

Boise State University

ScholarWorks

---

Computer Science Faculty Publications and  
Presentations

Department of Computer Science

---

2017

## CAPIA: Cloud Assisted Privacy-Preserving Image Annotation

Yifan Tian

*Embry-Riddle Aeronautical University*

Yantian Hou

*Boise State University*

Jiawei Yuan

*Embry-Riddle Aeronautical University*

# CAPIA: Cloud Assisted Privacy-preserving Image Annotation

Yifan Tian

Department of ECSSE  
Embry-Riddle Aeronautical University  
Daytona Beach, FL  
Email: tianyl@my.erau.edu

Yantian Hou

Department of CS  
Boise State University  
Boise, ID  
Email: yantianhou@boisestate.edu

Jiawei Yuan

Department of ECSSE  
Embry-Riddle Aeronautical University  
Daytona Beach, FL  
Email: yuanj@erau.edu

**Abstract**—Using public cloud for image storage has become a prevalent trend with the rapidly increasing number of pictures generated by various devices. For example, today’s most smartphones and tablets synchronize photo albums with cloud storage platforms. However, as many images contain sensitive information, such as personal identities and financial data, it is concerning to upload images to cloud storage. To eliminate such privacy concerns in cloud storage while keeping decent data management and search features, a spectrum of keywords-based searchable encryption (SE) schemes have been proposed in the past decade. Unfortunately, there is a fundamental gap remains open for their support of images, i.e., appropriate keywords need to be extracted for images before applying SE schemes to them. On one hand, it is obviously impractical for smartphone users to manually annotate their images. On the other hand, although cloud storage services now offer image annotation services, they rely on access to users’ unencrypted images. To fulfill this gap and open the first path from SE schemes to images, this paper proposes a cloud assisted privacy-preserving automatic image annotation scheme, namely CAPIA. CAPIA enables cloud storage users to automatically assign keywords to their images by leveraging the power of cloud computing. Meanwhile, CAPIA prevents the cloud from learning the content of images and their keywords. Thorough analysis is carried out to demonstrate the security of CAPIA. A prototype implementation over the well-known IAPR TC-12 dataset further validates the efficiency and accuracy of CAPIA.

## I. INTRODUCTION

The widespread use of smartphones brings the explosive growth in the number of pictures taken. According to a recent report from Mylio [1], over one trillion pictures will be taken by smartphones in 2017. To store and manage a large number of images in an efficient and economical way, public cloud storage has been widely adopted by most smartphone users. Currently, majority of smartphones are synchronizing their photo albums with cloud storage services, including Apple’s iCloud, Samsung Cloud, Google Photo, etc. Despite the decent features offered by cloud storage services, directly uploading images to public cloud also causes privacy breaches and even legal issues. This is because many images contain sensitive information, such as healthcare information, personal identities/locations, and financial information, etc. Nowadays, encrypting images with standard encryption algorithms, such as AES, is still the major approach for privacy protection in

cloud storage [2], [3]. Nevertheless, these traditional encryption algorithms inevitably sacrifice the usability of images outsourced to cloud, including efficient indexing, search, keywords extractions, etc.

To protect the privacy of data stored on public cloud while retaining the efficient search ability over these data, keywords-based searchable encryption (SE) has received a lot of research effort in recent years [4]–[9]. A SE scheme typically provides encrypted search indexes constructed based on proper keywords assigned to data files. With these encrypted indexes, the data owner can submit encrypted keywords-based-search request to search their data over ciphertexts. SE schemes have been demonstrated to be effective for text files, however, fundamental challenges remain for their support of images. Specifically, unlike text files that support automatic keyword extraction from their content or file names, keywords assignment for an image relies on manual description, or automatic annotation based on a large-scale pre-annotated image dataset. From the perspective of user experience, manually annotating each image from users’ devices is clearly an impractical choice. Meanwhile, automatic image annotation involving large-scale image datasets is too resource consuming to be developed on mobile terminals. Although cloud storage services, such as Apple’s iCloud and Google Photo, are now offering automatic image annotation to extract appropriate keywords for images, access to unencrypted images is a necessary requirement. As a matter of fact, directly utilizing this kind of annotation services contradicts the privacy protection purpose in SE. Therefore, the key gap to fulfill between SE schemes and images becomes how to automatically annotate images in a privacy-preserving manner. Unfortunately, to the best of our knowledge, there is no existing solution that can efficiently and effectively achieve such a desired functionality.

