# Privacy-Aware Similarity Search

## A Glance at Approximate $k$-Nearest Neighbors Search

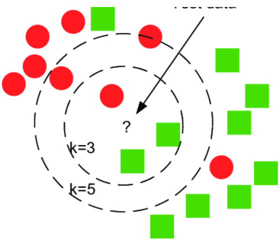
MENG Xiangyi

Computer Science Department
City University of Hong Kong

November 1, 2021

**Nearest neighbor (NN), $k$-NN, Approximate $k$-NN**

Shared by Jing last week...



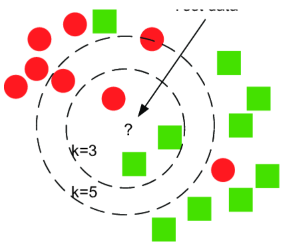
# Naive similarity search: Enough or not?

## Nearest neighbor (NN), $k$-NN, Approximate $k$-NN

Shared by Jing last week...

## Applications

- Recommendation System, Large-Scale Machine Learning
- Face Recognition, Biometric identification
- General-Purpose Similarity Search



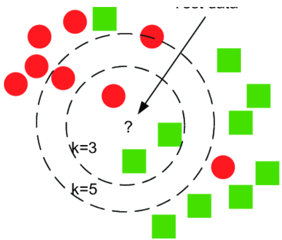
# Naive similarity search: Enough or not?

## Nearest neighbor (NN), *k*-NN, Approximate *k*-NN

Shared by Jing last week...

## Applications

- Recommendation System, Large-Scale Machine Learning
- Face Recognition, Biometric identification
- General-Purpose Similarity Search




## Sensitive Data

Disclosure of data is not always acceptable.

- Biometrics: Face, Genetic Sequence, Clinical Data; User Profiles...
- Laws and regulations (GDPR in EU, Data Security Law in China...)

# Privacy-Aware (Approximate) $k$-NN: Scenarios

## Data Owner

- Client: Outsource encrypted data to remote DB server (Searchable Encryption)
- **Server Provider (SP)**: The provider itself holds the data (stored in plaintext)

# Privacy-Aware (Approximate) $k$-NN: Scenarios

## Data Owner

- Client: Outsource encrypted data to remote DB server (Searchable Encryption)
- **Server Provider (SP)**: The provider itself holds the data (stored in plaintext)

## Number of Server

- **Single**: Most of works follow this trend
- **Dual**: Good for lightweight protocol
- Multi-Party: Each party has a part of the data (horizontal/vertical)

# Privacy-Aware (Approximate) $k$-NN: Scenarios

## Data Owner

- Client: Outsource encrypted data to remote DB server (Searchable Encryption)
- **Server Provider (SP)**: The provider itself holds the data (stored in plaintext)

## Number of Server

- **Single**: Most of works follow this trend
- **Dual**: Good for lightweight protocol
- Multi-Party: Each party has a part of the data (horizontal/vertical)

## Threat Models

- Semi-Honest (SH): Parties follow the protocol specification but try to infer information from what they received
- Malicious Adversaries (Client/Server) (MA/MC/MS): Parties may deviate from protocol, collude with others, or otherwise behave maliciously to learn more about the database.
- Mixed: Some parties are SH and some parties are MA.

# Privacy-Aware (Approximate) $k$-NN: State of the Art

| | TM/ #Srv/ DO | NN approach | Comm. | Comp. | Rounds | Crypt. Tools | Efficiency |
|---|---|---|---|---|---|---|---|
| [SFR20] | SH/ Single/ SP | $k$-ish NN | $\log N$ | $N$ | 1 | FHE | ★☆☆☆ |
| SANNS [Che+20] | SH/ Single/ SP | $k$-Means | $N/k$ | $N$ | $N$ | AHE/ GC/ ORAM | ★★☆☆ |
| [ZS21] | SH/ Single/ Both | Linear Scan | $\log N$ | $N^2$ | 1 | FHE | ★★☆☆ |
| PP-AkNN [BT21] | SH/ Single/ Client | extended LSH | $\log^2 N$ | N/A | $N$ | ODS | N/A |
| [SLD21] | MC/ Dual/ SP | LSH | $\sqrt{N}$ | $N$ | 1 | DPF | ★★★★ |

Table: Comparison between 5 recently proposed approaches. TM stands for *Threat Model*. DO stands for *Data Owner*. $N$ is the database size.

# To be discussed in this talk

- SANNS: Scaling Up Secure Approximate $k$-Nearest Neighbors Search (USENIX Security 20')[Che+20]
- What else can we do to exceed them (in some ways)?

