(Conjunctive) Search over High-Dimensional Data via Quantitative and Hierarchical Similarities

Anonymous

***Abstract*—The flexibility of cloud stimulates data owners to outsource their numerous feature-rich data, such as videos and images, to remote servers. This raises a privacy concern since some data is sensitive in nature and the owners lose their physical control. Encryption is a common approach for ensuring privacy, yet it hinders retrieval of video and images, especially for those with partial similarities. Retrieving (unencrypted) data has been widely studied, while receiving rare attention in encrypted domain, at best in a limited form of similarity searches on images via traditional hashing codes. The key challenge is to keep the similarities in the encryption domain and match the encrypted data fast and accurately. To fill this gap, we propose the encrypted high-dimensional data retrieval framework that enables a querier to find data (partially) similar to an image query. Our framework supports any type of data and retains quantitative similarities over encrypted data. We introduce a hierarchical structure of encrypted searching via multi-keyword queries, thus beneficial to both the similarity and efficiency. (To be continued)**

I. Introduction

With the explosive growth of multimedia-sharing services and social networking applications, massive visual data is generated daily and probably attracts attention to searching the relevant multimedia. The high-dimensional visual data, such as images and videos, is more complicated to extract its information than the textual contents; And thus, visuality brings difficulty in searching and matching the relevant/similar ones as requested. Deep neural networks (DNNs) have shown great success in handling complex data by capturing abstract features and statistical attributes, which stimulates the dimensional-reduction of data for efficient indexing and searching. As a result, hashing-based searching via DNNs can simultaneously quantize retrieval similarity over data and reduce the computational cost of matching, which indicates a promising direction.

Albeit the achievements in retrieval over raw data, a client expects to alleviate local storage costs, and therefore, naturally outsource her data to a remote (cloud) server. Despite its convenience and flexibility, privacy has emerged as a critical concern instantly since remote facilities are not fully trusted or even attacked by illegal commercial competitors. Searchable encryption (SE) enables to store an encrypted database on the server and meantime retains efficient accessibility by relaxing the privacy requirements reasonably. However, existing SE schemes are designed for exact keyword matching but rarely consider the property of visual and semantic similarity over high-dimensional data. Only a limited number of works solve encrypted image searching with clumsy/old tools at the early stage. This work aims to resolve this challenge and explore how to ***achieve similarity searching over encrypted high-dimensional data securely and accurately***.

A. Deep Hashing and Retrieval (*Unencrypted*)

Hashing maps high-dimensional representative data in arbitrary size to compact hash codes, which brings remarkable efficiency for data storage and retrieval. DNNs have Inherent excellence in representing data by extracting high-level features and reducing the dimensions [1], [2], [3], [4], [5], [6]. Thus, Applying DNNs to hashing, deep hashing (DH) for simplification, shows great success in large-scale image retrieval [5], [7], [8] and comprehensive video retrieval [9], [10], [11], [12]. Roughly, two branches of implementing deep hashing are supervised models [3], [1], [13], [14], [15] that learn hash functions with semantic labels and unsupervised models [16], [17], [18], [4], [6] that minimize the objective functions to quantize the similarities, respectively. Deep hashing is more flexible and efficient than traditional hashing methods, such as locality-sensitive hashing (LSH). The reason is that traditional methods need to assume the underlying (uniform) distributions of the data restrictively and cannot be applied in kernel space directly [19]. This paper adopts deep hashing [5] for capturing high-level semantic-and/or-visual similarities quantitatively, which is thus suitable for encrypted searching accurately.

B. State-of-Art Searchable Encryption and Similarity

Encryption eliminates the privacy concern about outsourcing raw data to a remote server, but it harms the accessibility to desired data requested by a client and excellent performance achieved in the plaintext domain. Specifically, the most advanced SE schemes [20], [21], [22], [23], [24] have a good ability to perform exact keyword-level searching and to access the encrypted data with the dynamic property; However, directly applying them to high-dimensional data retrieval is not satisfying enough since exact matching overlooks the similarity over raw data during the index building, *e.g.*, a small distance of predefined measuring metrics. Existing similarity-SE solutions [25], [26], [27], [28], [29], [30] using traditional evaluative metrics (*e.g.*, LSH) to measure similarities can only capture the low-level similarities, such as pixel-level matching. Besides, a small set of works [31], [26] group similar images into a unique (encrypted) index for subsequent similarity searching (*i.e.*, transforming similarity search to exact matching), and nevertheless, lose the similarity over encryption. These solutions are still inflexible and limit the similarity measurements in a preset manner (and thus no dynamical search). As a side consideration, a scenario-motivated solution of similarity searching over encryption is also expected to satisfy the privacy guarantee the same as up-to-date SE schemes, saying forward-and-backward privacy [20], [22].