In this paper, we propose the first privacy-preserving image annotation scheme (CAPIA) that can be securely delegated to the public cloud. CAPIA provides efficient annotation and can be easily parallelized for cloud computing environment. Meanwhile, CAPIA achieves comparable annotation accuracy compared with existing no-privacy-preserving image annotation approaches. To securely support required operations in automatic image annotation, we first design two privacy-preserving outsourcing schemes for  $L_1$  distance comparison

and Kullback-Leibler (KL) Divergence comparison respectively as building blocks for CAPIA. Furthermore, as the same keyword may have different importance for the semantic description of different images, we design a real-time weight to support accurate final keywords selection in the image annotation process. To evaluate CAPIA, we first conduct thorough analysis for it in terms of security and efficiency. Then, we provide an extensive experimental evaluation for CAPIA with a prototype implementation over the well-known IAPR TC-12 dataset [10]. Our evaluation results further validate the practical performance of CAPIA in terms of efficiency and accuracy.

The rest of this paper is organized as follows: In Section II: we present the system model and threat model of CAPIA. Section III introduces backgrounds of automatic image annotation and technical preliminaries for CAPIA. The detailed construction of CAPIA as well as its security analysis are provided in Section IV. Section V evaluates the performance of CAPIA. We review and discuss related works in Section VI and conclude this paper in Section VII.

## II. MODELS

### A. System Model

As shown in Fig.1, our CAPIA system is consistent with today's major cloud storage application, which consists of two entities: a *Cloud Server* and a *User*. The user stores his/her images on cloud, and the cloud helps the user to annotate his/her images. Once the user obtains keywords for his/her images, he/she can adopt existing SE schemes for his/her images to achieve keywords-based privacy-preserving image search. During the entire image storage and annotation process, CAPIA prevents the cloud storage server from learning the privacy of the user's images and their corresponding keywords.

In CAPIA, the user first performs a one-time preparation for extracting features of images in a pre-annotated dataset, and encrypts these features as well as keywords. These encrypted features and keywords will be outsourced to cloud to assist future privacy-preserving image annotation. For resource constrained mobile devices, this one-time setup process can be performed using desktops. Later on, when the user has a new image to annotate, he/she generates an encrypted request and sends it to the cloud. After processing the request, the cloud returns ciphertexts of top related keywords and auxiliary information. Finally, the user decrypts all keywords and ranks them based on their real-time weights to select final keywords.

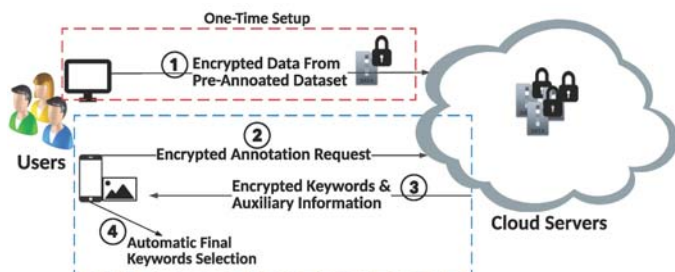


Fig. 1. System Model of CAPIA

### B. Threat Model

In CAPIA, we consider the cloud server to be “curious-but-honest”, i.e., the cloud server will follow our scheme to perform storage and annotation services correctly, but it may try to learn sensitive information in users' data. The cloud server has access to all encrypted images, encrypted image features, encrypted keywords, the user's request, and encrypted annotation results. This assumption is consistent with majority of existing works that focus on search over encrypted data on public cloud [6]–[9]. In addition, the cloud may get some ciphertexts-plaintexts pairs of the user's images as well as their corresponding features and keywords. For example, the cloud may obtain a user's image information by analyzing his/her social media posts. CAPIA should protect a user's privacy by preventing the cloud from learning the following information: 1) content of the user's images; 2) features extracted and keywords annotated for each image; 3) request linkability, i.e., tell whether multiple annotation requests are from the same image.

## III. PRELIMINARIES

### A. Image Feature Extraction

In this paper, we adopt global low-level image features as that are utilized in the baseline image annotation technique [11], because it can be applied to general images without complex models and subsequent training. Color features of an image are extracted in three different color spaces: RGB, HSV, and LAB. In particular, RGB feature is computed as a normalized 3D histogram of RGB pixel, in which each channel (R,G,B) has 16 bins that divide the color space values from 0 to 255. The HSV and LAB features can be processed similarly as RGB, and thus we can construct three feature vectors for RGB, HSV and LAB respectively as  $\mathbf{V}_{RGB}$ ,  $\mathbf{V}_{HSV}$ , and  $\mathbf{V}_{LAB}$ . Texture features of an image are extracted using Gabor and Haar wavelets. Specifically, an image is first filtered with Gabor wavelets at three scales and four orientations, resulting in twelve response images. Each response image is then divided into non-overlapping rectangle blocks. Finally, mean filter response magnitudes from each block over all response images are concatenated into a feature vector, denoted as  $\mathbf{V}_G$ . Meanwhile, a quantized Gabor feature of an image is generated using the mean Gabor response phase angle in non-overlapping blocks in each response image. These quantized values are concatenated into a feature vector, denoted as  $\mathbf{V}_{GQ}$ . The Haar feature of an image is extracted similarly as Gabor, but based on differently configured Haar wavelets. HaarQ stands for the quantized version of Haar feature, which quantizes Haar features into  $[0, -1, 1]$  if the sign of Haar response values are [zero, negative, positive], respectively. We denote feature vectors of Haar and HaarQ as  $\mathbf{V}_H$  and  $\mathbf{V}_{HQ}$  respectively. Therefore, given an image, seven feature vectors will be extracted as  $[\mathbf{V}_{RGB}, \mathbf{V}_{HSV}, \mathbf{V}_{LAB}, \mathbf{V}_G, \mathbf{V}_{GQ}, \mathbf{V}_H, \mathbf{V}_{HQ}]$ . For more details about the adopted image feature extraction, please refer to ref [11].

