Private Approximate Nearest Neighbor Search for Vector Database Querying

Sajani Vithana[1], Martina Cardone[2], and Flavio P. Calmon[1]

[1]School of Engineering and Applied Sciences, Harvard University (emails: sajani@seas.harvard.edu, flavio@seas.harvard.edu)
[2]Department of Electrical and Computer Engineering, University of Minnesota, Minneapolis (email: mcardone@umn.edu)

***Abstract*—We consider the problem of private approximate nearest neighbor (ANN) search. A user seeks the closest vector to a target query q among M vectors stored in a system of N non-colluding databases. The user aims to retrieve the ANN without revealing information about q to any of the N databases. We provide an information-theoretic formulation of the problem and propose a scheme based on a tree-structured ANN search mechanism. The proposed scheme uses a coding-theoretic approach to traverse the branch in the tree structure that leads to the approximately closest vector to q while guaranteeing perfect information-theoretic privacy. We prove that our approach achieves a communication cost of $O(N^2M^{\frac{1}{N-1}})$ for N databases. For large M, this communication cost is lower than competing cryptographic ANN search protocols.**

I. Introduction

Approximate nearest neighbor (ANN) search [1], [2] aims to retrieve the closest point within a dataset to a given target query. ANN search is used in a multitude of applications ranging from recommendation systems [3], image retrieval [4], anomaly detection [5], and computational biology [6].

Recently, the emergence of transformer models [7] has led to several new methods for creating high-dimensional vector representations (embeddings) of text, images, speech, and videos that capture the semantics of the underlying data [8], [9]. Vector embedding models map data from different modalities into a common vector space such that samples with similar semantics are positioned "close together." For instance, Open AI's CLIP model [8] maps images and text onto $\mathbb{R}^{512}$ such that semantically related image-text pairs have high cosine similarity. This new generation of embedding models reignited interest in vector databases optimized for ANN search to power applications ranging from reverse-image retrieval [10] to retrieval-augmented generative models [11]–[14].

This work provides an information-theoretic formulation to the problem of *private* ANN search. In this setting, a user with a query embedding q seeks to approximate the closest vector embedding to q within a database with M vectors, without revealing any information on q. This setting is the information-theoretic counterpart of the private ANN search problem studied in cryptography under computational security guarantees [15]–[18]. These protocols use computationally intensive tools such as oblivious RAM, garbled circuits, and homomorphic encryption to achieve privacy against computationally-bounded adversaries. In contrast, we aim to guarantee *perfect* information-theoretic privacy of a query q by considering *multiple* non-colluding databases. We propose a coding-theoretic construction for private ANN search that aims to reduce both communication and computational costs.

While our setting is closely related to private information retrieval (PIR) [19]–[27], a fundamental difference exists between the objectives of the two problems. In PIR, a user requires to download a given file from a system of databases that store multiple files, without revealing the index of the required file to any of the databases. Note that in PIR the user *knows the index* of the file to be downloaded prior to sending the queries. In contrast, in the problem of information-theoretically private ANN, a user aims to *find the index* of the nearest vector in the database to their query. Unlike PIR, here the index vector is unknown *a priori*. Once this index is privately obtained, the user can download the respective vector/underlying data content using classical PIR techniques. We foresee information-theoretic ANN as the first step of a two-step information-theoretically secure PIR protocol: First, a user finds the index of a file to be retrieved via private ANN search (e.g., the index of an image whose embedding is closest to that of a text query). Then, the user proceeds to privately download the file via a PIR protocol such as [20], [28], [29].

The proposed scheme is based on an r-level ANN search algorithm that divides the M vectors in the database into $M^{\frac{1}{r}}$ clusters in a hierarchical manner [30]–[33]. This results in a tree-structure of clusters. For a given query q, the algorithm traverses the branch that leads to the approximately closest vector to q. The coding theoretic approach ensures that no information on the branch traversed or the intermediate clusters investigated are revealed to any of the databases, which guarantees the privacy of q. The proposed scheme is able to achieve a communication cost of $O(N^2M^{\frac{1}{N-1}})$ with N non-colluding databases. The cryptographic protocols [15]–[18] that perform private ANN search achieve communication costs of $O(\sqrt{M}\log M)$, $O(M)$, $O(\log M)$ and $O(\log M)$, respectively, with computational privacy guarantees. Thus, our proposed scheme incurs a communication cost that is lower than the cryptographic protocols in [15], [16]. Moreover, the protocols in [17], [18] use fully-homomorphic encryption, which, to the best of our knowledge, is not practical over

This work is supported in part by the NSF awards CAREER-1845852, CIF-1900750, CIF-2231707, and CIF-2312667.

databases with thousands of entries (i.e., $M > 10^3$ [15]).

II. Problem Formulation

Notation. $[a:b]$ is the set of integers from a to $b \geq a$. x^T is the transpose of vector x and $\otimes$ denotes the Kronecker product.

We consider N non-colluding replicated vector databases consisting of M d-dimensional vectors denoted by v_i, $i \in [1:M]$. The entries of each v_i take values from a finite set specified by $[0:t-1]$ for some prime number t, i.e., $v_i \in [0:t-1]^d$ for $i \in [1:M]$. A user with a d-dimensional query vector $q \in [0:t-1]^d$ that is independent of all v_i, requires to retrieve the *closest* vector to q among all v_i, $i \in [1:M]$, without revealing any information on q to any of the N databases. The *closeness* between any two vectors is measured by the following similarity metric.