C. Observations and Objectives

****

on Similarity Search over Encryption;

Deep hash ave 10% captures data distribution Use NDSS 21 - conjunctive style; build the bridge; For a practical application, we need to consider the real-word scenario. Briefly, a scheme keeps (encrypted) similarities and searches fast:

- Real-world data is high-dimensional (or multi-dimensional features), considering multi-keyword search not a single one.
- Process the data to extract the quantitative similarities via deep hash or data distillation: the existing best ones are about 90% (recall-precision trade-off, but better than traditional hashing such as LSH).
- Keep the similarities in the encrypted domain, and design a matching algorithm for searching (encrypted) similar data, thus achieving partially similar searching (considering additive sharing or homomorphic encryption). HE for split hashing and then a ciphertext (find one)
- More detailed similarities, high-dimensional "keyword" space, thus considering hierarchy.
- SE supports default privacy (forward-and-backward privacy) as a side contribution/requirement.
- Reduce the client's computation if possible (no idea now).

D. Solutions Overview and Contributions

II. Related Works

A. Data Representation and Multimedia Retrieval

1) Deep Hashing and Data Distillation: general DH and basic requirement now and representing data via compression/distilling/approximation(ANN); detailed branches from algorithm-level improvement

2) Indexing and Searching Algorithms for Retrieval: summarize the plaintext matching algorithm; From traditional ones to the NN ones

B. Searchable Encryption

1) Searchable Encryption and Dynamic Search: general SE and basic requirement now

2) Security of DSE:

3) Similarity Search: seminal work is feature-rich data

4) Conjunctive Queries:

III. Background

A. Deep Hashing via Central Similarity

Let $f : x \in \mathbb{R}^D \mapsto h \in \{0,1\}^K$ be the hash function, where $x \in \mathcal{X} = \{\{x_i\}_{i=1}^N, L\}$ is data point and L is the semantic label set. Given the following hash centers, the hash codes are generated by a deep neural network via similarity estimation [5].

Definition 1: (Hash Center). Let hash centers be a set of points $\mathcal{C} = \{c_i\}_{i=1}^m \subset \{0,1\}^K$ in the K-dimensional Hamming space with an average pairwise distance satisfying,

$$\frac{1}{T}\sum_{i\neq j}^{m} D_H(c_i, c_j) \geq \frac{K}{2},$$

where D_H is the Hamming distance, m is the number of hash centers, and T is the number of combinations of different c_i and $c_j \in \mathcal{C}$.

B. General Notion of Searchable Encryption

We follow the commonly-used definition of searchable encryption [22] and introduce it in the following. This definition formalizes the functionalities of searching and updating over encrypted data on a remote server.

Definition 2: A dynamic symmetric searchable encryption scheme (SSE) $\Sigma = \mathsf{Setup}, \mathsf{Search}, \mathsf{Update}$ consists of algorithm Setup, and protocols $\mathsf{Search}, \mathsf{Update}$ between a client and a server:

- $\mathsf{Setup}(\lambda)$ is an algorithm that on input the security parameter outputs $(K, \sigma, \mathsf{EDB})$ where K is a secret key for the client, σ is the client's local state, and EDB is an (empty) encrypted database that is sent to the server.
- $\mathsf{Search}(K, \mathsf{q}, \sigma; \mathsf{EDB})$ is a protocol for searching the database, where q is the search query. The client's output is $\mathsf{DB}(\mathsf{q})$, and the protocol may also modify K, σ, EDB.
- $\mathsf{Update}(K, \mathsf{op}, \mathsf{in}, \sigma; \mathsf{EDB})$ is a protocol for inserting an entry to or removing an entry from the database. Operation op can be add or del; Input in consists of a file identifier id and a keyword w. The protocol may modify K, σ, EDB.