## B. Integer Vector Encryption (IVE)

In this section, we describe a homomorphic encryption scheme designed for integer vectors [12], which will be tailored in our construction to achieve privacy-preserving image annotation. For expression simplicity, following definitions will be used in the rest of this paper

- For a vector  $\mathbf{V}$  (or a matrix  $\mathbf{M}$ ), define  $|\max(\mathbf{V})|$  (or  $|\max(\mathbf{M})|$ ) to be the maximum absolute value of their elements.
- For  $a \in \mathbb{R}$ , define  $\lceil a \rceil$  to be the nearest integer of  $a$ ,  $\lceil a \rceil_q$  to be the nearest integer of  $a$  with modulus  $q$ .
- For matrix  $\mathbf{M} \in \mathbb{R}^{n \times m}$ , define  $\text{vec}(\mathbf{M})$  to be a  $nm$ -dimensional vector by concatenating the transpose of each column of  $\mathbf{M}$ .

**Encryption:** Given a  $m$ -dimensional vector  $\mathbf{V} \in \mathbb{Z}_p^m$  and the secret key matrix  $\mathbf{S} \in \mathbb{Z}_q^{m \times m}$ , output the ciphertext of  $\mathbf{V}$  as

$$\mathbf{C} = \mathbf{S}^{-1}(w\mathbf{V} + \mathbf{e})^T \quad (1)$$

where  $\mathbf{S}^{-1}$  is the inverse matrix of  $\mathbf{S}$ ,  $T$  is the transpose operator,  $\mathbf{e}$  is a random error vector,  $w$  is an integer parameter,  $q \gg p$ ,  $w > 2|\max(\mathbf{e})|$ .

**Decryption:** Given the ciphertext  $\mathbf{C}$ , it can be decrypted using  $\mathbf{S}$  and  $w$  as  $\mathbf{V} = \lceil \frac{(\mathbf{S}\mathbf{C})^T}{w} \rceil_q$ .

**Inner Product:** Given two ciphertexts  $\mathbf{C}_1, \mathbf{C}_2$  of  $\mathbf{V}_1, \mathbf{V}_2$ , and their corresponding secret keys  $\mathbf{S}_1$  and  $\mathbf{S}_2$ , the inner product operation of  $\mathbf{V}_1$  and  $\mathbf{V}_2$  over ciphertexts can be performed as

$$\text{vec}(\mathbf{S}_1^T \mathbf{S}_2) \lceil \frac{\text{vec}(\mathbf{C}_1 \mathbf{C}_2^T)}{w} \rceil_q = w\mathbf{V}_1 \mathbf{V}_2^T + \mathbf{e} \quad (2)$$

To this end,  $\text{vec}(\mathbf{S}_1^T \mathbf{S}_2)$  becomes the new secret key and  $\lceil \frac{\text{vec}(\mathbf{C}_1 \mathbf{C}_2^T)}{w} \rceil_q$  becomes the new ciphertext of  $\mathbf{V}_1 \mathbf{V}_2^T$ .

More details about this IVE encryption algorithm and its security proof are available in ref [12].