# SANNS: Scaling Up Secure Approximate k-Nearest Neighbors Search

Heavy cryptographic tools-based solution

# Previous works

## Linear scan: Secure $k$-NN in a 2PC setting

- Compute the distance between q and all the points in $X$
- Select the top-$k$ elements

# Previous works

## Linear scan: Secure $k$-NN in a 2PC setting

- Compute the distance between q and all the points in $X$
- Select the top-$k$ elements

## Distance Computation

Some of the previous works use

- Paillier[Pai99] (in [Bar+10; ESJ14; Erk+09; Hua+11]) (no SIMD acceleration)
- BFV[Bra12; FV12] (in [JVC18]) (used by SANNS)
- Oblivious Transfer (OT)-based multiplication (in [DSZ15]) (less computation but more communication)

# Previous works

## Linear scan: Secure $k$-NN in a 2PC setting

- Compute the distance between q and all the points in $X$
- Select the top-$k$ elements

## Distance Computation

Some of the previous works use

- Paillier[Pai99] (in [Bar+10; ESJ14; Erk+09; Hua+11]) (no SIMD acceleration)
- BFV[Bra12; FV12] (in [JVC18]) (used by SANNS)
- Oblivious Transfer (OT)-based multiplication (in [DSZ15]) (less computation but more communication)

## Top-$k$ selection

SANNS achieves this task by garbling a new top-k circuit ($O(n + k^2)$ comparators), some of the previous works use:

- The naive circuit of size $O(nk)$ (in [Ash+18; SGB18; Son+15])
- Homomorphic encryption such as BGV[BGV14] (in [SFR18; SFR20])

# Contributions

## SANNS

**S**ecure **A**pproximate $k$-**N**earest **N**eighbor **S**earch

- The server (data owner): learns nothing
- Client (query maker): learns nothing but the final result
- Query (query point $q$ and the result) and database are kept confidential

# Contributions

## SANNS

**S**ecure **A**pproximate $k$-**N**earest **N**eighbor **S**earch

- The server (data owner): learns nothing
- Client (query maker): learns nothing but the final result
- Query (query point $q$ and the result) and database are kept confidential

## Two protocols

Based on **approximate top-$k$ selection** (semi-honest model)

- Optimized linear scan
- Sublinear-time clustering-based algorithm

# Contributions

## SANNS

**S**ecure **A**pproximate $k$-**N**earest **N**eighbor **S**earch

- The server (data owner): learns nothing
- Client (query maker): learns nothing but the final result
- Query (query point $q$ and the result) and database are kept confidential

## Two protocols

Based on **approximate top-$k$ selection** (semi-honest model)

- Optimized linear scan
- Sublinear-time clustering-based algorithm

## Scalability

SANNS scales up to databases with 10 million entries

- Tests on various datasets: SIFT, Deep1B, Amazon reviews text.

$k = 10$ over $10M$ entries with accuracy 0.9 in a bunch of seconds

## Secret Sharing

- Arithmetic: $x \in \mathbb{Z}_t \longmapsto (x_A, x_B)$ such that $x_A + x_B \equiv x \pmod{t}$
- Boolean: $x \in \{0, 1\}^\tau \longmapsto (x_A, x_B)$ such that $x_A \oplus x_B \equiv x$

# Preliminaries

## Secret Sharing

- Arithmetic: $x \in \mathbb{Z}_t \longmapsto (x_A, x_B)$ such that $x_A + x_B \equiv x \pmod{t}$
- Boolean: $x \in \{0,1\}^\tau \longmapsto (x_A, x_B)$ such that $x_A \oplus x_B \equiv x$

## AHE

A (private-key) additive homomorphic encryption

- Operations: add two ciphertexts, add/multiply a ciphertext by a constant
- SANNS uses the **BFV**[Bra12; FV12] implementation in SEAL[a]

[a] https://github.com/Microsoft/SEAL

# Preliminaries

## Secret Sharing

- Arithmetic: $x \in \mathbb{Z}_t \longmapsto (x_A, x_B)$ such that $x_A + x_B \equiv x \pmod{t}$
- Boolean: $x \in \{0,1\}^\tau \longmapsto (x_A, x_B)$ such that $x_A \oplus x_B \equiv x$

## AHE

A (private-key) additive homomorphic encryption

- Operations: add two ciphertexts, add/multiply a ciphertext by a constant
- SANNS uses the **BFV**[Bra12; FV12] implementation in SEAL[a]

[a] https://github.com/Microsoft/SEAL

## GC

Garbled circuit[Yao86]: achieves generic secure two-party computation for arbitrary Boolean circuits.
(A method that enables two parties with private inputs $x$ and $y$ to jointly compute a function $f(x, y)$, useful especially in comparison)

## DORAM

Distributed version of oblivious RAM

- 2 parties hold secret shares of an array
- oblivious read and write operations are supported
- SANNS uses Floram[DS17], which leverages Functional Secret Sharing to mask the database