Informally, an SSE is correct if the returned results are correct for every query q. The confidentiality of an SSE scheme is parameterized by a leakage function $\mathcal{L}$ that captures the information being revealed to an adversarial server throughout the protocol execution. The security of an SSE is guaranteed if revealing nothing of the database DB except for the reasonable predefined leakage. The formal definition is captured by a standard real/ideal experiment with two games below.

Definition 3: (Adaptive Security) An SSE scheme Σ is adaptively-secure with respect to leakage function $\mathcal{L}$, *iff* for any PPT adversary $\mathcal{A}$ issuing polynomial number of queries q, there exists a stateful PPT simulator $\mathcal{S}$ such that,

$$\Pr[\mathsf{Real}_{\mathcal{A}}^{\Sigma}(\lambda, \mathsf{q}) = 1] - \Pr[\mathsf{Ideal}_{\mathcal{A},\mathcal{S},\mathcal{L}}^{\Sigma}(\lambda, \mathsf{q}) = 1] \leq \mathsf{negl}(\lambda)$$

where $\mathsf{negl}(\lambda)$ is negligible function with parameter λ.

C. Forward-and-Backward Privacy with Leakages

Forward-and-backward privacy controls what information can be leaked during the updates in dynamic SSE schemes. Forward privacy ensures that an update cannot be linked with any previous queries, formally defined below.

Definition 4: (Forward Privacy) An $\mathcal{L}$-adaptively-secure DSE scheme is forward private iff the update leakage satisfies $\mathcal{L}_{\mathsf{Update}}(\mathsf{op}, w, \mathsf{id}) = \mathcal{L}'(\mathsf{op}, \mathsf{id})$. $\mathcal{L}'$ is a stateless function, and leaks no keyword information.

Backward privacy limits the leakage from search for a keyword w for which some identifiers/entries have been previously deleted. Let $\mathcal{Q}$ be a list that has one entry for each query executed. The entry for a search is of the form (u, w), where u is the query timestamp and w is the keyword for searching. An update entry of modified file id is $(u, \mathsf{op}, (w, \mathsf{id}))$. For a keyword w that have been added to $\mathbf{DB}$ and have not been subsequently deleted, we define $\mathbf{TimeDB}(w)$ to be a function that returns the list of all timestamp/file-identifier pairs of keyword w in the following.

$$\mathbf{TimeDB}(w) = \{(u, \mathsf{id}) | (u, \mathsf{add}, (w, \mathsf{id})) \in \mathcal{Q} \text{ and } \forall u', (u', \mathsf{del}, (w, \mathsf{id})) \notin \mathcal{Q}\}$$

$\mathbf{Updates}(w)$ is a function that returns the timestamp of all insertion and deletion operations for w in $\mathcal{Q}$.

$$\mathbf{Updates}(w) = \{u | (u, \mathsf{add}, (w, \mathsf{id})) \in \mathcal{Q}\}\{u | (u, \mathsf{del}, (w, \mathsf{id})) \in \mathcal{Q}\}$$

$\mathbf{DelHist}(w)$ is a function that returns the history of deleted entries by giving all pairs of insertion timestamp and deletion timestamp to the adversary, which reveals the explicit matching of deletion and addition.

$$\mathbf{DelHist}(w) = \{(u^{\mathsf{add}}, u^{\mathsf{del}}) | \ \exists \ \mathsf{id} : (u^{\mathsf{add}}, \mathsf{add}, (w, \mathsf{id})) \in \mathcal{Q} \text{ and } (u^{\mathsf{del}}, \mathsf{del}, (w, \mathsf{id})) \in \mathcal{Q}\}$$

Given the functional leakages above, backward privacy is formally defined in the following.