## IV. CLOUD ASSISTED PRIVACY-PRESERVING IMAGE ANNOTATION

The core idea of automatic image annotation is built on the hypothesis that images contain similar objects are likely to share keywords. Hence, given a large-scale pre-annotated image dataset, the annotation process for a new image can be first treated as a process of transferring keywords from images that contains similar objects. In CAPIA, similarity of images is measured by seven low-level color and texture feature vectors  $[\mathbf{V}_{i,RGB}, \mathbf{V}_{i,HSV}, \mathbf{V}_{i,LAB}, \mathbf{V}_{i,G}, \mathbf{V}_{i,GQ}, \mathbf{V}_{i,H}, \mathbf{V}_{i,HQ}]$  as discussed in Section III-A. Specifically, given two images  $I_a, I_b$ , their similarity can be computed as a combined distance

$$\begin{aligned} Dis_{ab} = & DL1_{ab}^{RGB} + DL1_{ab}^{HSV} + DL1_{ab}^G + DL1_{ab}^{GQ} \\ & + DL1_{ab}^H + DL1_{ab}^{HQ} + DKL_{ab}^{LAB} \end{aligned}$$

where  $DL1$  and  $DKL$  denote  $L_1$  distance and KL-Divergence of two vectors after data normalization. We consider these seven basic distances contribute equally to the total combined distance  $Dis_{ab}$ . Based on this observation, we first propose two privacy-preserving distance comparison solutions for  $L_1$  (namely, PLIC) and KL-Divergence (namely, PKLC) respectively, which support two key operations in CAPIA.

## A. PLIC: Privacy-preserving $L_1$ Distance Comparison

In PLIC, we consider a user has three  $m$ -dimensional integer vectors  $\mathbf{V}_i, i \in \{a, b, c\}$  that will be outsourced to cloud after encryption. The cloud later compares  $L_1$  distances  $DL1_{ac}$  and  $DL1_{bc}$  directly over ciphertexts to figure out which one is smaller.

**Data Preparation:** Given a vector  $\mathbf{V}_i = [v_{i1}, \dots, v_{im}], i \in \{a, b, c\}$ , the user converts it to a  $m\beta$ -dimensional binary vector  $\tilde{\mathbf{V}}_i = [F(v_{i1}), \dots, F(v_{im})]$ , where  $\beta = |\max(\mathbf{V}_i)|$ , and  $F(v_{ij}) = [1, 1, \dots, 1, 0, \dots, 0]$  such that the first  $v_{ij}$  terms are 1 and the rest  $\beta - v_{ij}$  terms are 0. The  $L_1$  distance between  $\mathbf{V}_a$  and  $\mathbf{V}_b$  now can be calculated as

$$DL1_{ab} = \sum_{j=1}^m |v_{aj} - v_{bj}| = \sum_{j=1}^{m\beta} (\tilde{v}_{aj} - \tilde{v}_{bj})^2$$

Then, the user adopts an approximation method introduced in ref [13] to reduce the dimension of  $\tilde{\mathbf{V}}_i$  from  $m\beta$  to  $\hat{m} = \alpha m \log_{\gamma}^{\beta+1}$  based on the Johnson Lindenstrauss (JL) Lemma [14]. By denoting the approximated vector as  $\hat{\mathbf{V}}_i$ , we have  $DL1_{ab} = \sum_{j=1}^{m\beta} (\tilde{v}_{aj} - \tilde{v}_{bj})^2 \approx \sum_{j=1}^{\hat{m}} (\hat{v}_{aj} - \hat{v}_{bj})^2$ .

The correctness and accuracy of such an approximation have been proved in ref [13]. According to our experimental evaluation in Section V, we set  $\alpha = 1$  and  $\gamma = 100$  in our scheme to achieve balanced accuracy and efficiency.

**Data Encryption:** Given an approximated vector  $\hat{\mathbf{V}}_i, i \in \{a, b\}$ , the user appends two elements to it as  $\hat{\mathbf{V}}_i = [\hat{v}_{i1}, \hat{v}_{i2}, \dots, \hat{v}_{i\hat{m}}, r - \frac{1}{2} \sum_{j=1}^{\hat{m}} \hat{v}_{ij}^2, \epsilon_i]$ , where  $r$  is a random number and  $\epsilon_i$  is a small random noise. Then, the user encrypts  $\hat{\mathbf{V}}_i$  using the **Encryption** algorithm of IVE presented in Section III-B as

$$\mathbf{C}_i = \mathbf{S}^{-1}(w\hat{\mathbf{V}}_i + \mathbf{e}_i)^T \quad (3)$$

where  $\mathbf{S}$  is the secret matrix,  $\mathbf{e}_i$  is an error vector, and  $w$  is an integer parameter.  $\mathbf{C}_a, \mathbf{C}_b$ , and  $w$  are outsourced to the cloud.

**Request Generation:** Given the approximated vector  $\hat{\mathbf{V}}_c$ , the user selects a positive random number  $r_c$  and applies it to  $\hat{\mathbf{V}}_c$  as  $\hat{\mathbf{V}}_c = [r_c \hat{v}_{c1}, \dots, r_c \hat{v}_{c\hat{m}}, r_c, 1]$ .  $\hat{\mathbf{V}}_c$  is then encrypted as  $\mathbf{C}_c = \mathbf{S}^{-1}(w\hat{\mathbf{V}}_c + \mathbf{e}_c)^T$ , where  $\mathbf{S}_c$  is the secret key generated for  $\hat{\mathbf{V}}_c$ .  $\mathbf{C}_c$  and  $\mathbf{S}^T \mathbf{S}_c$  are sent to the cloud as request.

