Privacy-Preserving Outsourced Media Search Using Secure Sparse Ternary Codes

Behrooz Razeghi and Slava Voloshynovskiy
Stochastic Information Processing Group
Department of Computer Science, University of Geneva, Switzerland
{behrooz.razeghi, svolos}@unige.ch

***Abstract*—In this paper, we propose a privacy preserving framework for *outsourced media search* applications. Considering three parties, a data owner, clients and a server, the data owner outsources the description of his data to an external server, which provides a search service to clients on the behalf of the data owner. The proposed framework is based on a sparsifying transform with ambiguization, which consists of a *trained linear map*, an *element-wise nonlinearity* and a *privacy amplification*. The proposed privacy amplification technique makes it infeasible for the server to learn the structure of the database items and queries. We demonstrate that the privacy of the database outsourced to the server as well as the privacy of the client are ensured at a low computational cost, storage and communication burden.**

***Keywords*—data privacy; sparse approximation; transform learning; ambiguization; content-based retrieval.**

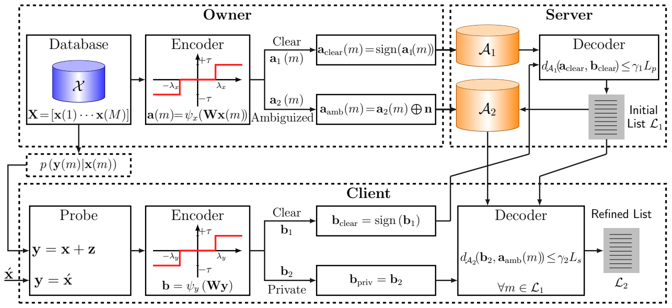
Owner
Server
Client
Database
$\mathcal{X}$
$\mathbf{X}=[\mathbf{x}(1)\cdots\mathbf{x}(M)]$
Encoder
$+\tau$
$-\tau$
$-\lambda_x$
λ_x
$\mathbf{a}(m)=\psi_x(\mathbf{W}\mathbf{x}(m))$
Clear
$\mathbf{a}_1(m)$
$\mathbf{a}_2(m)$
Ambiguized
$\mathbf{a}_{\text{clear}}(m)=\text{sign}(\mathbf{a}_1(m))$
$\mathbf{a}_{\text{amb}}(m)=\mathbf{a}_2(m)\oplus\mathbf{n}$
$\mathcal{A}_1$
$\mathcal{A}_2$
Decoder
$d_{\mathcal{A}_1}(\mathbf{a}_{\text{clear}},\mathbf{b}_{\text{clear}})\leq\gamma_1 L_p$
Initial List $\mathcal{L}_1$
$p(\mathbf{y}(m)|\mathbf{x}(m))$
Probe
$\mathbf{y}=\mathbf{x}+\mathbf{z}$
$\acute{\mathbf{x}}$
$\mathbf{y}=\acute{\mathbf{x}}$
Encoder
$+\tau$
$-\tau$
$-\lambda_y$
λ_y
$\mathbf{b}=\psi_y(\mathbf{W}\mathbf{y})$
Clear
$\mathbf{b}_1$
$\mathbf{b}_2$
Private
$\mathbf{b}_{\text{clear}}=\text{sign}(\mathbf{b}_1)$
$\mathbf{b}_{\text{priv}}=\mathbf{b}_2$
Decoder
$d_{\mathcal{A}_2}(\mathbf{b}_2,\mathbf{a}_{\text{amb}}(m))\leq\gamma_2 L_s$
$\forall m\in\mathcal{L}_1$
Refined List
$\mathcal{L}_2$

Fig. 1: Block diagram of the proposed outsourced media search.

I. Introduction

The main challenge for outsourced media search is that the server must remain capable of performing the search service whilst knowing little about the owner's data and the clients' interests. This paper presents a new privacy preserving strategy for the third party outsourced media search problem based on the recently proposed concept of sparse approximation with ambiguization [1]. Our main contribution consists of a novel framework based on *Scaled Sparse Ternary Coding (SSTC)* and *partial ambiguization*. In our framework the owner and the client compute the sparse representations from the media data that they own using a *trained linear* map followed by a *element-wise nonlinearity*. Each sparse representation is split into two parts. One part left almost in-the-clear and the other part is ambiguized. Both in-the-clear and ambiguized representations of the owner's database are send to the server. The in-the-clear part is used by the server for the initial similarity search. The ambiguized part is used by the client to refine the list. A similar idea based on DCT transform with binarization has been proposed in [2].