Definition 1 (Dot product similarity (DPS)) *Let a and b be two vectors such that $a, b \in [0:t-1]^d \subset \mathbb{F}_p^d$, where $\mathbb{F}_p$ is a large prime field with $p > (t-1)^2 d$. The DPS between a and b is defined as $\mathsf{S} : [0:t-1]^d \times [0:t-1]^d \to \mathbb{F}_p$,*

$$\mathsf{S}(a,b) = a^T b = \sum_{i=1}^{d} a_i b_i \pmod{p} = \sum_{i=1}^{d} a_i b_i. \tag{1}$$

For any three vectors $a, b, c \in [0:t-1]^d$, we say that vectors a and b are more similar compared to a and c if,[1] *$\mathsf{S}(a,b) > \mathsf{S}(a,c)$, where the comparison is performed considering the corresponding integers, i.e., $\mathsf{S}(a,b), \mathsf{S}(a,c) \in \mathbb{Z}_+$.*

In this problem setting, the user wishes to retrieve

$$i_{\text{DPS}} = \arg\max_{i \in [1:M]} \mathsf{S}(q, v_i), \tag{2}$$

without revealing any information on q to any of the databases.

To obtain the closest vector to a given query q in (2), the user sends a *privatized* query R_n to database n, $n \in [1:N]$, which responds with an answer A_n. The answer A_n is a function of R_n and the contents of the database, i.e.,

$$H(A_n | R_n, v_{[1:M]}) = 0, \quad n \in [1:N], \tag{3}$$

where $H(\cdot)$ denotes entropy. The user then approximates i_{DPS} using the answers received by all N databases as,

$$\hat{i}_{\text{DPS}} = f(R_{[1:N]}, A_{[1:N]}, q), \tag{4}$$

where $f(\cdot)$ is a deterministic function used to approximate i_{DPS}. The privacy constraint on the user's query q is given by,

$$I(q; R_n, v_{[1:M]}) = 0, \quad n \in [1:N], \tag{5}$$

where $I(\cdot)$ denotes mutual information. This ensures perfect information-theoretic privacy of q against non-colluding databases. We seek to design retrieval mechanisms that approximate (2) for a given q while satisfying (3)-(5) with the goal of minimizing the total communication cost, defined as,

$$C = C_D + C_U, \tag{6}$$

[1] In this formulation, we do not specify an exact finite field representation of the vectors that preserves the dot product similarity. An example case would be $t = 2$ with $p > d$, where the dot product between any two vectors reflects the similarity via a measure related to the Hamming distance.

Algorithm 1: r-level hierarchical ANN search

Data: r, q, $w_{i_1,\ldots,i_\ell}$, $\mathsf{C}_{i_1,\ldots,i_\ell}$, $\ell \in [1:r-1]$, for all $i_j \in [1:M^{\frac{1}{r}}]$, $j \in [1:\ell]$, and $v_{[1:M]}$
Result: Approximate of (2): $\hat{i}_{\text{DPS}}$
$\ell \leftarrow 2$;
$\hat{i}_1^* = \arg\max_{k \in \left[1:M^{\frac{1}{r}}\right]} \mathsf{S}(q, w_k)$;
while $\ell < r$ **do**
  $\hat{i}_\ell^* = \arg\max_{k \in \left[1:M^{\frac{1}{r}}\right]} \mathsf{S}(q, w_{\hat{i}_1^*,\ldots,\hat{i}_{\ell-1}^*,k})$;
  $\ell = \ell + 1$;
end
$\hat{i}_r^* = \arg\max_{i: v_i \in \mathsf{C}_{\hat{i}_1^*,\ldots,\hat{i}_{r-1}^*}} \mathsf{S}(q, v_i)$;
$\hat{i}_{\text{DPS}} \leftarrow \hat{i}_r^*$.

---

where C_D and C_U are the total numbers of $\mathbb{F}_p$ symbols downloaded and uploaded by the user, respectively. To this end, we fix a (non-private) ANN search algorithm and suitably modify it to incorporate privacy, while providing the same search accuracy as its non-private counterpart.

III. Main Result

The ANN search algorithm that we leverage is based on an r-level hierarchical clustering mechanism, as shown in Fig. 1. In level 0, all the M vectors belong to a single cluster. In level 1, the cluster in level 0 is partitioned into $M^{\frac{1}{r}}$ clusters denoted by[2] C_i, $i \in [1:M^{\frac{1}{r}}]$. In level 2, each cluster in level 1 is further divided into $M^{\frac{1}{r}}$ clusters. The clusters in level 2 are denoted by C_{i_1,i_2}, $i_1, i_2 \in [1:M^{\frac{1}{r}}]$, where i_1 and i_2 denote the cluster indices in levels 1 and 2 from which it was rooted. In general, a cluster in level $\ell \in [1:r-1]$ is denoted by $\mathsf{C}_{i_1,\ldots,i_\ell}$, where each i_j represents the index of its root cluster in level j. Each cluster $\mathsf{C}_{i_1,\ldots,i_\ell}$ in level $\ell \in [1:r-1]$ is assigned a corresponding representative vector $w_{i_1,\ldots,i_\ell} \in [0:t]^d$ (with the same subscript notation). An example of a cluster representative vector would be the average of all vectors within the cluster. Once a query is received, the r-level hierarchical ANN search protocol follows the steps shown in Algorithm 1 to approximately find the closest vector $v_i, i \in [1:M]$ to q.