**Distance Comparison:** On receiving the request, the cloud computes  $\lceil \frac{\text{vec}(\mathbf{C}_a \mathbf{C}_c^T)}{w} \rceil_q$ ,  $\lceil \frac{\text{vec}(\mathbf{C}_b \mathbf{C}_c^T)}{w} \rceil_q$ , and decrypts them using  $\text{vec}(\mathbf{S}^T \mathbf{S}_c)$  to obtain  $\hat{\mathbf{V}}_a \hat{\mathbf{V}}_c^T$  and  $\hat{\mathbf{V}}_b \hat{\mathbf{V}}_c^T$  as Eq.2. Finally, the approximated  $L_1$  distance comparison is performed as

$$\begin{aligned} & \hat{\mathbf{V}}_b \hat{\mathbf{V}}_c^T - \hat{\mathbf{V}}_a \hat{\mathbf{V}}_c^T \quad (4) \\ & = r_c \sum_{j=1}^{\hat{m}} \hat{v}_{bj} \hat{v}_{cj} - \frac{r_c}{2} \sum_{j=1}^{\hat{m}} \hat{v}_{bj}^2 + r_c r + \epsilon_b \\ & \quad - (r_c \sum_{j=1}^{\hat{m}} \hat{v}_{aj} \hat{v}_{cj} - \frac{r_c}{2} \sum_{j=1}^{\hat{m}} \hat{v}_{aj}^2 + r_c r + \epsilon_a) \\ & = \frac{r_c}{2} \left( \sum_{j=1}^{\hat{m}} (\hat{v}_{aj} - \hat{v}_{bj})^2 - \sum_{j=1}^{\hat{m}} (\hat{v}_{bj} - \hat{v}_{cj})^2 \right) + (\epsilon_b - \epsilon_a) \\ & \approx \frac{r_c}{2} (DL1_{ac} - DL1_{bc}) + (\epsilon_b - \epsilon_a) \end{aligned}$$

It is worth to note that PL1C is only interested in which distance is smaller during the comparison. Therefore, instead of letting the cloud get exact  $L_1$  distances for comparison, PL1C adopts approximated distance comparison result scaled and obfuscated by  $r_c$  and  $\epsilon_b - \epsilon_a$  as shown in Eq.4. As  $r_c$  is a positive random number, the sign of  $\frac{r_c}{2}(DL1_{ac} - DL1_{bc})$  is consistent with  $DL1_{ac} - DL1_{bc}$ . Meanwhile, since  $r_c \gg \epsilon_b - \epsilon_a$ , the added noise term has negligible influence to the sign of  $DL1_{ac} - DL1_{bc}$  unless these two distances are very close to each other. Fortunately, instead of finding the most related one, our CAPIA design will utilize PL1C to figure out top 10 related candidates during the comparison. Such a design makes important candidates (say top 5 out of top 10) not be bypassed by the error introduced in  $\epsilon_b - \epsilon_a$ . This hypothesis is further validated by our experimental results in Section V.

### B. PKLC: Privacy-preserving KL-Divergence Comparison

In PKLC, we consider a user has three  $m$ -dimensional vectors  $\mathbf{V}_i, i \in \{a, b, c\}$ , and wants to outsource the comparison of  $DKL_{ac}$  and  $DKL_{bc}$  to the cloud without disclosing the content of  $\mathbf{V}_i, i \in \{a, b, c\}$ . The definition of KL-Divergence for two vectors  $\mathbf{V}_a, \mathbf{V}_b$  is:

$$\begin{aligned} DKL_{ab} &= \sum_{j=1}^m v_{aj} \times \log\left(\frac{v_{aj}}{v_{bj}}\right) \\ &= \sum_{j=1}^m v_{aj} \times \log(v_{aj}) - \sum_{j=1}^m v_{aj} \times \log(v_{bj}) \end{aligned} \quad (5)$$

where  $\log\left(\frac{v_{aj}}{v_{bj}}\right) = \log(v_{aj}) = \log(v_{bj}) = 0$  if  $v_{aj} = 0$  or  $v_{bj} = 0$ .