Given a database, the dictionary learning problem is the task of learning a transform that allows sparse representations of the data. The main downside of the learned transforms is that they lack structure and therefore are not computationally efficient, unlike the classical well-known transforms such as Fourier, Hadamard and so forth. Analogous to [1], in this paper we construct an orthonormal structured dictionary. Then we project the data onto the column space of the dictionary and keep the largest S_x components in magnitude to obtain the provable best S_x-term approximation. We construct the structured sparsifying transform that may keep or extend the dimension of the original signal. We impose no restrictions on the input data, i.e., we assume that as an input we might have raw data, extracted features using any known hand crafted methods, aggregated local descriptors based on BoW, FV, VLAD [3]–[5], etc., or from the last layers of deep nets [6].

In comparison to the most recent outsourced media search framework based on robust hashing and partial encryption [8], there are several fundamental differences:
1) *Transformation*: In [8], the authors used a dimensionality reduction transform with random entries. In this paper, we consider a square or an over-complete transform, which may extend the dimensions. That is, instead of projecting data to a lower dimension, we project our data to the same or higher dimensions. Moreover, our transform is trained using the sparsifying Procrustes problem to ensure an optimal sparse representation that is information preserving in general, whereas the locality-sensitive hashing (LSH) in [8] might preserve the distances only under certain conditions of the Johnson-Lindenstrauss Lemma.
2) *Codes*: In [8], the codes are dense and binary, whereas in our method the codes are sparse and scaled ternary, which form a basis of our ambiguization framework.
3) *Encryption*: In [8], the authors used standard encryption for preserving privacy. However, we simply add ambiguization noise to the nonzero components of the sparse representation. The standard encryption has more computation and communication cost in comparison to our ambiguization method.
4) *Decryption*: In [8], the returned encrypted hash values must be decrypted at the client side in order to perform similarity search. However, in our method the client just computes the similarity measure on the support of its sparse representation that is more computationally efficient.

A. Notation

The superscript $(\cdot)^T$ stands for the transpose and $(\cdot)^\dagger$ stands for the pseudo-inverse. Vectors and matrices are denoted by boldface lower-case ($\mathbf{x}$) and upper-case ($\mathbf{X}$) letters, respectively. We consider the same notation for a random vector $\mathbf{x}$ and its realization. The difference should be clear from the context. x_i denotes the i-th entry of vector $\mathbf{x}$. For a matrix $\mathbf{X}$, $\mathbf{x}(j)$ denotes the j-th column of $\mathbf{X}$. We use the notation $[N]$ for the set $\{1, 2, ..., N\}$ and $\operatorname{card}(\mathcal{S})$ for the cardinality of a set $\mathcal{S}$.

B. Outline of the Paper

The remainder of the paper is organized as follows. In Section II, the problem formulation is introduced. Then, in Section III we present our framework. We provide the privacy performance in Section IV. Finally, conclusions are drawn in Section V.

II. Problem Formulation

Consider that a owner has a collection of M (raw) feature vector $\mathbf{x}(m), m \in [M]$ in the database $\mathbf{X} = [\ \mathbf{x}(1), \ \cdots\ ,\mathbf{x}(m), \ \cdots\ ,\mathbf{x}(M)\ ]$, where each feature vector $\mathbf{x}(m), m \in [M]$ from a set $\mathcal{X} \subset \mathbb{R}^N$ is a random vector with distribution $p(\mathbf{x})$ and bounded variance $\sigma^2_{\mathbf{x}}$. The user has a query $\mathbf{y}(m) \in \mathbb{R}^N$ which is a noisy version of $\mathbf{x}(m)$, i.e., $\mathbf{y}(m) = \mathbf{x}(m) + \mathbf{z}$, where we assume $\mathbf{z} \in \mathbb{R}^N$ is a Gaussian noise vector with distribution $\mathcal{N}\big(\mathbf{0}, \boldsymbol{\Sigma}_{\mathbf{z}} = \sigma^2_{\mathbf{z}}\mathbf{I}_N\big)$. The user is interested in some information about the subset $\mathcal{L}(\mathbf{y})$ of the γ-NN (or γ-ANN) of $\mathbf{y}$. The owner subcontracts the similarity search to an entity called the server.

III. Proposed Framework

A. Framework Overview

Our framework is composed of the following five steps (Fig. 1):

1) Preparation at Owner Side: The owner computes the sparse representations from the media data that he owns using the *trained linear* map followed by the *element-wise nonlinearity*. Each sparse representation is split into two parts. One part is left in-the-clear in sparsified form and the other part is sparsified and ambiguized. These representations are send to the server.

2) Indexing at Server Side: The server indexes the in-the-clear parts of the received sparse codes to a database.

3) Querying at Client Side: The client computes a sparse representation from his query data using the same *trained linear* map followed by the *element-wise nonlinearity*. Then, the client sends the signs of a fraction of its sparse representation to the server. The indices of these components correspond to the in-the-clear part components of the sparse representation generated by the owner.