With the above definitions, we now present the main result of this paper (the proof of which can be found in Section IV).

Theorem 1 *For a given query q, Algorithm 1 (r-level ANN search with M vectors) can be applied to approximate (2) while guaranteeing perfect privacy in (5) with a communication cost in (6) given by,*

$$C = \begin{cases} O\left(d\sqrt{M}\right), & \text{for } r = 2, \\ O\left(r^2 M^{\frac{1}{r}}\right), & \text{for } r > 2, \end{cases} \tag{7}$$

with $N \geq r+1$ if $r > 2$, and $N \geq r$ if $r = 2$.

[2] We assume: $M^{\frac{1}{r}} \in \mathbb{Z}_+$ and each cluster has the same number of vectors.

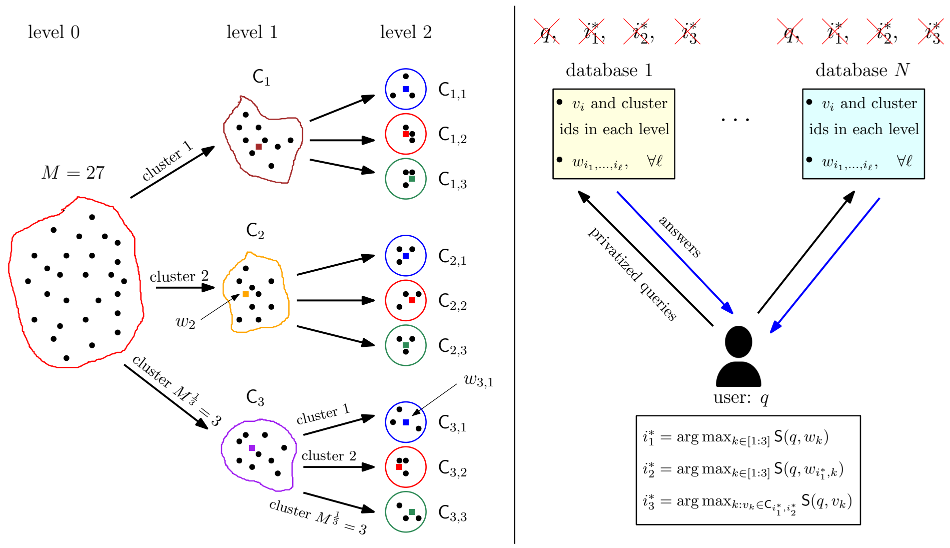

Fig. 1. An example setting with $M = 27$ vectors in each database for a 3-level hierarchical ANN search.

Corollary 1 *With N non-colluding databases, a user can perform $N-1$-level private ANN search with a communication cost of $O(N^2 M^{\frac{1}{N-1}})$ if $N > 2$, and 2-level private ANN search with a communication cost $O(d\sqrt{M})$ if $N = 2$.*[3]

Remark 1 *In Section IV we show that the computation complexity of the proposed scheme at each database over r rounds is $O(Md)$, i.e., independent of r. At the user's end, the number of computations decreases as r increases. This is because as r increases the tree of clusters gets narrower (since $M^{1/r}$ decreases) and deeper (since r increases). This reduces the number of dot products that the user needs to compute. Therefore, choosing a large value of r in the r-level ANN search decreases the overall number of computations and communications. However, to perform the r-level hierarchical ANN search with perfect privacy, the proposed scheme requires at least $r + 1$ non-colluding databases (except when $r=2$). Moreover, in practice, to maintain a certain level of accuracy, ANN is usually performed T times [34] with different cluster initializations, where $T \ll M^{\frac{1}{r}}$ in general. However, since we perform ANN for a total of r rounds, T increases exponentially with r as the clusters in level ℓ depend on the realization of clusters in level $\ell-1$. Therefore, r cannot be made arbitrarily large even with a sufficient number of databases.*

IV. Proposed Scheme

In this section, we prove Theorem 1. In particular, we first provide an example of the proposed scheme with $N = 4$, $M = 27$, and $r = 3$ (Fig. 1), followed by the general scheme.

A. Representative Example

In this example, the goal is to privately retrieve the closest vector to a given query q out of all the vectors v_i, $i \in [1 : 27]$, stored in each of the four databases. The cluster structure is fixed to have $M^{\frac{\ell}{r}} = 3^\ell$ clusters in each level $\ell \in [0 : 2]$, as shown in Fig. 1. To approximate (2), we follow the same steps as in Algorithm 1 with added steps to ensure the privacy constraint in (5). The scheme consists of $r = 3$ rounds. In rounds 1 and 2, the user obtains the clusters in levels 1 and 2, respectively, to which the query q is the closest. In round 3, the user finds the closest vector to q among the vectors in the selected cluster in the last level of the hierarchy using exhaustive search. We next describe these rounds in detail.