**Data Encryption:** The user first appends  $m + 2$  elements to  $\mathbf{V}_i, i \in \{a, b\}$  as  $\mathbf{V}_i = [v_{i1}, v_{i2}, \dots, v_{im}, v_{i1} \times \log(v_{i1}), \dots, v_{im} \times \log(v_{im}), r, \epsilon_i]$ , where  $r$  is a random number and  $\epsilon_i$  is a small random noise. Then,  $\mathbf{V}_i, i \in \{a, b\}$  are encrypted with the **Encryption** algorithm of IVE as

$$\mathbf{C}_i = \mathbf{S}^{-1}(w\mathbf{V}_i + \mathbf{e}_i)^T \quad (6)$$

$\mathbf{C}_a$  and  $\mathbf{C}_b$  are outsourced to the cloud.

**Request Generation:** The user processes  $\mathbf{V}_c$  to generate a privacy-preserving KL-Divergence comparison request as

- Replace elements  $v_{cj}$  with  $-r_c \times \log(v_{cj})$ , and append  $m+2$  elements to  $\mathbf{V}_c$  as  $\mathbf{V}_c = [-r_c \times \log(v_{c1}), \dots, -r_c \times \log(v_{cm}), G(v_{c1}), \dots, G(v_{cm}), r_c, -1]$ , where  $G(v_{cj}) = \begin{cases} r_c, v_{cj} \neq 0 \\ 0, v_{cj} = 0 \end{cases}$ ,  $r_c$  is a positive random number changing for every request.
- Encrypt  $\mathbf{V}_c$  as  $\mathbf{C}_c$  using the **Encryption** algorithm of IVE as  $\mathbf{C}_c = \mathbf{S}_c^{-1}(w\mathbf{V}_c + \mathbf{e}_c)^T$ .

$\mathbf{C}_c$  and  $\mathbf{S}^T \mathbf{S}_c$  are sent to the cloud as request.

**Distance Comparison:** On receiving the request, the cloud first computes  $\left\lceil \frac{vec(\mathbf{C}_a \mathbf{C}_c^T)}{w} \right\rceil_q$ ,  $\left\lceil \frac{vec(\mathbf{C}_b \mathbf{C}_c^T)}{w} \right\rceil_q$  and decrypts them

using  $vec(\mathbf{S}^T \mathbf{S}_c)$  to get  $\mathbf{V}_a \mathbf{V}_c^T$  and  $\mathbf{V}_b \mathbf{V}_c^T$  as Eq.2. Then, the cloud compares  $DKL_{ac}$  and  $DKL_{bc}$  by computing

$$\begin{aligned} &\mathbf{V}_a \mathbf{V}_c^T - \mathbf{V}_b \mathbf{V}_c^T \\ &= r_c \left( r + \sum_{j=1}^m v_{aj} \times \log(v_{aj}) - \sum_{j=1}^m v_{aj} \times \log(v_{cj}) \right) - \epsilon_a \\ &- r_c \left( r + \sum_{j=1}^m v_{bj} \times \log(v_{bj}) - \sum_{j=1}^m v_{bj} \times \log(v_{cj}) \right) + \epsilon_b \\ &= r_c (DKL_{ac} - DKL_{bc}) + (\epsilon_b - \epsilon_a) \end{aligned} \quad (7)$$

Similar to our PL1C construction, we have  $r_c > 0$  and  $r_c \gg (\epsilon_b - \epsilon_a)$ . Therefore, the cloud can figure out which KL-Divergence is smaller based on the scaled and obfuscated comparison result.

### C. Detailed Construction of CAPIA

CAPIA consists of five major procedures. In the *System Setup*, the user selects system parameters, extracts and pre-processes feature vectors of images in a pre-annotated dataset. Then, the user executes the *Data Encryption* procedure to encrypt these processed feature vectors. Both the *System Setup* procedure and the *Data Encryption* procedure are one-time cost in CAPIA. Later on, the user can use the *Secure Annotation Request* procedure to generate an encrypted annotation request. On receiving the request, the cloud server performs the *Privacy-preserving Annotation on Cloud* procedure to return encrypted keywords for the requested image. At the end, the user obtains final keywords by executing the *Final Keyword Selection* procedure.