4) Searching at Server Side: The server runs a similarity search to identify the sparse representations that are most similar to the query. Similarity is computed using the received partial query code and the in-the-clear parts of the database codes. Finally, the ambiguized parts of the most similar codes are send back to the client, along with the distance information computed at the server for the in-the-clear parts.

5) Refining at Client Side: The client runs a similarity search on the *support* (non-zero components) of his sparse code. The final list is computed using similarity search within the initial list.

B. Scaled Sparse Ternary Coding

In [1], we use a transform model [9] for the sparse representation of vectors at the enrollment and identification phases. This model suggests that a feature vector $\mathbf{x}(m) \in \mathbb{R}^N$ is approximately sparsifiable using a transform $\mathbf{W} \in \mathbb{R}^{L\times N}$, that is $\mathbf{W}\mathbf{x}(m) = \mathbf{a}(m) + \mathbf{e}_{\mathbf{a}}$, where $\mathbf{a}(m) \in \mathbb{R}^L$ is sparse, i.e., $\|\mathbf{a}(m)\|_0 \ll L$, and $\mathbf{e}_{\mathbf{a}} \in \mathbb{R}^L$ is the representation error of the feature vector or residual in the *transform domain*. The sparse coding problem for this model is a direct constraint projection problem. This sparse approximation is as follows:

$$\widehat{\mathbf{a}}(m) = \underset{\mathbf{a}(m)\in\mathcal{A}^L}{\arg\min} \ \|\mathbf{W}\mathbf{x}(m) - \mathbf{a}(m)\|_2^2 + \lambda\Omega\left(\mathbf{a}(m)\right), \forall m \in [M]\,. \tag{1}$$

The above direct problem has a closed-form solution for any of the two important regularizers $\Omega(\cdot) = \|\cdot\|_0$ or $\Omega(\cdot) = \|\cdot\|_1$. Analogous to [1], we consider the ℓ_0-"norm" as our sparsity-inducing penalty. In this case, the solution $\widehat{\mathbf{a}}(m)$ is obtained exactly by hard-thresholding the projection $\mathbf{W}\mathbf{x}(m)$ and keeping the S_x entries of the largest magnitudes while setting the remaining low magnitude entries to zero. For this purpose, we define an intermediate vector $\mathbf{f}(m) \triangleq \mathbf{W}\mathbf{x}(m) \in \mathbb{R}^L$ and denote by λ_x the S_x-th largest magnitude amongst the set $\{|f_1(m)|, ..., |f_L(m)|\}$. Then the closed-form solution is achieved by applying a hard-thresholding operator to $\mathbf{f}(m)$, which is defined as $\mathbf{a}_H(m) = H_{\lambda_x}(\mathbf{f}(m)) = \mathbb{1}_{|f_l(m)|\geq\lambda_x}\mathbf{f}(m), \forall m \in [M], \forall l \in [L]$. In [1], the authors consider the alphabet of sparse representation vectors as $\mathcal{A} = \{-1, 0, +1\}$ and apply the ternary hash mapping to $H_{\lambda_x}(\mathbf{W}\mathbf{x}(m))$ as:

$$\mathbf{a}_T(m) \triangleq T_{\lambda_x}(\mathbf{W}\mathbf{x}(m)) \in \{-1, 0, +1\}^L, \ \forall m \in [M]\,, \tag{2}$$

where $T_{\lambda_x}(\mathbf{W}\mathbf{x}(m)) = \operatorname{sign}\left(H_{\lambda_x}(\mathbf{W}\mathbf{x}(m))\right)$.

In order to reduce the information loss of ternary hash mapping, we propose the *Scaled Ternary Sparse Coding (SSTC)* scheme, which enhances the accuracy of similarity search. It is clear that, given a fixed λ_x (or S_x), the operator $T_{\lambda_x}(\cdot)$ imposes a greater loss of information in comparison to the operator $H_{\lambda_x}(\cdot)$. By employing SSTC, we make a balance between the sparse vector $\mathbf{a}_H(m)$ and ternarized sparse vector $\mathbf{a}_T(m)$. Given λ_x, to find an optimal scale factor $\tau_m \in \mathbb{R}^+$ such that $\mathbf{a}_H(m) \approx \tau_m \mathbf{a}_T(m)$, we solve an optimization problem:

$$\tau_m^* = \underset{\tau_m}{\arg\min}\ \|\mathbf{a}_H(m) - \tau_m\mathbf{a}_T(m)\|_2^2, \quad \text{s.t. } \tau_m > 0. \tag{3}$$