For each $n \in [1 : 4]$, we let $S_n^{[i]}$ denote the part of the content stored at the nth database that will be useful at round $i \in [1 : 3]$. These are given by,

$$S_n^{[1]} = \begin{bmatrix} w_1 & w_2 & w_3 \end{bmatrix}, \tag{8}$$

$$S_n^{[2]} = \begin{bmatrix} w_{1,1} & w_{1,2} & w_{1,3} \\ w_{2,1} & w_{2,2} & w_{2,3} \\ w_{3,1} & w_{3,2} & w_{3,3} \end{bmatrix}, \tag{9}$$

$$S_n^{[3]} = \begin{bmatrix} x_{1,1,1} & x_{1,1,2} & x_{1,1,3} \\ x_{1,2,1} & x_{1,2,2} & x_{1,2,3} \\ x_{1,3,1} & x_{1,3,2} & x_{1,3,3} \\ \vdots & \vdots & \vdots \\ x_{3,1,1} & x_{3,1,2} & x_{3,1,3} \\ x_{3,2,1} & x_{3,2,2} & x_{3,2,3} \\ x_{3,3,1} & x_{3,3,2} & x_{3,3,3} \end{bmatrix}, \tag{10}$$

where each w is the respective cluster representative vector and $x_{i,j,k}$ refers to the kth vector in the jth cluster in level 2 of the ith cluster in level 1, i.e., the triplet (i, j, k) in each $x_{i,j,k}$ corresponds to the cluster index in level 1, cluster index in level 2, and vector index in the cluster identified by $\mathsf{C}_{i,j}$, respectively. Each vector $w_i, w_{i,j}, x_{i,j,k}$ is of size $d \times 1$.

Round 1: In round 1, the user finds the closest cluster to q in level 1. For that, the user sends the following privatized query,

$$R_n^{[1]} = q + \alpha_n Z \tag{11}$$

to database n, $n \in [1 : 4]$ where $Z \sim \text{unif}(\mathbb{F}_p^d)$ is a random noise vector and α_n's are distinct constants from $\mathbb{F}_p$. Note

[3] The special case of $N = 2$ is described in Section IV-D.

that, by Shannon's one-time pad theorem [35], no information on q is revealed to the databases from each individual $R_n^{[1]}$. In round 1, answers from only two out of the four databases suffice to decode the closest cluster in level 1. The response from database n, $n \in [1:2]$ is given by,

$$A_n^{[1]}=S_n^{[1]T}R_n^{[1]}=\left[w_1^T q+\alpha_n w_1^T Z \dots w_3^T q+\alpha_n w_3^T Z\right]^T \quad (12)$$

from which the user obtains $w_i^T q$ by solving

$$\begin{bmatrix} 1 & \alpha_1 \\ 1 & \alpha_2 \end{bmatrix} \begin{bmatrix} w_i^T q \\ w_i^T Z \end{bmatrix} = \begin{bmatrix} A_{1,i}^{[1]} \\ A_{2,i}^{[1]} \end{bmatrix}, \quad i \in [1:3], \quad (13)$$

where $A_{n,i}^{[1]}$ is the ith entry of the answer vector from database $n \in [1:2]$. The user obtains the closest cluster to q in level 1 as $i_1^*=\arg\max_{i\in[1:3]} w_i^T q$. For this example, assume that $i_1^* = 2$.

Round 2: The goal of round 2 is to find the cluster index within C_2 that is the closest to q, without revealing any information on q or on the chosen cluster in level 1, i.e., C_2. To indicate the cluster chosen in level 1 that is investigated in level 2, the user sends the randomized query $R_n^{[2]} = [0\ 1\ 0]^T + \alpha_n \tilde{Z}$ to database n, $n \in [1:4]$, where $\tilde{Z} \sim \text{unif}(\mathbb{F}_p^3)$ is a random noise vector of size 3×1 independent of Z. Each database then combines the privatized queries from rounds 1 and 2 to obtain,

$$\tilde{R}_n^{[2]} = R_n^{[2]} \otimes R_n^{[1]} = \left([0\ 1\ 0]^T + \alpha_n \tilde{Z}\right) \otimes (q + \alpha_n Z) \quad (14)$$

$$= \left[0_d^T \quad q^T \quad 0_d^T\right]^T + \alpha_n \xi_1 + \alpha_n^2 \xi_2, \quad (15)$$

where 0_d is the all zeros vector of size $d \times 1$ and $\xi_j, j \in [1:2]$ (size $3d \times 1$) represents the coefficient of α_n^j in the polynomial in (15) that is common to all databases. In round 2, the scheme only requires answers from three out of the four databases. The response of database $n \in [1:3]$ is given by,

$$A_n^{[2]}=S_n^{[2]T}\tilde{R}_n^{[2]}=\begin{bmatrix} w_{1,1}^T & w_{2,1}^T & w_{3,1}^T \\ w_{1,2}^T & w_{2,2}^T & w_{3,2}^T \\ w_{1,3}^T & w_{2,3}^T & w_{3,3}^T \end{bmatrix}\left(\begin{bmatrix} 0_d \\ q \\ 0_d \end{bmatrix}+\sum_{i=1}^{2} \alpha_n^i \xi_i\right) \quad (16)$$

$$= \left[w_{2,1}^T q \quad w_{2,2}^T q \quad w_{2,3}^T q\right]^T + \alpha_n \tilde{\xi}_1 + \alpha_n^2 \tilde{\xi}_2, \quad (17)$$

where $\tilde{\xi}_i, i \in [1:2]$ (size 3×1) is the coefficient of α_n^i in the polynomial in (17) that is common to all databases. Then, the user obtains the dot products between q and the representative vectors of the sub clusters in C_2 by solving,

$$\begin{bmatrix} 1 & \alpha_1 & \alpha_1^2 \\ 1 & \alpha_2 & \alpha_2^2 \\ 1 & \alpha_3 & \alpha_3^2 \end{bmatrix} \begin{bmatrix} w_{2,i}^T q \\ \tilde{\xi}_{1,i} \\ \tilde{\xi}_{2,i} \end{bmatrix} = \begin{bmatrix} A_{1,i}^{[2]} \\ A_{2,i}^{[2]} \\ A_{3,i}^{[2]} \end{bmatrix}, \quad i \in [1:3], \quad (18)$$

where $A_{n,i}^{[2]}$ and $\tilde{\xi}_{k,i}$ are the ith elements of $A_n^{[2]}$ and $\tilde{\xi}_k$. The user obtains the closest cluster to q in level 2 as $i_2^* = \arg\max_{i\in[1:3]} w_{2,i}^T q$. For this example, we assume that $i_2^* = 1$.