## DORAM

Distributed version of oblivious RAM

- 2 parties hold secret shares of an array
- oblivious read and write operations are supported
- SANNS uses Floram[DS17], which leverages Functional Secret Sharing to mask the database

## $k$-means clustering

Iterative algorithm[Llo82]

- Find a clustering $X = C_1 \cup C_2 \cup ... \cup C_k$
- Find centers $\mathbf{c_1}, \mathbf{c_2}, ..., \mathbf{c_k} \in \mathbb{R}^d$ which approximately minimize

$$\sum_{i=1}^{k} \sum_{\mathbf{x} \in C_i} ||\mathbf{c_i} - \mathbf{x}||^2$$

# SANNS: optimized linear scan

## First algorithm: optimized linear scan

- Compute the distance between $q$ and all the points in $X \rightarrow$ AHE
- **Approximately select the** $k$ **closest elements to** $q \rightarrow$ GC

## First algorithm: optimized linear scan

- Compute the distance between $q$ and all the points in $X \rightarrow$ AHE
- **Approximately select the $k$ closest elements to $q$** $\rightarrow$ GC

## Approximate top-$k$ selection

Being used in both optimized linear scan and clustering based algorithm

- Given a list of $n$ numbers ($b$-bits), output the $k \leq n$ smallest elements in the sorted order
- Augmented functionality: output the ID together with the $k$ smallest values

# Approximate top-$k$

## Approach: the output has to be **approximately correct** (query point $q$)

1. Shuffle inputs in uniformly random order

$$x_1, .., x_n \longmapsto x_{\pi(1)}, ..., x_{\pi(n)}$$

# Approximate top-$k$

**Approach: the output has to be approximately correct (query point $q$)**

1. Shuffle inputs in uniformly random order

$$x_1, .., x_n \longmapsto x_{\pi(1)}, ..., x_{\pi(n)}$$

2. Partition inputs into $l \leq n$ bins (of size $n/l$)

$$U_1 = \{x_{\pi(1)}, ..., x_{\pi(n/l)}\}, ..., U_l = \{x_{\pi((l-1)n/l)}, ..., x_{\pi(n)}\}$$

## Approach: the output has to be **approximately correct** (query point $q$)

1. Shuffle inputs in uniformly random order

$$x_1, .., x_n \longmapsto x_{\pi(1)}, ..., x_{\pi(n)}$$

2. Partition inputs into $l \leq n$ bins (of size $n/l$)

$$U_1 = \{x_{\pi(1)}, ..., x_{\pi(n/l)}\}, ..., U_l = \{x_{\pi((l-1)n/l)}, ..., x_{\pi(n)}\}$$

3. Compute the minimum within each bin

$$M_i = \min_{x \in U_i} dist(x, q)$$

# Approximate top-$k$

**Approach: the output has to be approximately correct (query point $q$)**

1. Shuffle inputs in uniformly random order

$$x_1, .., x_n \longmapsto x_{\pi(1)}, ..., x_{\pi(n)}$$

2. Partition inputs into $l \leq n$ bins (of size $n/l$)

$$U_1 = \{x_{\pi(1)}, ..., x_{\pi(n/l)}\}, ..., U_l = \{x_{\pi((l-1)n/l)}, ..., x_{\pi(n)}\}$$

3. Compute the minimum within each bin

$$M_i = \min_{x \in U_i} dist(x, q)$$

4. Compute (naive algorithm) the $k$-min between the $M_i$

**Approach: the output has to be approximately correct (query point $q$)**

1. Shuffle inputs in uniformly random order

$$x_1, .., x_n \longmapsto x_{\pi(1)}, ..., x_{\pi(n)}$$

2. Partition inputs into $l \leq n$ bins (of size $n/l$)

$$U_1 = \{x_{\pi(1)}, ..., x_{\pi(n/l)}\}, ..., U_l = \{x_{\pi((l-1)n/l)}, ..., x_{\pi(n)}\}$$

3. Compute the minimum within each bin

$$M_i = \min_{x \in U_i} dist(x, q)$$

4. Compute (naive algorithm) the $k$-min between the $M_i$

Circuit size: $O(bnk) \to O(b \cdot (n + kl))$

# Approximate top-$k$

## Approach: the output has to be **approximately correct** (query point $q$)

1. Shuffle inputs in uniformly random order

$$x_1, .., x_n \longmapsto x_{\pi(1)}, ..., x_{\pi(n)}$$

2. Partition inputs into $l \leq n$ bins (of size $n/l$)

$$U_1 = \{x_{\pi(1)}, ..., x_{\pi(n/l)}\}, ..., U_l = \{x_{\pi((l-1)n/l)}, ..., x_{\pi(n)}\}$$