The cost function $\|\mathbf{a}_H(m) - \tau_m\mathbf{a}_T(m)\|_2^2 =: J(\tau_m)$, can be express as $J(\tau_m) = \mathbf{a}_H^T(m)\mathbf{a}_H(m) - 2\tau_m\mathbf{a}_H^T(m)\mathbf{a}_T(m) + \tau_m^2\mathbf{a}_T^T(m)\mathbf{a}_T(m)$. Since $\mathbf{a}_H^T(m)\mathbf{a}_H(m) = g_m$ is a known constant variable, and also $\mathbf{a}_T^T(m)\mathbf{a}_T(m) = S_x$, we can rewrite $J(\tau_m)$ as $g_m - 2\tau_m\mathbf{a}_H^T(m)\mathbf{a}_T(m) + \tau_m^2 S_x$. Therefore, the optimal weight τ_m^* can be simply obtained by taking the derivative of $J(\tau_m)$ with respect to τ_m and set to zero. As a result, we have $\tau_m^* = \left(\mathbf{a}_H^T(m)\mathbf{a}_T(m)\right)/S_x$. Since $\mathbf{a}_T(m) = \operatorname{sign}(\mathbf{a}_H(m))$, we have $\tau_m^* = \frac{1}{S_x}\sum |a_H(m)|$.

We denote the space of public storage as $\mathcal{A}^{L\times M}$. For simplicity, through out this paper, we denote by $\mathcal{S} \subset \mathcal{X}^N$ the space of vectors in the signal (original) domain and by $\mathcal{T} \subset \mathcal{A}^L$ the space of vectors in the transform domain. Also, we denote by $\psi(\cdot)$ the operator $T_\lambda(\cdot)$ in general, i.e., $\psi(\mathbf{W}\mathbf{x}) = \boldsymbol{\tau} \odot T_\lambda(\mathbf{W}\mathbf{x})$, where $\odot$ is the Hadamard product.

C. Learning Structured Overcomplete Transform

We construct our overcomplete transform by stacking the C orthonormal sub-transforms as $\mathbf{W} = \left[\begin{array}{ccc} \mathbf{W}_1^T & \cdots & \mathbf{W}_C^T \end{array} \right]^T \in \mathbb{R}^{L \times N}$, with $L = CN$, where $\mathbf{W}_c \in \mathbb{R}^{N \times N}$, $c \in [C]$ are sufficiently different. Our similarity measure between $\mathbf{W}_c$ and $\mathbf{W}_{\acute{c}}$, $c, \acute{c} \in [C]$ is based on the mutual coherence [10] of the $\mathbf{G}_{c\acute{c}} = \left[\begin{array}{cc} \mathbf{W}_c^T & \mathbf{W}_{\acute{c}}^T \end{array} \right] \in \mathbb{R}^{N \times 2N}$, which is defined as:

$$\mu\left(\mathbf{G}_{c\acute{c}}\right) = \max_{1 \leq k,j \leq N, k \neq j} \frac{|\mathbf{g}_{c\acute{c}}^T(k)\mathbf{g}_{c\acute{c}}(j)|}{\|\mathbf{g}_{c\acute{c}}(k)\|_2 . \|\mathbf{g}_{c\acute{c}}(j)\|_2}. \tag{4}$$

Since $\mathbf{W}_c$ and $\mathbf{W}_{\acute{c}}$ are orthonormal matrices, the mutual coherence of this transform satisfies $1/\sqrt{N} \leq \mu\left(\mathbf{G}_{c\acute{c}}\right) \leq 1$ [11]. Note that one can propose various interesting privacy-preserving scenarios by considering different structures, communication schemes, sparsifying rates etc. for transform matrices.

Our sparsifying transform learning is based on the classical Procrustes matrix problem [12]. That is, we seek orthonormal matrices $\mathbf{W}_c \in \mathbb{R}^{N \times N}, c \in [C]$, which most closely transforms a fix matrix $\mathbf{X} \in \mathbb{R}^{N \times M}$ into a sparse matrix $\mathbf{A}_c \in \mathbb{R}^{N \times M}$. Therefore, using the Frobenius norm, the problem is to find $\mathbf{W}_c$ minimizing $\|\mathbf{W}_c\mathbf{X} - \mathbf{A}_c\|_F^2$, subject to $\mathbf{W}_c\mathbf{W}_c^T = \mathbf{I}$. For the square sparsifying transform, it can be shown that for variable $\mathbf{W}_c$ and fixed $\mathbf{X}$, the closed-form solution $\mathbf{W}_c = \mathbf{U}_c\mathbf{V}_c^T$ is given by the singular value decomposition $\mathbf{A}_c\mathbf{X}^T = \mathbf{U}_c\mathbf{\Sigma}_c\mathbf{V}_c^T$. Our algorithm for the above minimization problem alternates between solving for $\mathbf{A}_c = \psi\left(\mathbf{W}_c\mathbf{X}\right)$ (sparse coding step) and $\mathbf{W}_c = \mathbf{U}_c\mathbf{V}_c^T$ (transform update step), whilst the other variables are kept fixed. Moreover, note that we can learn each transform matrix $\mathbf{W}_c$ based on the training signals of class $c \in [C]$. In this case, each block of our over-complete transform will be optimal for its corresponding class.