Round 3: In this round, the user performs exhaustive search among the vectors in cluster $\mathsf{C}_{2,1}$, without revealing any information on q or the closest cluster indices found in rounds 1 and 2. Note that database $n \in [1:4]$ has already received the privatized queries on q and the chosen cluster index in level 1 via $R_n^{[1]}$ and $R_n^{[2]}$. To indicate the cluster index in level 2 on which exhaustive search is performed, the user sends $R_n^{[3]} = [1\ 0\ 0]^T + \alpha_n \hat{Z}$ to database n, $n \in [1:4]$, where $\hat{Z} \sim \text{unif}(\mathbb{F}_p^3)$ is a random noise vector of size 3×1 independent of Z and $\tilde{Z}$. Then, each database $n \in [1:4]$ combines the privatized queries from all the three rounds as,

$$\tilde{R}_n^{[3]} = R_n^{[2]} \otimes R_n^{[3]} \otimes R_n^{[1]} \quad (19)$$

$$= \left([0\ 1\ 0]^T+\alpha_n \tilde{Z}\right)\otimes\left([1\ 0\ 0]^T+\alpha_n \hat{Z}\right)\otimes(q+\alpha_n Z) \quad (20)$$

$$= [0_{3d}^T\ q^T\ 0_{5d}^T]^T+\alpha_n\eta_1+\alpha_n^2\eta_2+\alpha_n^3\eta_3, \quad (21)$$

where η_i is the coefficient of α_n^i in the polynomial in (21), that is common to all databases. The response of database n, $n \in [1:4]$ is given by,

$$A_n^{[3]} = S_n^{[3]T} \tilde{R}_n^{[3]} \quad (22)$$

$$= \begin{bmatrix} x_{1,1,1}^T & x_{1,2,1}^T & x_{1,3,1}^T & \dots & x_{3,1,1}^T & x_{3,2,1}^T & x_{3,3,1}^T \\ x_{1,1,2}^T & x_{1,2,2}^T & x_{1,3,2}^T & \dots & x_{3,1,2}^T & x_{3,2,2}^T & x_{3,3,2}^T \\ x_{1,1,3}^T & x_{1,2,3}^T & x_{1,3,3}^T & \dots & x_{3,1,3}^T & x_{3,2,3}^T & x_{3,3,3}^T \end{bmatrix}$$

$$\times \left([0_{3d}^T\ q^T\ 0_{5d}^T]^T + \alpha_n\eta_1 + \alpha_n^2\eta_2 + \alpha_n^3\eta_3\right) \quad (23)$$

$$= \left[x_{2,1,1}^T q \quad x_{2,1,2}^T q \quad x_{2,1,3}^T q\right]^T + \sum_{i=1}^{3} \alpha_n^i \tilde{\eta}_i, \quad (24)$$

where $\tilde{\eta}_i$ (size 3×1) is the coefficient of α_n^i in (24) that is common to all databases. Then, the user obtains the dot products between q and the vectors in $\mathsf{C}_{2,1}$ by solving,

$$\begin{bmatrix} 1 & \alpha_1 & \alpha_1^2 & \alpha_1^3 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & \alpha_4 & \alpha_4^2 & \alpha_4^3 \end{bmatrix} \begin{bmatrix} x_{2,1,i}^T q \\ \tilde{\eta}_{[1:3],i} \end{bmatrix} = \begin{bmatrix} A_{1,i}^{[3]} \\ \vdots \\ A_{4,i}^{[3]} \end{bmatrix}, \ i \in [1:3], \quad (25)$$

where $A_{n,i}^{[3]}$ and $\tilde{\eta}_{[1:3],i}$ represent the ith elements of $A_n^{[3]}$ and $\tilde{\eta}_{[1:3]} = [\tilde{\eta}_1, \tilde{\eta}_2, \tilde{\eta}_3]^T$, respectively. Finally, the user approximates the closest vector to q out of all the $M = 27$ vectors as $x_{i_1^*, i_2^*, i_3^*}$ where $i_3^* = \arg\max_{i\in[1:3]} x_{2,1,i}^T q$.