3. Compute the minimum within each bin

$$M_i = \min_{x \in U_i} dist(x, q)$$

4. Compute (naive algorithm) the $k$-min between the $M_i$

Circuit size: $O(bnk) \rightarrow O(b \cdot (n + kl))$

## Approximate distances

Discard the $r$ low-order bits of the $b$-bit inputs

- Circuit size: $O(b \cdot (n + kl)) \rightarrow O((b - r)) \cdot (n + kl))$

# SANNS: clustering based algorithm

## Second algorithm: clustering based

- Sublinear time: avoid computing all the distances

# SANNS: clustering based algorithm

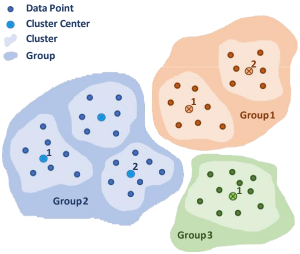
## Second algorithm: clustering based

- Sublinear time: avoid computing all the distances
- Clustering approach is not the best in plaintext $k$-NNS[ABF20], but suitable for secure computation

# SANNS: clustering based algorithm

## Second algorithm: clustering based

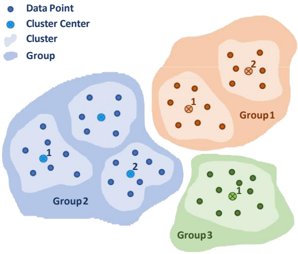
- Sublinear time: avoid computing all the distances
- Clustering approach is not the best in plaintext $k$-NNS[ABF20], but suitable for secure computation
- Cluster rebalancing: cluster of the same size via padding



# SANNS: clustering based algorithm

## Second algorithm: clustering based

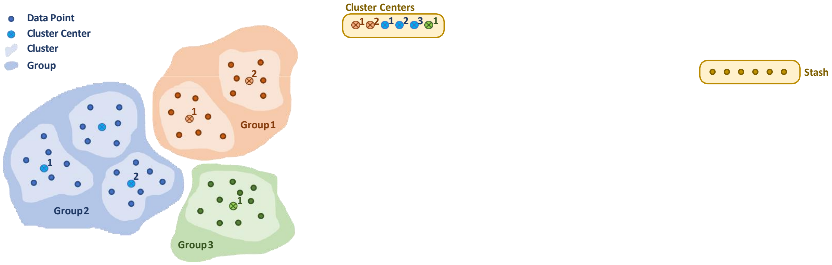
- Sublinear time: avoid computing all the distances
- Clustering approach is not the best in plaintext $k$-NNS[ABF20], but suitable for secure computation
- Cluster rebalancing: cluster of the same size via padding
- **Stash**: linearly scanned (reduce number of accesses)

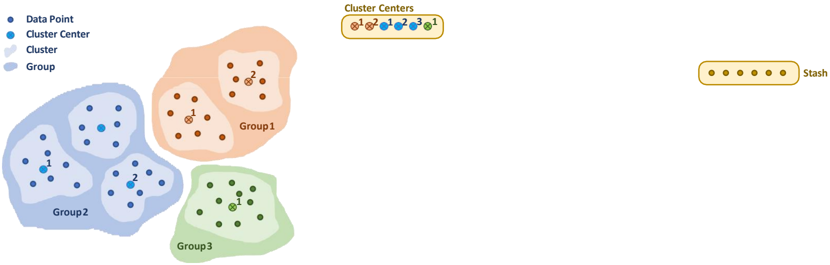



## SANNS: clustering based algorithm

### Second algorithm: clustering based

- Sublinear time: avoid computing all the distances
- Clustering approach is not the best in plaintext $k$-NNS[ABF20], but suitable for secure computation
- Cluster rebalancing: cluster of the same size via padding
- **Stash**: linearly scanned (reduce number of accesses)