D. Algorithm

1) Preparation at Owner Side: The owner transforms offline the feature vectors with *trained linear* map $\mathbf{W}$ followed by the *element-wise nonlinearity* map $\psi(\cdot)$. Then, the owner splits each scaled sparse ternary code into a *public* sparse code $\mathbf{a}_1$ and *private (secret)* sparse code $\mathbf{a}_2$, with respective lengths L_p and L_s such that $L = L_p + L_s$. The sign of the sparse components of the public part, i.e., $\mathbf{a}_{\text{clear}}(m) = \text{sign}\left(\mathbf{a}_1(m)\right), \forall m \in [M]$, along with the ambiguized non-sparse code $\mathbf{a}_{\text{amb}}(m) = \mathbf{a}_2(m) \oplus \mathbf{n}, \forall m \in [M]$ are outsourced to the server, where $\mathbf{n} \in \{\pm\tau_m\}, m \in [M]$ and $\oplus$ is orthogonal direct sum. The ambiguization scheme is similar to the proposed method in [1]. It is clear that there is a trade-off between privacy and utility at the server. If we send more components (larger value for L_p) to the server, we increase the privacy leakage, since the server might cluster the public codes in the database. In contrast, a small value for L_p leads to higher privacy.

2) Indexing at Server Side: Since the code is sparse, it can be indexed as in [13], [14].

3) Querying at Client Side: The client transforms the feature vector $\mathbf{y}$ from its query, using the shared *trained linear* map $\mathbf{W}$ followed by the *element-wise nonlinearity* map. Then, the client splits his scaled sparse ternary code into two parts. The sign of the public part then forms the query, which is send to the server. In contrast to the ambiguization part at the owner side, we have no ambiguization for the private part at the client side.

4) Searching at Server Side: Provided that meaningful similarity search is possible between $\mathbf{b}_{\text{clear}}$ and $\mathbf{a}_{\text{clear}}\left(m\right), m \in [M]$, the server seeks all $\{\mathbf{a}_{\text{clear}}\left(m\right), m \in [M]\}$ NNs in the radius $\gamma_1 L_p$ from the query $\mathbf{b}_{\text{clear}}$ in order to produce an initial list $\mathcal{L}_1$ of possible candidates as $\mathcal{L}_1\left(\mathbf{b}_{\text{clear}}\right) = \{m \in [M] : d_{\mathcal{A}_1}\left(\mathbf{a}_{\text{clear}}\left(m\right), \mathbf{b}_{\text{clear}}\right) \leq \gamma_1 L_p\}$, where $d_{\mathcal{A}_1}\left(.,.\right)$ is a similarity measure in space $\mathcal{A}_1$.

Finally, the server sends back the initial list $\mathcal{L}_1$ along with the corresponding calculated similarity measure $d_{\mathcal{A}_1}\left(\mathbf{a}_{\text{clear}}\left(m\right), \mathbf{b}_{\text{clear}}\right)$ (optional side information for the improved search efficiency) as well as retrieved corresponding ambiguized parts $\{\mathbf{a}_{\text{amb}}(m), \forall m \in \mathcal{L}_1\}$. The list size $\text{card}\left(\mathcal{L}_1\right)$ is supposed to be sufficiently large for privacy preservation. The server can either fix the threshold or the number of K similar elements. These parameters are key elements setting the privacy-utility for the clients. A longer initial candidate list results in a higher quality of search at the client side, while a shorter candidate list provides higher client privacy.

5) Refining at Client Side: In [1], the authors show that by imposing ambiguization noise, all distances from the server viewpoint go to a constant value, i.e., all vectors $\mathbf{a}_{\text{amb}}(m), \forall m \in [M]$ seem equally likely from the server standpoint. However, at the client side we can effectively preserve distances up to the desired radius, just by computing distances (or similarity measure) on non-zero components of the sparse representation of the private part $\mathbf{b}_{\text{priv}}$, i.e., on $\text{supp}\left(\mathbf{b}_{\text{priv}}\right)$.

The client receives the initial list $\mathcal{L}_1$, the corresponding distances, and ambiguized vectors $\{\mathbf{a}_{\text{amb}}(m), \forall m \in \mathcal{L}_1\}$. It then computes the distances (or similarity measure) based on the support of its sparse vector, i.e., $d_{\mathcal{A}_2}\left(\mathbf{b}_{\text{priv}}, \cdot\right)$ is defined on the support of $\mathbf{b}_{\text{priv}}$. Next, it produces the final list $\mathcal{L}_2$.