B. General Scheme

The proposed scheme that guarantees perfect privacy with $N \geq r+1$ databases consists of r rounds. The stored content relevant to each round in database n, $n \in [1:N]$ is given by,

$$S_n^{[1]} = \left[w_1 \quad \dots \quad w_{M^{\frac{1}{r}}}\right], \quad (26)$$

$$S_n^{[\ell]} = \begin{bmatrix} w_{\gamma_1,1} & w_{\gamma_1,2} & \dots & w_{\gamma_1,M^{\frac{1}{r}}} \\ w_{\gamma_2,1} & w_{\gamma_2,2} & \dots & w_{\gamma_2,M^{\frac{1}{r}}} \\ \vdots & \vdots & \vdots & \vdots \\ w_{\gamma_\lambda,1} & w_{\gamma_\lambda,2} & \dots & w_{\gamma_\lambda,M^{\frac{1}{r}}} \end{bmatrix}, \quad \ell \in [2:r], \quad (27)$$

where $\lambda = M^{\frac{\ell-1}{r}}$, with each w replaced by x for $\ell = r$ based on the notation in Section IV-A. Each γ_i in level ℓ refers to the subscripts of the root clusters in level $\ell - 1$. In particular, column i of $S_n^{[\ell]}$ contains the representative vector of the ith sub cluster of each of the clusters in level $\ell - 1$ in the exact order shown in Fig. 1.

Round 1: The user sends the privatized query $R_n^{[1]}$ in (11) to database n, $n \in [1:N]$. The user downloads the following answers from any two databases,

$$A_n^{[1]} = S_n^{[1]T} R_n^{[1]} = \{w_i^T q + \alpha_n w_i^T Z;\ i \in [1:M^{\frac{1}{r}}]\}. \quad (28)$$

As $\alpha_i \neq \alpha_j$, the user obtains $w_i^T q$, $i \in [1:M^{\frac{1}{r}}]$ and computes the closest cluster in level 1 as $i_1^* = \arg\max_{i\in[1:M^{\frac{1}{r}}]} w_i^T q$.

Round 2: The user finds the closest cluster to q in level 2 among those generated from $\mathsf{C}_{i_1^*}$. The user sends the following privatized query to database n, $n \in [1:N]$ to indicate that the search is narrowed down to cluster $\mathsf{C}_{i_1^*}$ in level 1,

$$R_n^{[2]} = \mathrm{e}_{M^{\frac{1}{r}}}(i_1^*) + \alpha_n Z_2, \quad (29)$$

where $\mathrm{e}_{M^{\frac{1}{r}}}(i_1^*)$ is the all zeros vector of size $M^{\frac{1}{r}} \times 1$ with a 1 in the i_1^*th position and Z_2 is a random noise vector from $\mathbb{F}_p^{M^{\frac{1}{r}}}$, independent of Z. The user downloads the answers from any three databases as,

$$A_n^{[2]} = S_n^{[2]T}\Big(R_n^{[2]} \otimes R_n^{[1]}\Big) = \Big[w_{i_1^*,1}^T q \quad \cdots \quad w_{i_1^*,M^{\frac{1}{r}}}^T q\Big]^T + \sum_{j=1}^{2} \alpha_n^i \tilde{\xi}_i, \quad (30)$$

where the notation is the same as the one used in Section IV-A. As (30) is a polynomial of α_n of degree 2, the user can obtain $w_{i_1^*,k}^T q$ for $k \in [1:M^{\frac{1}{r}}]$ using the answers from any three databases. The user then computes the closest cluster in level 2 as $i_2^* = \arg\max_{i\in[1:M^{\frac{1}{r}}]} w_{i_1^*,i}^T q$.

Round ℓ: The user requires to find the cluster i_ℓ^* (or vector i_ℓ^* when $\ell = r$) that is the closest to q among the clusters (or vectors if $\ell = r$) within $\mathsf{C}_{i_1^*,\dots,i_{\ell-1}^*}$. The user sends the following privatized query,

$$R_n^{[\ell]} = \mathrm{e}_{M^{\frac{1}{r}}}(i_{\ell-1}^*) + \alpha_n Z_\ell, \quad n \in [1:N], \quad (31)$$

where $Z_\ell \sim \mathrm{unif}(\mathbb{F}_p^{M^{\frac{1}{r}}})$ is independent of all the previous $Z_j, j \in [1:\ell-1]$. Database $n \in [1:N]$ responds as,

$$A_n^{[\ell]} = S_n^{[\ell]T}\left(R_n^{[2]} \otimes \dots \otimes R_n^{[\ell]} \otimes R_n^{[1]}\right) \quad (32)$$

$$= \Big[w_{i_1^*,\dots,i_{\ell-1}^*,1}^T q \dots w_{i_1^*,\dots,i_{\ell-1}^*,M^{\frac{1}{r}}}^T q\Big]^T + \sum_{i=1}^{\ell} \alpha_n^i \hat{\xi}_{i,\ell}, \quad (33)$$

where the notation is the same as the one used in Section IV-A with the exception of $\hat{\xi}_{i,\ell}$ that indicates the coefficient of α_n^i in the polynomial in (33) in the ℓth round (this coefficient is common to all databases). As (33) is a polynomial of α_n of degree ℓ, the user can obtain $w_{i_1^*,\dots,i_{\ell-1}^*,k}^T q$ for $k \in [1:M^{\frac{1}{r}}]$ using the answers from any $\ell+1$ databases. Note that $N \geq \ell+1$ must be satisfied for each $\ell \in [2:r]$ to solve (33), which imposes the constraint $N \geq r+1$ on the number of databases. The user then computes the closest cluster in level $\ell \in [2:r]$ as $i_\ell^* = \arg\max_{i\in[1:M^{\frac{1}{r}}]} w_{i_1^*,\dots,i_{\ell-1}^*,i}^T q$, (replace w by x for $\ell = r$). The vector index denoted by $(i_1^*,\dots,i_r^*)$, i.e., $x_{i_1^*,\dots,i_r^*}$ is the approximately closest vector to q in the databases.

