research, Redmond, WA.

[107] A. Ibarrondo and A. Viand, "Pyfhel: Python for homomorphic encryption libraries," in *Proceedings of the 9th on Workshop on Encrypted Computing & Applied Homomorphic Cryptography*, 2021, pp. 11–16.

[108] M. R. Albrecht, "A sage module for estimating the concrete security of learning with errors instances," https://bitbucket.org/malb/lwe-estimator, 2017, accessed: 2025-03-30.

[109] VIA Metropolitan Transit, "Resources for developers - VIA metropolitan transit," 2024, accessed: 2025-03-30. [Online]. Available: https://www.viainfo.net/developers-resources/

[110] M. E. Andrés, N. E. Bordenabe, K. Chatzikokolakis, and C. Palamidessi, "Geo-indistinguishability: Differential privacy for location-based systems," in *Proceedings of the 2013 ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, 2013, pp. 901–914.

[111] S. Fortune, "A sweepline algorithm for voronoi diagrams," in *Proceedings of the Second Annual Symposium on Computational Geometry*. ACM, 1987, pp. 313–322, establishes the $\Theta(n \log n)$ bound for planar Voronoi construction.

[112] M. de Berg, O. Cheong, M. van Kreveld, and M. Overmars, *Computational Geometry: Algorithms and Applications*, 3rd ed. Berlin, Germany: Springer, 2008, standard reference covering Delaunay triangulations and Voronoi diagrams.

[113] C. B. Barber, D. P. Dobkin, and H. Huhdanpää, "The quickhull algorithm for convex hulls," *ACM Transactions on Mathematical Software*, vol. 22, no. 4, pp. 469–483, 1996, qhull implements Quickhull and related routines used by many libraries (e.g., SciPy) for Delaunay/Voronoi.