IV. Privacy Performance

We use mutual information between the N-dimensional random feature vector $\mathbf{x}$ and the reconstructed feature $\hat{\mathbf{x}}$, i.e. $I(\mathbf{x}; \hat{\mathbf{x}})^1$, as a metric for privacy leakage. Since the mutual information quantifies the Kullback-Leibler distance between the prior and posterior knowledge of the original data $\mathbf{x}$ and reconstructed data $\hat{\mathbf{x}}$ and is also related to the Fisher information for asymptotically large databases. Indeed, our *privacy leakage* is equivalent to the *invertibility* of our scheme. We know that $I(\mathbf{x}; \hat{\mathbf{x}}) = h(\mathbf{x}) - h(\mathbf{x}|\hat{\mathbf{x}}) = h(\mathbf{x}) - h(\mathbf{x} - \hat{\mathbf{x}}|\hat{\mathbf{x}}) \geq h(\mathbf{x}) - h(\mathbf{x} - \hat{\mathbf{x}})$, where the last inequality is based on the fact that conditioning reduces entropy. We simplified our problem by considering a single-letter formulation and defined separable distortion metrics by averaging the single-letter distortions. Since for a given variance, the normal distribution maximizes entropy, and also assuming that $x - \hat{x}$ has a normal distribution with zero mean and a variance of $\mathbb{E}[(x - \hat{x})^2]$, we have: $h(x) - h(x - \hat{x}) \geq \frac{1}{2}\log_2\left(2\pi e \sigma_x^2\right) - \frac{1}{2}\log_2(2\pi e\, \mathbb{E}[(x - \hat{x})^2])$. Therefore, we have the lower bound $I(x; \hat{x}) \geq \frac{1}{2}\log_2(\frac{\sigma_x^2}{D})$. We denote by $\delta_{\max} = \max_{D \in D_0}\left(\frac{1}{2}\log_2(\frac{\sigma_x^2}{D})\right)$, the maximal privacy leakage on the distortion interval D_0.

[1]All variables and vectors in the entropy and mutual information functions are considered to be random variables and vectors.

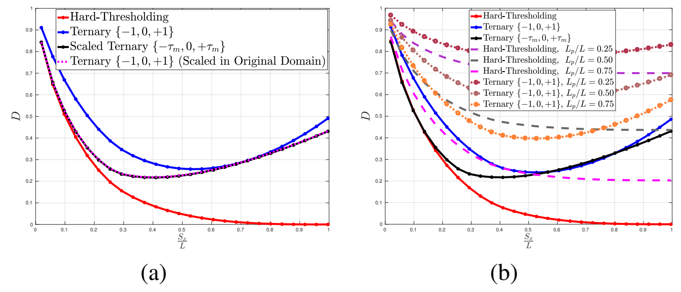

Fig. 2: The relation between distortion and sparsity ratio S_x/L.

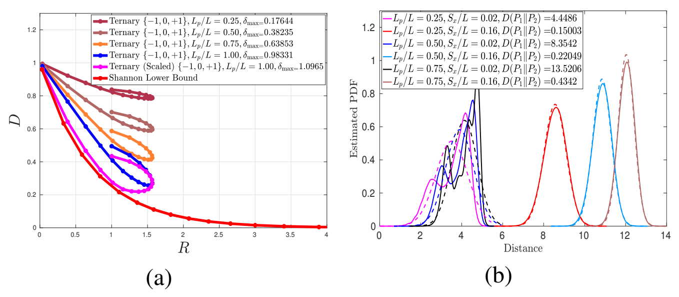

Fig. 3: Effect of public length ratio L_p/L on privacy measures: a) "reconstruction" leakage, b) "clustering" leakage.

Here, we propose a *scaled reconstruction scheme*, which achieves the same performance as SSTC, while reducing the computational cost and storage space effectively. To this end, we make a balance between the (raw) feature vector $\mathbf{x}(m)$ and reconstructed feature vector $\widehat{\mathbf{x}}(m)$, which is obtained from the ternary sparse code $\mathbf{a}_T(m)$. Given λ_x, to find an optimal global scale factor θ such that $\mathbf{X} \approx \theta\widehat{\mathbf{X}}$, we solve an optimization problem:

$$\theta^* = \arg\min_{\theta} \quad \|\mathbf{X} - \theta\widehat{\mathbf{X}}\|_F^2, \quad \text{s.t. } \theta > 0. \tag{5}$$