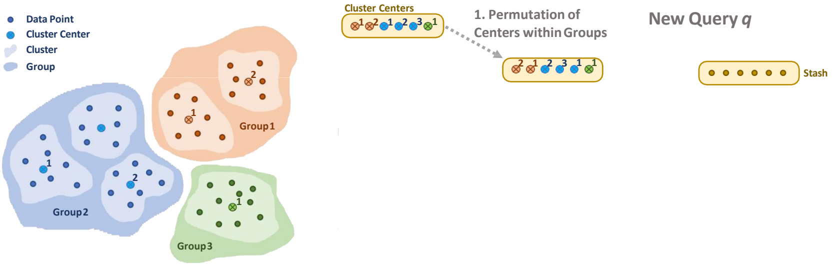



**Dataset:** Floram(init)

**Clusters**

**Stash**

# SANNS: clustering based algorithm

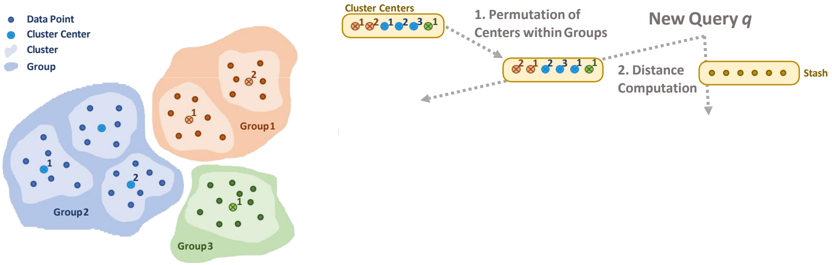



**Dataset:** Floram(init)

**Clusters**

**Stash**

# SANNS: clustering based algorithm




**Dataset:** Floram(init)

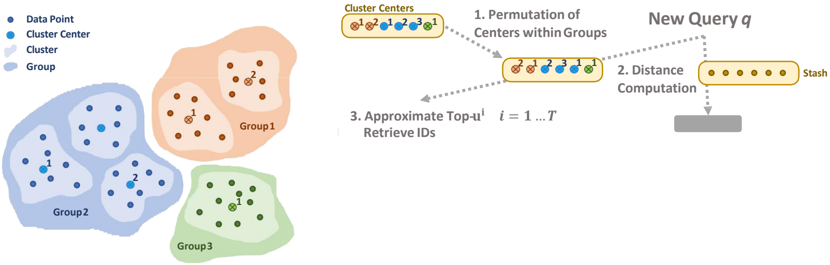
**Clusters**

- **Step 2:** AHE

**Stash**

- **Step 2:** AHE

# SANNS: clustering based algorithm




**Dataset:** Floram(init)

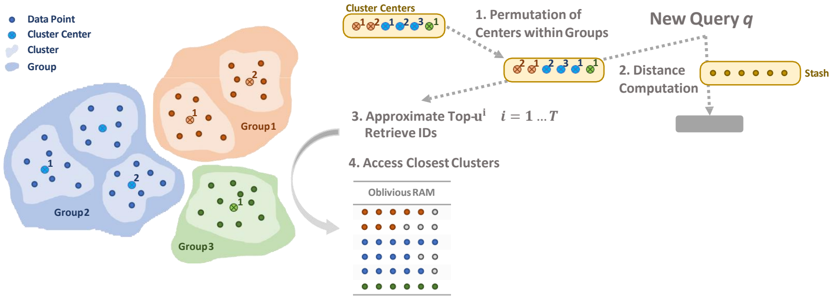
**Clusters**

- **Step 2:** AHE
- **Step 3:** GC

**Stash**

- **Step 2:** AHE

# SANNS: clustering based algorithm




**Dataset:** Floram(init)

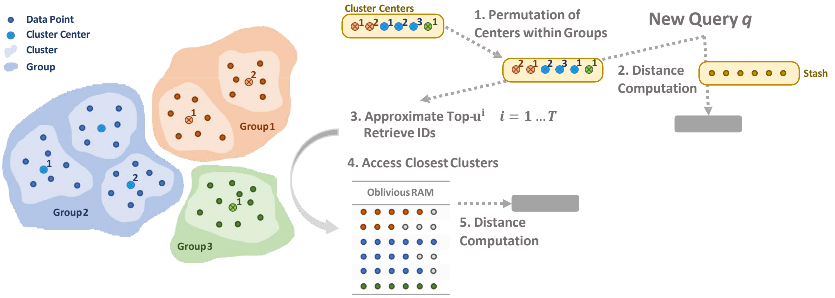
**Clusters**

- **Step 2:** AHE
- **Step 3:** GC
- **Step 4:** Floram(read)