Remark 2 *The main idea of the proposed scheme is to privately traverse the branch that leads to the approximately closest vector to q within the tree-structure of clusters. It is essentially PIR that is used in each level to hide the intermediate clusters investigated, to prevent the information leakage on q. For example, obtaining information on cluster C_2 in level 1 of Fig. 1 without revealing the index 2 is a PIR problem with 3 files. Once this information on C_2 is used to find the cluster index to be investigated in level 2 (e.g., $\mathsf{C}_{2,1}$) obtaining information on cluster $\mathsf{C}_{2,1}$ without revealing its index is another PIR problem with 9 files corresponding to the nine $\mathsf{C}_{i,j}$'s in level 2. Note that the number of files in these PIR formulations increases exponentially with the number of levels. As capacity-achieving PIR schemes have an optimal upload cost that scales with the number of files [28], using such approaches in this problem increases the upload cost up to $O(M)$ in level r. Therefore, we have proposed a coding theoretic approach that serves as a sub-optimal PIR scheme (order-wise optimal) with respect to the communication cost, which only requires the user to upload $O(M^{\frac{1}{r}})$ symbols, resulting in a total communication cost that scales with $M^{\frac{1}{r}}$.*

C. Privacy and Communication Cost

Privacy: All the information sent by the user to each database $n \in [1:N]$ is of the form $R_n^{[\ell]}$, $\ell \in [1:r]$. Note that the private information (query and cluster indices) in each $R_n^{[\ell]}$ is one-time padded with the noise vectors Z_ℓ that are randomly selected from $\mathbb{F}_p^{M^{\frac{1}{r}}}$. This makes q and the cluster indices independent of all the $R_n^{[\ell]}$'s sent to each database, which guarantees (5).

Communication cost: The cost C_U of the proposed scheme (total number of uploads in $R_n^{[\ell]}$, $\forall \ell, n$, by noting that the user needs to query only at most $r+1$ databases) is given by $C_U = O(r^2 M^{\frac{1}{r}})$ since, in practice, d is much smaller than M in vector databases [8]. The download cost is $C_D = O(r^2 M^{\frac{1}{r}})$ since $M^{\frac{1}{r}}$ single-symbol dot products are downloaded from at most $r+1$ databases in each round. Therefore, with reference to (6), we have that $C = O(r^2 M^{\frac{1}{r}})$.

Remark 3 *The proposed scheme can be directly extended to general r-level ANN structures with K_i clusters in each branch of each level for $i \in [1:r-1]$. In particular, each of the K_i clusters contains $M/\left(\prod_{j=1}^{i} K_j\right)$ vectors. The resulting communication cost is $O\left(r\sum_{i=1}^{r-1} K_i + Mr/\left(\prod_{j=1}^{r-1} K_j\right)\right)$.*

D. The Special Case $N = 2$

For the case $N = 2$, round 1 is identical to what is described above. In round 2, the user sends the privatized query $R_n^{[2]} = (\mathrm{e}_{M^{\frac{1}{r}}}(i_1^*) \otimes q) + \alpha_n \bar{Z}$ to database $n \in [1:2]$, where $\bar{Z} \in \mathbb{F}_p^{M^{\frac{1}{r}} d}$ is a random noise vector independent of Z. Each database answers with $A_n^{[2]} = S_n^{[2]T} R_n^{[2]}$, which is the same as (30) with a polynomial of degree 1. This lets the user decode the dot products with only two answers. The upload cost of this case is $O(2M^{\frac{1}{r}} d)$. As this modification can only be done in the first two rounds (the order of the Kronecker products matters after round 2) the only value of r that allows this is $r = 2$, which makes the upload cost of this case $O(\sqrt{M} d)$.

REFERENCES

[1] P. Indyk and R. Motwani. Approximate Nearest Neighbors: Towards Removing the Curse of Dimensionality. In *ACM Symposium on Theory of Computing*, page 604–613, 1998.

[2] W. Li, Y. Zhang, Y. Sun, W. Wang, M. Li, W. Zhang, and X. Lin. Approximate Nearest Neighbor Search on High Dimensional Data — Experiments, Analyses, and Improvement. *IEEE Transactions on Knowledge and Data Engineering*, 32(8):1475–1488, 2020.

[3] R. Chen, B. Liu, H. Zhu, Y. Wang, Q. Li, B. Ma, Q. Hua, J. Jiang, Xu Y, H. Deng, and B. Zheng. Approximate Nearest Neighbor Search under Neural Similarity Metric for Large-Scale Recommendation. In *ACM International Conference on Information and Knowledge Management*, 2022.

[4] M. Badr, D. Vodislav, D. Picard, S. Yin, and P.-H. Gosselin. Multi-criteria search algorithm: An efficient approximate k-NN algorithm for image retrieval. In *2013 IEEE International Conference on Image Processing*, 2013.

[5] X. Gu, L. Akoglu, and A. Rinaldo. Statistical Analysis of Nearest Neighbor Methods for Anomaly Detection. In *Conference on Neural Information Processing Systems*, 2019.

[6] P. Anagnostou, P. Barbas, A. G. Vrahatis, and S. K. Tasoulis. Approximate kNN Classification for Biomedical Data. In *2020 IEEE International Conference on Big Data*, pages 3602–3607, 2020.

[7] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser, and I. Polosukhin. Attention is All you Need. In *Conference on Neural Information Processing Systems*, 2017.