The cost function $\|\mathbf{X} - \theta\widehat{\mathbf{X}}\|_F^2 =: J(\theta)$, can be expressed as $J(\theta) = \text{tr}\left[\mathbf{X}^T\mathbf{X} - 2\theta\mathbf{X}^T\widehat{\mathbf{X}} + \theta^2\widehat{\mathbf{X}}^T\widehat{\mathbf{X}}\right]$. Since $\mathbf{X}^T\mathbf{X}$ and $\widehat{\mathbf{X}}^T\widehat{\mathbf{X}}$ are known constant variables, the optimal scale factor θ^* can be simply obtained by taking the derivative of $J(\theta)$ with respect to θ and setting to zero. As result, we have $\theta^* = \text{tr}\left[\mathbf{X}^T\widehat{\mathbf{X}}\right]/\text{tr}\left[\widehat{\mathbf{X}}^T\widehat{\mathbf{X}}\right]$. This means that, instead of storing the SSTC of the private part of the database at the server, we just need to store the STC of them. The owner can obtain the global scale factor θ based on the training data and then shared it with authenticated clients. The performance of the scaled reconstruction scheme is depicted in Fig. 2a. This scheme leads us to the more interesting scenario for large scale identification problems using distributed servers, which is not the scope of this paper. Fig. 2 depicts the amount of reconstruction distortion as a function of the sparsity ratio $\alpha_x = S_x/L$ for the condition that $x_n \sim \mathcal{N}(0,1)$, $L = N$. In Fig. 2a, we compare the distortion measure for four different cases. As shown, both the STC and SSTC have a global minimum, which are obtained at around $\alpha_x = 0.53$ and $\alpha_x = 0.41$, respectively. As we expected, the SSTC outperforms STC in accuracy. However, it requires more storage space and computational cost.

In Fig. 2b, we illustrate the reconstruction distortion for the cases in which the owner sends the fraction L_p of his L-dimensional ternarized projected data with the alphabet $\{-1,0,+1\}$ or $\{-\tau_m,0,+\tau_m\}$ to the server, as the public clear database. Also, we compare the distortion measure with the hard-thresholding case. Therefore, the curious server just capable to reconstruct the $\mathbf{x}(m), \forall m \in [M]$ from $\mathbf{a}_{\text{clear}}(m), \forall m \in [M]$ with effectively high distortion level, even if it knows the sparsifying transform $\mathbf{W}$.

The bit rate of our encoding scheme can be formulated as $R = \frac{1}{L}\log_2\left(\binom{L}{S_x}2^{S_x}\right)$. In Fig. 3a, we depict and compare the distortion-rate behaviour of the ternary encoding scheme for various ratios of L_p/L along with the maximal privacy leakage $\delta_{\max}$. As it is shown, the maximal achievable rate is 1.585 which is equal to the entropy of the ternary alphabet. The recursive part of the distortion-rate curve corresponds to the increasing behaviour of distortion after a certain sparsity ratio S_x/L. This behavior is studied with more details in [15].

Moreover, the curious server might want to cluster the database vectors from $\mathbf{a}_{\text{clear}}(m), \forall m \in [M]$. In [1], the authors introduced a Kullback-Leibler divergence privacy protection measure, in order to address this clustering threat. Utilizing the same measure, we study the privacy protection of our outsourced media search scheme for the case in which the owner sends the fraction L_p of its own L-dimensional ternarized projected data with the alphabet $\{-1,0,+1\}$ to the server. To this end, we generate four 500-dimensional i.i.d. vectors with distribution $\mathcal{N}(\mathbf{0},\mathbf{1})$ and 1000 500-dimensional i.i.d. vectors with distribution $\mathcal{N}(\mathbf{0},\mathbf{0.1})$. Then we add each 250 (out of 1000) low variance vectors to the four high variance ones (schematically shown in [1, Fig. 6]). This results in a database of 1000 vectors adhering to four clusters. We denote the probability density functions (PDFs) of 'intra-cluster' and 'inter-cluster' of distances by P_{intra} and P_{inter}, respectively. To this end, we define distribution P_1 as mixture of P_{intra} and P_{inter}, that is, $P_1 = \alpha_x P_{\text{intra}} + (1-\alpha_x) P_{\text{inter}}$, $0 \le \alpha_x \le 1$. Also, we denote the Gaussian distribution $\mathcal{N}\left(\mu_2, \sigma_2^2\right)$ by P_2, such that μ_2 and σ_2^2 are the mean and variance of P_1, respectively. Therefore, the privacy protection measure of disclosing the structure of database by the curious server can be defined by the Kullback-Leibler divergence (KLD) as $D(P_1\|P_2) = \mathbb{E}_{P_1}\left[\log\frac{P_1}{P_2}\right]$. Now, the privacy protection constraint can be expressed as $D(P_1\|P_2) \le \epsilon$, where ϵ determines the allowable privacy leakage from database clustering. In Fig. 3b, we illustrate the estimated PDFs of pair-wise distances in the transform domain. The solid lines indicate P_1 with three public-length ratios $L_p/L = 0.25, 0.50, 0.75$ and two sparsity ratios $S_x/L = 0.02$, 0.16. As evident, by increasing the sparsity ratio, the distribution P_1 becomes uni-modal Gaussian, therefore, the curious server cannot cluster the database. The dashed lines indicate the corresponding Gaussian distribution P_2 fit to each solid plot. It is clear that the larger L_p requires larger S_x/L in order to satisfy the specific privacy protection constraint ϵ. Therefore, given the desired privacy protection constraint ϵ, the curious serer cannot cluster the public database, provided $D(P_1\|P_2) \le \epsilon$.