**Stash**

- **Step 2:** AHE

# SANNS: clustering based algorithm




**Dataset:** Floram(init)

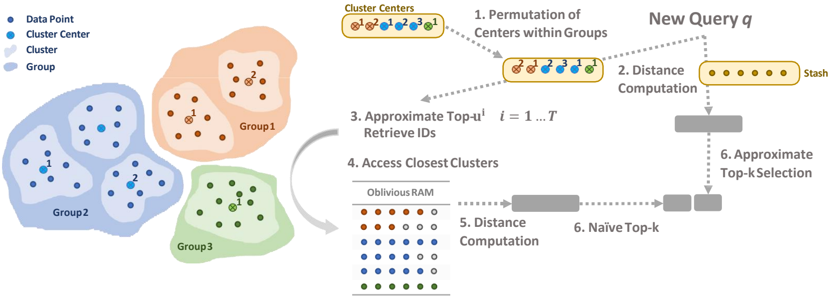
**Clusters**

- **Step 2:** AHE
- **Step 3:** GC
- **Step 4:** Floram(read)
- **Step 5:** AHE

**Stash**

- **Step 2:** AHE

# SANNS: clustering based algorithm




**Dataset:** Floram(init)

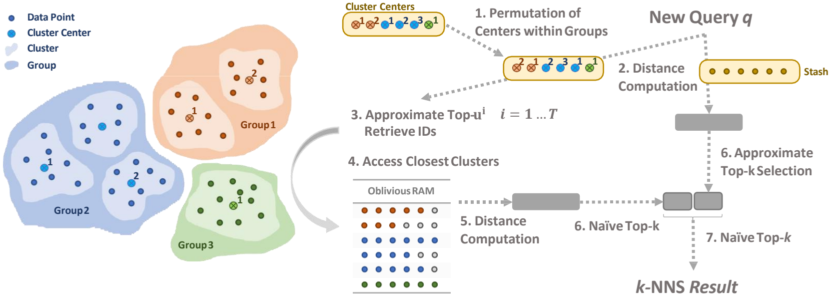
**Clusters**

- **Step 2:** AHE
- **Step 3:** GC
- **Step 4:** Floram(read)
- **Step 5:** AHE
- **Step 6:** GC

**Stash**

- **Step 2:** AHE
- **Step 6:** GC

# SANNS: clustering based algorithm




**Dataset:** Floram(init)

**Clusters**

- **Step 2:** AHE
- **Step 3:** GC
- **Step 4:** Floram(read)
- **Step 5:** AHE
- **Step 6:** GC

**Step 7 - final top-$k$:** GC

**Stash**

- **Step 2:** AHE
- **Step 6:** GC

# Experiments

## Datasets

SIFT[Low99]

- Standard dataset of image descriptors (similarity between images)
- $1M$ entries of size $d = 128$
- 8-bits

Deep1B[BL16]

- Dataset of image descriptors (feature vectors extracted from a DNN)
- $1B$ entries of size $d = 96$
- Quantized to 8-bits
- Deep1B-1M: first $1M$ images, Deep1B-10M: first $10M$ imagess

Amazon[McA+15]

- Dataset of reviews on Amazon
- $1M$ entries of size $d = 50$
- Quantized to 9-bits

# Possible improvements?

- Computing distance (especially euclidean distance) using HE is really expensive.
  - Differential private Euclidean Distance Approximation[Sta21]
  - LSH-based solution avoiding heavy cryptographic tools(An e-print[SLD21])
- How to leverage state-of-the-art plaintext approach?
- How to excatly and accurately quantize the information leakage?
- Other distance metrics? (Hamming, Cosine similarity...)
- Distributed System for extremely large database?

[ABF20] Martin Aumüller, Erik Bernhardsson, and Alexander John Faithfull. "ANN-Benchmarks: A benchmarking tool for approximate nearest neighbor algorithms". In: *Inf. Syst.* 87 (2020). DOI: 10.1016/j.is.2019.02.006. URL: https://doi.org/10.1016/j.is.2019.02.006.

[Ash+18] Gilad Asharov et al. "Privacy-Preserving Search of Similar Patients in Genomic Data". In: *Proc. Priv. Enhancing Technol.* 2018.4 (2018), pp. 104–124. DOI: 10.1515/popets-2018-0034. URL: https://doi.org/10.1515/popets-2018-0034.

[Bar+10] Mauro Barni et al. "Privacy-preserving fingercode authentication". In: *Multimedia and Security Workshop, MM&Sec 2010, Roma, Italy, September 9-10, 2010*. Ed. by Patrizio Campisi, Jana Dittmann, and Scott Craver. ACM, 2010, pp. 231–240. DOI: 10.1145/1854229.1854270. URL: https://doi.org/10.1145/1854229.1854270.

[BGV14] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. "(Leveled) Fully Homomorphic Encryption without Bootstrapping". In: *ACM Trans. Comput. Theory* 6.3 (2014), 13:1–13:36. DOI: 10.1145/2633600. URL: https://doi.org/10.1145/2633600.

[BL16] Artem Babenko and Victor S. Lempitsky. "Efficient Indexing of Billion-Scale Datasets of Deep Descriptors". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*. IEEE Computer Society, 2016, pp. 2055–2063. DOI: 10.1109/CVPR.2016.226. URL: https://doi.org/10.1109/CVPR.2016.226.

[Bra12] Zvika Brakerski. "Fully Homomorphic Encryption without Modulus Switching from Classical GapSVP". In: *Advances in Cryptology - CRYPTO 2012 - 32nd Annual Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2012. Proceedings*. Ed. by Reihaneh Safavi-Naini and Ran Canetti. Vol. 7417. Lecture Notes in Computer Science. Springer, 2012, pp. 868–886. DOI: 10.1007/978-3-642-32009-5_50. URL: https://doi.org/10.1007/978-3-642-32009-5_50.