[8] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever. Learning Transferable Visual Models From Natural Language Supervision. In *International Conference on Machine Learning*, 2021.

[9] J. Devlin, M. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *North American Chapter of the Association for Computational Linguistics*, 2019.

[10] C. Schuhmann, R. Beaumont, R. Vencu, C. Gordon, R. Wightman, M. Cherti, T. Coombes, A. Katta, C. Mullis, M. Wortsman, P. Schramowski, S. Kundurthy, K. Crowson, L. Schmidt, R. Kaczmarczyk, and J. Jitsev. LAION-5b: An open large-scale dataset for training next generation image-text models. In *Conference on Neural Information Processing Systems*, 2022.

[11] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W. Yih, T. Rocktäschel, S. Riedel, and D. Kiela. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. In *Conference on Neural Information Processing Systems*, 2020.

[12] M. Bleeker, P. Swietojanski, S. Braun, and X. Zhuang. Approximate Nearest Neighbour Phrase Mining for Contextual Speech Recognition. In *Interspeech*, 2023.

[13] M. Karppa, M. Aumüller, and R. Pagh. DEANN: Speeding up Kernel-Density Estimation using Approximate Nearest Neighbor Search. In *International Conference on Artificial Intelligence and Statistics*, pages 3108–3137, 2022.

[14] J. Johnson, M. Douze, and H. Jégou. Billion-Scale Similarity Search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547, 2021.

[15] S. Servan-Schreiber, S. Langowski, and S. Devadas. Private Approximate Nearest Neighbor Search with Sublinear Communication. In *IEEE Symposium on Security and Privacy*, 2022.

[16] H. Chen, I. Chillotti, Y. Dong, O. Poburinnaya, I. Razenshteyn, and M S. Riazi. SANNS: Scaling up secure approximate k-nearest neighbors search. In *29th USENIX Conference on Security Symposium*, 2020.

[17] H. Shaul, D. Feldman, and D. Rus. Secure k-ish Nearest Neighbors Classifier. In *Privacy Enhancing Technologies*, 2020.

[18] M. Zuber and R. Sirdey. Efficient homomorphic evaluation of k-NN classifiers. In *Privacy Enhancing Technologies*, 2021.

[19] B. Chor, E. Kushilevitz, O. Goldreich, and M. Sudan. Private Information Retrieval. *Journal of the ACM*, 45(6):965–981, 1998.

[20] H. Sun and S. A. Jafar. The Capacity of Private Information Retrieval. *IEEE Transactions on Information Theory*, 63(7):4075–4088, 2017.

[21] K. Banawan and S. Ulukus. The Capacity of Private Information Retrieval From Coded Databases. *IEEE Transactions on Information Theory*, 64(3):1945–1956, 2018.

[22] H. Sun and S. A. Jafar. The Capacity of Robust Private Information Retrieval With Colluding Databases. *IEEE Transactions on Information Theory*, 64(4):2361–2370, 2018.

[23] H. Sun and S. A. Jafar. The Capacity of Symmetric Private Information Retrieval. *IEEE Transactions on Information Theory*, 65(1):329–322, 2019.

[24] K. Banawan and S. Ulukus. Multi-Message Private Information Retrieval: Capacity Results and Near-Optimal Schemes. *IEEE Transactions on Information Theory*, 64(10):6842–6862, 2018.

[25] S. Kadhe, B. Garcia, A. Heidarzadeh, S. El Rouayheb, and A. Sprintson. Private Information Retrieval with Side Information. *IEEE Transactions on Information Theory*, 66(4):2032–2043, 2020.

[26] Z. Chen, Z. Wang, and S.A. Jafar. The Asymptotic Capacity of Private Search. *IEEE Transactions on Information Theory*, 66(8):4709–4721, 2020.

[27] C. Chor, N. Gilboa, and M. Naor. Private Information Retrieval by Keywords. *Cryptology ePrint Archive*, 1998.

[28] C. Tian, H. Sun, and J. Chen. Capacity-Achieving Private Information Retrieval Codes with Optimal Message Size and Upload Cost. *IEEE Transactions on Information Theory*, 65(11):7613–7627, 2019.

[29] I. Samy, R. Tandon, and L. Lazos. On the Capacity of Leaky Private Information Retrieval. In *IEEE International Symposium on Information Theory*, 2019.

[30] J. Vargas Mu noz, M. A. Gonçalves, Z. Dias, and R. da S. Torres. Hierarchical Clustering-Based Graphs for Large Scale Approximate Nearest Neighbor Search. *Pattern Recognition*, 96(C), 2019.

[31] S. Dasgupta and K. Sinha. Randomized Partition Trees for Exact Nearest Neighbor Search. *Algorithmica*, 72(1):237–263, 2015.

[32] T. Liu, A. Moore, A. Gray, and K. Yang. An Investigation of Practical Approximate Nearest Neighbor Algorithms. In *Conference on Neural Information Processing Systems*, 2004.

[33] A. Fahim, M.E. Ali, and M. A. Cheema. Unsupervised Space Partitioning for Nearest Neighbor Search. In *26th International Conference on Extending Database Technology*, 2023.

[34] E. Bernhardsson. Annoy at github. https://github.com/spotify/annoy, 2015.

[35] C. E. Shannon. Communication Theory of Secrecy Systems. *Bell System Technical Journal*, 28(4):656–715, 1949.