V. Conclusion

We have proposed a novel privacy preserving framework for outsourced media search applications based on sparse ternary coding with partial ambiguization. Our intuition for the similarity search in the public in-the-clear database is based on the fact that the sparse significant components can effectively provide the initial list, while impeding the reconstruction and clustering at the server. One of the main points illustrated by this study is that the owner can compute only one global scale factor based on the training data and then share it with the authorized clients to scale their reconstructed media in the original domain. The results show that the curious server cannot reconstruct and cluster the samples in the database, provided the mutual information privacy leakage $I(x;\hat{x})$ and Kullback-Leibler privacy protection $D(P_1\|P_2)$ constraints are satisfied.

Acknowledgment

This research has been supported by the ERA-Net project ID_IoT No 20CH21_167534.

References

[1] B. Razeghi, S. Voloshynovskiy, D. Kostadinov, and O. Taran, “Privacy preserving identification using sparse approximation with ambiguization,” in *IEEE International Workshop on Information Forensics and Security (WIFS)*, Rennes, France, December 2017, pp. 1–6.

[2] M. Diephuis, S. Voloshynovskiy, O. Koval, and F. Beekhof, “Robust message-privacy preserving image copy detection for cloud-based systems,” in *CBMI 2012,10th Workshop on Content-Based Multimedia Indexing*, 2012.

[3] H. Jégou, M. Douze, and C. Schmid, “On the burstiness of visual elements,” in *IEEE Conf. on Comp. Vision and Pattern Recog. (CVPR)*, 2009, pp. 1169–1176.

[4] F. Perronnin and C. Dance, “Fisher kernels on visual vocabularies for image categorization,” in *IEEE Conf. on Comp. Vision and Pattern Recog. (CVPR)*, 2007, pp. 1–8.

[5] H. Jégou, M. Douze, C. Schmid, and P. Pérez, “Aggregating local descriptors into a compact image representation,” in *IEEE Conf. on Comp. Vision and Pattern Recog. (CVPR)*, 2010, pp. 3304–3311.

[6] A. Babenko, A. Slesarev, A. Chigorin, and V. Lempitsky, “Neural codes for image retrieval,” in *Europ. Conf. on Comp.Vision*, 2014, pp.584–599.

[7] D. P. Kingma and M. Welling, “Auto-encoding variational bayes,” in *International Conference on Learning Representations (ICLR)*, 2014.

[8] L. Weng, L. Amsaleg, and T. Furon, “Privacy-preserving outsourced media search,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 10, pp. 2738–2751, 2016.

[9] S. Ravishankar and Y. Bresler, “Learning sparsifying transforms,” *IEEE Trans. on Signal Processing*, vol. 61, no. 5, pp. 1072–1086, 2013.

[10] A. M. Bruckstein, D. L. Donoho, and M. Elad, “From sparse solutions of systems of equations to sparse modeling of signals and images,” *SIAM review*, vol. 51, no. 1, pp. 34–81, 2009.

[11] D. L. Donoho and X. Huo, “Uncertainty principles and ideal atomic decomposition,” *IEEE Transactions on Information Theory*, vol. 47, no. 7, pp. 2845–2862, 2001.

[12] P. H. Schönemann, “A generalized solution of the orthogonal procrustes problem,” *Psychometrika*, vol. 31, no. 1, pp. 1–10, 1966.

[13] S. Ferdowsi, S. Voloshynovskiy, D. Kostadinov, and T. Holotyak, “Fast content identification in high-dimensional feature spaces using sparse ternary codes,” in *IEEE Int. Work. on Inf. Forensics and Security (WIFS)*, 2016, pp. 1–6.

[14] ——, “Sparse ternary codes for similarity search have higher coding gain than dense binary codes,” in *IEEE Int. Symp. on Inf. Theory (ISIT), 2017*.

[15] S. Ferdowsi, S. Voloshynovskiy, and D. Kostadinov, “A multi-layer network based on sparse ternary codes for universal vector compression,” *ArXiv e-prints*, Oct 2017.