[BT21] Alexandra Boldyreva and Tianxin Tang. "Privacy-Preserving Approximate k-Nearest-Neighbors Search that Hides Access, Query and Volume Patterns". In: *Proc. Priv. Enhancing Technol.* 2021.4 (2021), pp. 549–574. DOI: 10.2478/popets-2021-0084. URL: https://doi.org/10.2478/popets-2021-0084.

# Reference III


[Che+20] Hao Chen et al. "SANNS: Scaling Up Secure Approximate k-Nearest Neighbors Search". In: *29th USENIX Security Symposium, USENIX Security 2020, August 12-14, 2020*. Ed. by Srdjan Capkun and Franziska Roesner. USENIX Association, 2020, pp. 2111–2128. URL: https://www.usenix.org/conference/usenixsecurity20/presentation/chen-hao.

[DS17] Jack Doerner and Abhi Shelat. "Scaling ORAM for Secure Computation". In: *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017*. Ed. by Bhavani M. Thuraisingham et al. ACM, 2017, pp. 523–535. DOI: 10.1145/3133956.3133967. URL: https://doi.org/10.1145/3133956.3133967.

[DSZ15] Daniel Demmler, Thomas Schneider, and Michael Zohner. "ABY - A Framework for Efficient Mixed-Protocol Secure Two-Party Computation". In: *22nd Annual Network and Distributed System Security Symposium, NDSS 2015, San Diego, California, USA, February 8-11, 2015*. The Internet Society, 2015. URL: https://www.ndss-symposium.org/ndss2015/aby---framework-efficient-mixed-protocol-secure-two-party-computation.

[Erk+09] Zekeriya Erkin et al. “Privacy-Preserving Face Recognition”. In: *Privacy Enhancing Technologies, 9th International Symposium, PETS 2009, Seattle, WA, USA, August 5-7, 2009. Proceedings*. Ed. by Ian Goldberg and Mikhail J. Atallah. Vol. 5672. Lecture Notes in Computer Science. Springer, 2009, pp. 235–253. DOI: 10.1007/978-3-642-03168-7_14. URL: https://doi.org/10.1007/978-3-642-03168-7_14.

[ESJ14] Yousef Elmehdwi, Bharath K. Samanthula, and Wei Jiang. “Secure k-nearest neighbor query over encrypted data in outsourced environments”. In: *IEEE 30th International Conference on Data Engineering, Chicago, ICDE 2014, IL, USA, March 31 - April 4, 2014*. Ed. by Isabel F. Cruz et al. IEEE Computer Society, 2014, pp. 664–675. DOI: 10.1109/ICDE.2014.6816690. URL: https://doi.org/10.1109/ICDE.2014.6816690.

[Fu+19] Cong Fu et al. “Fast Approximate Nearest Neighbor Search With The Navigating Spreading-out Graph”. In: *Proc. VLDB Endow.* 12.5 (2019), pp. 461–474. DOI: 10.14778/3303753.3303754. URL: http://www.vldb.org/pvldb/vol12/p461-fu.pdf.

[FV12] Junfeng Fan and Frederik Vercauteren. “Somewhat Practical Fully Homomorphic Encryption”. In: *IACR Cryptol. ePrint Arch.* (2012), p. 144. URL: http://eprint.iacr.org/2012/144.

[Hua+11] Yan Huang et al. "Efficient Privacy-Preserving Biometric Identification". In: *Proceedings of the Network and Distributed System Security Symposium, NDSS 2011, San Diego, California, USA, 6th February - 9th February 2011*. The Internet Society, 2011. URL: https://www.ndss-symposium.org/ndss2011/efficient-privacy-preserving-biometric-identification.

[JVC18] Chiraag Juvekar, Vinod Vaikuntanathan, and Anantha P. Chandrakasan. "GAZELLE: A Low Latency Framework for Secure Neural Network Inference". In: *27th USENIX Security Symposium, USENIX Security 2018, Baltimore, MD, USA, August 15-17, 2018*. Ed. by William Enck and Adrienne Porter Felt. USENIX Association, 2018, pp. 1651–1669. URL: https://www.usenix.org/conference/usenixsecurity18/presentation/juvekar.

[Llo82] Stuart P. Lloyd. "Least squares quantization in PCM". In: *IEEE Trans. Inf. Theory* 28.2 (1982), pp. 129–136. DOI: 10.1109/TIT.1982.1056489. URL: https://doi.org/10.1109/TIT.1982.1056489.

# Reference VI


[Low99] David G. Lowe. "Object Recognition from Local Scale-Invariant Features". In: *Proceedings of the International Conference on Computer Vision, Kerkyra, Corfu, Greece, September 20-25, 1999*. IEEE Computer Society, 1999, pp. 1150–1157. DOI: 10.1109/ICCV.1999.790410. URL: https://doi.org/10.1109/ICCV.1999.790410.