1) *System Setup:* To perform the one-time setup of CAPIA system, the user first prepares a pre-annotated image dataset with  $n$  images, which can be obtained from public sources, such as IAPR TC-12 [10], LabelMe [15], etc. For each image  $I_i$  in the dataset, the user extracts seven feature vectors  $[\mathbf{V}_{i,RGB}, \mathbf{V}_{i,HSV}, \mathbf{V}_{i,LAB}, \mathbf{V}_{i,G}, \mathbf{V}_{i,GQ}, \mathbf{V}_{i,H}, \mathbf{V}_{i,HQ}]$ . Compared with other five feature vectors that have dimension up to 256,  $\mathbf{V}_{i,H}$  and  $\mathbf{V}_{i,HQ}$  have a high dimension as 4096. To guarantee the efficiency while processing feature vectors, Principal Component Analysis (PCA) [16] is utilized to reduce the dimension of  $\mathbf{V}_{i,H}$  and  $\mathbf{V}_{i,HQ}$ . According to our experimental evaluation in Section V-B, PCA based dimension reduction with proper setting can significantly improve the efficiency of CAPIA with slight accuracy loss. After that,  $L_1$  normalization will be performed for each feature vector, which normalizes elements in these vectors to  $[-1, 1]$ . Besides  $\mathbf{V}_{i,LAB}$ , the user also increases each element in  $\mathbf{V}_{i,k}, k \in \{RGB, HSV, G, GQ, H, HQ\}$  as  $v_{i,k,j} = v_{i,k,j} + 1$  to avoid negative values. Next, each element in all feature vectors are scaled by the same value. Given three processed vectors  $\mathbf{V}_i, i \in \{a, b, c\}$ , it is easy to verify that the sign of  $L_1$  distance comparison result  $DL1_{ab} - DL1_{ac}$  and KL-Divergence comparison result  $DKL_{ab} - DKL_{ac}$  with processed vectors remain the same as that using original vectors. Six feature vectors that use  $L_1$  distance for similarity measurement are concatenated as a  $m_{L1}$ -dimensional vector  $\mathbf{V}_{i,L1}$ .  $\mathbf{V}_{i,LAB}$  is

denoted as a  $m_{KL}$ -dimensional vector  $\mathbf{V}_{i,KL}$  for expression simplicity. It is easy to verify that  $DL1_{ab}^{L1} = DL1_{ab}^{RGB} + DL1_{ab}^{HSV} + DL1_{ab}^G + DL1_{ab}^{GQ} + DL1_{ab}^H + DL1_{ab}^{HQ}$ .

2) *Dataset Encryption*: Given an image  $I_i$  in the pre-annotated dataset, its keywords  $\{K_{i,t}\}$  are first encrypted using AES. Then, feature vectors  $\mathbf{V}_{i,L1}$  and  $\mathbf{V}_{i,KL}$  are encrypted as  $\mathbf{C}_{i,L1}$  and  $\mathbf{C}_{i,KL}$  using the *Data Encryption* methods in our proposed PLIC and PKLC schemes respectively. During the encryption, same secret keys  $\mathbf{S}_{L1}$ ,  $\mathbf{S}_{KL}$ , public parameter  $w$ , and random number  $r$  will be used for all images. However, different error vector  $\mathbf{e}_i$  and noise term  $\epsilon_i$  are generated for each image  $I_i$  correspondingly. The user also computes  $\mathbf{S}_{L1}^T \mathbf{S}_{s,L1}$  and  $\mathbf{S}_{KL}^T \mathbf{S}_{s,KL}$ , in which  $\mathbf{S}_{s,L1}$  and  $\mathbf{S}_{s,KL}$  are secret keys for the encryption of later annotation requests. These  $\mathbf{C}_{i,L1}$ ,  $\mathbf{C}_{i,KL}$  and encrypted keywords of each image  $I_i$ , as well as  $\mathbf{S}_{L1}^T \mathbf{S}_{s,L1}$  and  $\mathbf{S}_{KL}^T \mathbf{S}_{s,KL}$  are outsourced to the cloud.

3) *Secure Annotation Request*: When the user has a new image  $I_s$  for annotation, he/she first extracts seven feature vectors as  $\mathbf{V}_s, s \in [RGB, HSV, LAB, G, GQ, H, HQ]$ . These vectors will be normalized and scaled to output  $\mathbf{V}_{s,L1}$  and  $\mathbf{V}_{s,KL}$  as that in the *System Setup* procedure. Then, the user processes and encrypts  $\mathbf{V}_{s,L1}$  and  $\mathbf{V}_{s,KL}$  as  $\mathbf{C}_{s,L1}$  and  $\mathbf{C}_{s,KL}$  using the *Request Generation* methods in our PLIC and PKLC schemes respectively. For each annotation request, the user generates a new positive random number  $r_s$  and a new error vector  $\mathbf{e}_s$ .  $\mathbf{C}_{s,L1}$  and  $\mathbf{C}_{s,KL}$  are sent to the cloud as the annotation request.