Definition 5: An $\mathcal{L}$-adaptively-secure SSE scheme has backward privacy (BP) over stateless leakage functions $\mathcal{L}', \mathcal{L}''$:
Type-I (BP-I): iff $\mathcal{L}^{\mathsf{Updt}}(\mathsf{op}, w, \mathsf{id}) = \mathcal{L}'(\mathsf{op})$ and $\mathcal{L}^{\mathsf{Srch}}(w) = \mathcal{L}''(\mathbf{TimeDB}(w), a_w)$.
Type-II (BP-II): iff $\mathcal{L}^{\mathsf{Updt}}(\mathsf{op}, w, \mathsf{id}) = \mathcal{L}'(\mathsf{op}, w)$ and $\mathcal{L}^{\mathsf{Srch}}(w) = \mathcal{L}''(\mathbf{TimeDB}(w), \mathbf{Updates}(w))$.
Type-III (BP-III): iff $\mathcal{L}^{\mathsf{Updt}}(\mathsf{op}, w, \mathsf{id}) = \mathcal{L}'(\mathsf{op}, w)$ and $\mathcal{L}^{\mathsf{Srch}}(w) = \mathcal{L}''(\mathbf{TimeDB}(w), \mathbf{DelHist}(w))$.

IV. A Warm-Up solution and Quagmire

This section introduces a straightforward solution to combine deep hashing and searchable encryption. For generating the hash codes, we adopt the central hashing [5] which is generic and applicable to both image and video hashing scenarios. We start with a simple approach [22] of encrypted searching satisfying forward-and-backward privacy, and try to apply it over the hash codes in a black-box manner. The nature of this method is to transform similarity search to exact keyword search and meantime keep similarity relation in hash codes. Later, we analyze the insufficiency in practice usage and explore the challenges on effective similarity search.

A. Revisiting and A Straightforward Combination

We follow the common protocols of SSE and construct a simple approach of keyword-approximating searches over encrypted data. We design Maeve which supports large-scale searching on images and videos. Instead of matching them exactly (like matching a textual word in an email), we transform the multi-dimensional media into hash codes with quantitative similarities both in semantics and visuals. In a real-world image, one unique cat exist in different images but always has varied actions. It is not trivial to find an reasonable method to represent the multimedia and design an SSE scheme atop it. Maeve leaks nothing to the server during searches and updates (additions and deletions) except for a set of (predefined) reasonable leakages. Following the Definition 4,5, a key-value dictionary stores encrypted values of the form $(\mathsf{id}, \mathsf{op})$ for recording the time at which updates taking place. In the phase of building the encrypted database, the keys (*i.e.*, locations of storing the responding values in a dictionary) are generated via a pseudorandom function (PRF) and specified by a keyword w. Later, the client can efficiently generate the set of all locations related to this specific keyword w for a given search operation.

Setup. The setup algorithm generates a secret key K on the inputting security parameter λ as a standard SE scheme. The client initiates two empty maps (**DictW**, **FileCnt**) for storing encrypted entries, where **DictW** is sent to the server and **FileCnt** is kept locally. **FileCnt** aims to count the number of updates that have taken place in relation to w. The multimedia data is represented by hash codes, *e.g.*, an image being transformed into multiple codes. To differentiate the exact keyword in the textual scenario, we denote the hash codes with similarity attributes by $\breve{w}$ in the following.

Update. During an update, the client has a keyword $\breve{w}$, a file identifier id, and an operation op. Given the accessibility of key K and local state **FileCnt**, the client checks the $\mathbf{FileCnt}[\breve{w}]$ and increments the counter of $\breve{w}$ by 1. Later, the client runs the PRF G with the key K twice to get $G_K(w, \mathbf{FileCnt}[\breve{w}] \| 0)$ and $G_K(w, \mathbf{FileCnt}[\breve{w}] \| 1)$. The output $G_K(w, \mathbf{FileCnt}[\breve{w}] \| 0)$ denotes the location addr to store the encrypted value of $(\mathsf{id}, \mathsf{op})$, while the other output $G_K(w, \mathbf{FileCnt}[\breve{w}] \| 0)$ encrypts $(\mathsf{id}, \mathsf{op})$ to be val via XOR operation. At last, the server stores $\mathbf{DictW}[\mathsf{addr}] = \mathsf{val}$ once it receives the pair $(\mathsf{addr}, \mathsf{val})$.