[McA+15] Julian J. McAuley et al. "Image-Based Recommendations on Styles and Substitutes". In: *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, Santiago, Chile, August 9-13, 2015*. Ed. by Ricardo Baeza-Yates et al. ACM, 2015, pp. 43–52. DOI: 10.1145/2766462.2767755. URL: https://doi.org/10.1145/2766462.2767755.

[MY20] Yury A. Malkov and Dmitry A. Yashunin. "Efficient and Robust Approximate Nearest Neighbor Search Using Hierarchical Navigable Small World Graphs". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 42.4 (2020), pp. 824–836. DOI: 10.1109/TPAMI.2018.2889473. URL: https://doi.org/10.1109/TPAMI.2018.2889473.

[Pai99] Pascal Paillier. "Public-Key Cryptosystems Based on Composite Degree Residuosity Classes". In: *Advances in Cryptology - EUROCRYPT '99, International Conference on the Theory and Application of Cryptographic Techniques, Prague, Czech Republic, May 2-6, 1999, Proceeding*. Ed. by Jacques Stern. Vol. 1592. Lecture Notes in Computer Science. Springer, 1999, pp. 223–238. DOI: 10.1007/3-540-48910-X_16. URL: https://doi.org/10.1007/3-540-48910-X_16.

[SFR18] Hayim Shaul, Dan Feldman, and Daniela Rus. "Scalable Secure Computation of Statistical Functions with Applications to k-Nearest Neighbors". In: *CoRR* abs/1801.07301 (2018). arXiv: 1801.07301. URL: http://arxiv.org/abs/1801.07301.

[SFR20] Hayim Shaul, Dan Feldman, and Daniela Rus. "Secure k-ish Nearest Neighbors Classifier". In: *Proc. Priv. Enhancing Technol.* 2020.3 (2020), pp. 42–61. DOI: 10.2478/popets-2020-0045. URL: https://doi.org/10.2478/popets-2020-0045.

[SGB18] Phillipp Schoppmann, Adrià Gascón, and Borja Balle. "Private Nearest Neighbors Classification in Federated Databases". In: *IACR Cryptol. ePrint Arch.* (2018), p. 289. URL: https://eprint.iacr.org/2018/289.

[SLD21] Sacha Servan-Schreiber, Simon Langowski, and Srinivas Devadas. "Lightweight Private Similarity Search". In: *IACR Cryptol. ePrint Arch.* (2021), p. 1157. URL: https://eprint.iacr.org/2021/1157.

[Son+15] Ebrahim M. Songhori et al. "Compacting privacy-preserving k-nearest neighbor search using logic synthesis". In: *Proceedings of the 52nd Annual Design Automation Conference, San Francisco, CA, USA, June 7-11, 2015*. ACM, 2015, 36:1–36:6. DOI: 10.1145/2744769.2744808. URL: https://doi.org/10.1145/2744769.2744808.

[Sta21] Nina Mesing Stausholm. "Improved Differentially Private Euclidean Distance Approximation". In: *PODS'21: Proceedings of the 40th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems, Virtual Event, China, June 20-25, 2021*. Ed. by Leonid Libkin, Reinhard Pichler, and Paolo Guagliardo. ACM, 2021, pp. 42–56. DOI: 10.1145/3452021.3458328. URL: https://doi.org/10.1145/3452021.3458328.

[Yao86] Andrew Chi-Chih Yao. "How to Generate and Exchange Secrets (Extended Abstract)". In: *27th Annual Symposium on Foundations of Computer Science, Toronto, Canada, 27-29 October 1986*. IEEE Computer Society, 1986, pp. 162–167. DOI: 10.1109/SFCS.1986.25. URL: https://doi.org/10.1109/SFCS.1986.25.

[ZS21] Martin Zuber and Renaud Sirdey. "Efficient homomorphic evaluation of k-NN classifiers". In: *Proc. Priv. Enhancing Technol.* 2021.2 (2021), pp. 111–129. DOI: 10.2478/popets-2021-0020. URL: https://doi.org/10.2478/popets-2021-0020.

# Why is clustering algorithm appealing?

Compared with graph-based ANN algorithms, in clustering algorithms

- Most memory accesses are adaptive
    - SOTA graph-based algorithms[MY20; Fu+19] query by following edges in certain carefully constructed graphs
    - Certainty (distinguished by queries) $\rightarrow$ non-adaptive memory access $\rightarrow$ more rounds of interaction protected by ORAM
    - hence inefficient :(
- Many distances are computed at once (large batches of points)
    - On the contrary, those graph-based approaches adaptively compute *individual* distance
    - Hard to accelerate via vectorized SIMD optimization