Search. When searching the files containing the $\breve{w}$, the client first looks up the counter $\mathbf{FileCnt}[\breve{w}]$ and then utilizes PRF G to generate all locations relevant to the $\breve{w}$ in the **DictW**. Upon receiving the locations, the server retrieves all encrypted files with respect to the $\breve{w}$ and sends them back. The client decrypts the encryptions with $G_K(\breve{w}, i \| 1)_{i \in \{1, \ldots, \mathbf{FileCnt}[\breve{w}]\}}$ to get the final results.

B. Challenges and Objectives

Maeve achieves similarity searching to some content by relying on the method of hashing in the plaintext domain. The security follows the definition of forward-and-backward privacy, where forward privacy comes from the pseudorandomness of PRF G and backward privacy only reveals update time. The simple Maeve neither increases any burden on performing searching and updating nor weakens the security, which brings the same computational complexity and communication complexity like a standard SE. However, this method is not perfect enough in the scenario of retrieving high-dimensional data since the encrypted data loses the distinctiveness of hash codes (*e.g.*, the small hamming distance on hash codes) produced by the advanced techniques in the plaintext domain. In summary, the success of (plain-text) image/video retrieval have not be shown or applicable in the context of searchable encryption.

1) Quantitative similarity. By directly applying a standard SE, Maeve resorts to prevailing (plaintext) preprocessing/hashing techniques singly for seeking similarity searches but ineluctably forfeits the quantitative relation over encryptions. The origin of suffering such losses in similarity SE comes from the pseudorandomness of a PRF and the ciphertext indistinguishability to the randomness. Homomorphic encryption allowing a limited number of additions and/or multiplications can be deemed a feasible direction for supporting similarity comparison on encrypted data. However, replacing symmetric encryption with homomorphic encryption is not that promising on account of the heavy computation and large size of encryption. In view of retrieving high-dimensional data, this paper leverages practical search efficiency by introducing property-preserving hash functions (PPH), a new cryptographic primitive recently introduced by Boyle *et al.* [32].

2) More leakages and adaptive security.

We analyze functionality and security abovexxx

C. Formalization and Security Model

Definition 6: (Keyword-Approximating SE) functionality and security implying k leq n/2,3,4

Definition 7: (Quasi-Optimality in KASE) implying k leq n/2,3,4 in search/storage

V. System Overview and Construction

A. High-Level Overview

system functionality

VI. Application and Implementation on Image

This is application idea so the followings are the application in specific scenario and experiments (combine implementation and experimental evaluation) select SE and Hash targeting to scenario

VII. Application and Implementation on Video

VIII. Related Works

References

[1] W. Li, S. Wang, and W. Kang, "Feature learning based deep supervised hashing with pairwise labels," in *IJCAI*, 2016.

[2] Y. Shen, L. Liu, F. Shen, and L. Shao, "Zero-shot sketch-image hashing," in *CVPR*, 2018.

[3] Z. Chen, X. Yuan, J. Lu, Q. Tian, and J. Zhou, "Deep hashing via discrepancy minimization," in *CVPR*, 2018.

[4] E. Yang, T. Liu, C. Deng, W. Liu, and D. Tao, "Distillhash: Unsupervised deep hashing by distilling data pairs," in *CVPR*, 2019.

[5] L. Yuan, T. Wang, X. Zhang, F. E. H. Tay, Z. Jie, W. Liu, and J. Feng, "Central similarity quantization for efficient image and video retrieval," in *CVPR*, 2020.

[6] H. Hu, L. Xie, R. Hong, and Q. Tian, "Creating something from nothing: Unsupervised knowledge distillation for cross-modal hashing," in *CVPR*, 2020.

[7] R. Xia, Y. Pan, H. Lai, C. Liu, and S. Yan, "Supervised hashing for image retrieval via image representation learning," in *AAAI*, 2014.

[8] H. Zhu, M. Long, J. Wang, and Y. Cao, "Deep hashing network for efficient similarity retrieval," in *AAAI*, 2016.

[9] Y. Gu, C. Ma, and J. Yang, "Supervised recurrent hashing for large scale video retrieval," in *MM*, 2016.

[10] J. Qin, L. Liu, M. Yu, Y. Wang, and L. Shao, "Fast action retrieval from videos via feature disaggregation," in *Comput. Vis. Image Underst.*, 2017.

[11] V. E. Liong, J. Lu, Y. Tan, and J. Zhou, "Deep video hashing," *IEEE Trans. Multim.*, 2017.

[12] S. Qiao, R. Wang, S. Shan, and X. Chen, "Deep heterogeneous hashing for face video retrieval," *IEEE Trans. Image Process.*, 2020.

[13] V. Gattupalli, Y. Zhuo, and B. Li, "Weakly supervised deep image hashing through tag embeddings," in *CVPR*, 2019.

[14] Y. Zhan, X. Luo, Y. Wang, and X. Xu, "Supervised hierarchical deep hashing for cross-modal retrieval," in *MM*, 2020.

[15] D. Wang, H. Huang, C. Lu, B. Feng, G. Wen, L. Nie, and X. Mao, "Supervised deep hashing for hierarchical labeled data," in *AAAI*, 2018.

[16] B. Dai, R. Guo, S. Kumar, N. He, and L. Song, "Stochastic generative hashing," in *ICML*, 2017.

[17] C. Deng, E. Yang, T. Liu, J. Li, W. Liu, and D. Tao, "Unsupervised semantic-preserving adversarial hashing for image search," in *IEEE Trans. Image Process.*, 2019.

[18] K. G. Dizaji, F. Zheng, N. Sadoughi, Y. Yang, C. Deng, and H. Huang, "Unsupervised deep generative adversarial hashing network," in *CVPR*, 2018.

[19] B. Kulis and T. Darrell, "Learning to hash with binary reconstructive embeddings," in *NeurIPS*, 2009.

[20] R. Bost, "$\sum o\varphi o\varsigma$: Forward secure searchable encryption," in *CCS*, 2016.

[21] K. S. Kim, M. Kim, D. Lee, J. H. Park, and W. Kim, "Forward secure dynamic searchable symmetric encryption with efficient updates," in *CCS*, 2017.

[22] J. G. Chamani, D. Papadopoulos, C. Papamanthou, and R. Jalili, "New constructions for forward and backward private symmetric searchable encryption," in *CCS*, 2018.

[23] I. Demertzis, J. G. Chamani, D. Papadopoulos, and C. Papamanthou, "Dynamic searchable encryption with small client storage," in *NDSS*, 2020.

[24] Q. Song, Z. Liu, J. Cao, K. Sun, Q. Li, and C. Wang, "SAP-SSE: protecting search patterns and access patterns in searchable symmetric encryption," *IEEE Trans. Inf. Forensics Secur.*, 2021.

[25] Q. Wang, M. He, M. Du, S. S. M. Chow, R. W. F. Lai, and Q. Zou, "Searchable encryption over feature-rich data," *IEEE Trans. Dependable Secur. Comput.*, 2018.

[26] C. Guo, X. Tang, Y. Chen, P. Tian, and M. Z. A. Bhuiyan, "An efficient distributed approach on high dimensional data similarity searchable encryption," in *TrustCom*, 2018.

[27] M. Li, C. Jia, and W. Shao, "Blockchain based multi-keyword similarity search scheme over encrypted data," in *SecureComm*, 2020.

[28] M. Li, M. Zhang, Q. Wang, S. S. M. Chow, M. Du, Y. Chen, and C. Li, "Instantcryptogram: Secure image retrieval service," in *INFOCOM*, 2018.

[29] Z. Wan and R. H. Deng, "Vpsearch: Achieving verifiability for privacy-preserving multi-keyword search over encrypted cloud data," in *IEEE Trans. Dependable Secur. Comput.*, 2018.

[30] C. Guo, W. Liu, X. Liu, and Y. Zhang, "Secure similarity search over encrypted non-uniform datasets," in *IEEE Trans. on Cloud Comput.*, 2020.

[31] Y. Li, J. Ma, Y. Miao, Y. Wang, X. Liu, and K.-K. R. Choo, "Similarity search for encrypted images in secure cloud computing," in *IEEE Trans. on Cloud Comput.*, 2020.

[32] E. Boyle, R. LaVigne, and V. Vaikuntanathan, in *ITCS*, 2019.