4) *Privacy-preserving Annotation on Cloud*: On receiving the request, the cloud first outputs  $\mathbf{V}_{i,L1} \mathbf{V}_{s,L1}^T$  and  $\mathbf{V}_{i,KL} \mathbf{V}_{s,KL}^T$  for each image in the pre-annotated dataset as

$$\mathbf{V}_{i,L1} \mathbf{V}_{s,L1}^T = \text{vec}(\mathbf{S}_{L1}^T \mathbf{S}_{s,L1}) \left[ \frac{\text{vec}(\mathbf{C}_{i,L1} \mathbf{C}_{s,L1}^T)}{w} \right]_q \quad (8)$$

$$\mathbf{V}_{i,KL} \mathbf{V}_{s,KL}^T = \text{vec}(\mathbf{S}_{KL}^T \mathbf{S}_{s,KL}) \left[ \frac{\text{vec}(\mathbf{C}_{i,KL} \mathbf{C}_{s,KL}^T)}{w} \right]_q \quad (9)$$

where  $1 \leq i \leq n$ . Then, the cloud ranks all the images according to their combined distances to the request image  $I_s$ . Specifically, a distance comparison candidate  $Comp_i = -2(\mathbf{V}_{i,L1} \mathbf{V}_{s,L1}^T) + \mathbf{V}_{i,KL} \mathbf{V}_{s,KL}^T$  can be generated for each image  $I_i$ . Given  $I_a$  and  $I_b$  for example, the cloud can rank them as

$$\begin{aligned} & Comp_a - Comp_b \\ &= 2(\mathbf{V}_{b,L1} \mathbf{V}_{s,L1}^T - \mathbf{V}_{a,L1} \mathbf{V}_{s,L1}^T) \\ &+ \mathbf{V}_{a,KL} \mathbf{V}_{s,KL}^T - \mathbf{V}_{b,KL} \mathbf{V}_{s,KL}^T \\ &= r_s(DL1_{as}^{L1} - DL1_{bs}^{L1}) + 2(\epsilon_b - \epsilon_a) \\ &+ r_s(DKL_{as}^{LAB} - DKL_{bs}^{LAB}) + (\epsilon_b - \epsilon_a) \\ &= r_s(Dis_{as} - Dis_{bs}) + 3(\epsilon_b - \epsilon_a) \end{aligned} \quad (10)$$

As  $r_s$  is a positive value and  $r_s \gg (\epsilon_b - \epsilon_a)$ , the cloud can figure out which image is more relative to  $I_s$  according to the above distance comparison result. According to the ranking of all pre-annotated images, the cloud outputs top related images to  $I_s$  and denotes them as a set  $RST$ . Finally, the cloud returns

distance comparison candidates  $Comp_i, i \in RST$  as well as corresponding encrypted keywords back to the user.

5) *Final Keyword Selection*: In this stage, the user first decrypts encrypted keywords and obtains  $K_{i,t}, i \in RST$ , where  $K_{i,t}$  is the  $t$ -th pre-annotated keyword in image  $I_i$ . Then, the user computes distances  $Dis_{is}, i \in RST$  as

$$\begin{aligned} Dis_{is} &= (2r + \sum_{j=1}^{m_{L1}} v_{s,L1,j}^2) + \frac{Comp_i}{r_s} \\ &= (2r + \sum_{j=1}^{m_{L1}} v_{s,L1,j}^2) + \frac{-2(\mathbf{V}_{i,L1} \mathbf{V}_{s,L1}^T) + \mathbf{V}_{i,KL} \mathbf{V}_{s,KL}^T}{r_s} \end{aligned}$$

To achieve higher accuracy in keywords selection, we consider that keywords in images that have smaller distance to the requested one are more relevant. Thus, we define a real-time weight  $W_i$  for each keyword based on distances  $Dis_{is}$  as

$$W_i = 1 - \frac{Dis_{is}}{\sum_{i \in RST} Dis_{is}} \quad (11)$$

$$W_t = \sum W_i, \text{ for } I_i \text{ contains } K_{i,t} \quad (12)$$

Specifically, we first figure out the weight  $W_i$  of each image according to their distance based similarity. As our definition in Eq.11, images with smaller distance will receive a larger weight value. Then, considering the same keyword can appear in multiple images, the final weight  $W_t$  of a keyword  $K_{i,t}$  is generated by adding weights of images that contain this keyword. Finally, the user selects keywords for his/her image according to their ranking of weight  $W_i$ .

#### D. Security Analysis

In CAPIA, we have the following privacy related data: feature vectors  $\{\mathbf{V}_{i,L1}, \mathbf{V}_{i,KL}\}_{1 \leq i \leq n}$  and keywords of image  $I_i$  in the pre-annotated dataset; feature vectors  $\mathbf{V}_{s,L1}, \mathbf{V}_{s,KL}$  of the image requested for annotation. As keywords are encrypted using standard AES encryption, we consider them secure against the cloud server as well as outside adversaries. With regards to  $\mathbf{V}_{i,L1}, \mathbf{V}_{i,KL}, \mathbf{V}_{s,L1}, \mathbf{V}_{s,KL}$ , they are encrypted using the encryption scheme of IVE [12] after pre-processing as presented in our PLIC and PKLC schemes. The IVE scheme [12] has been proved to be secure based on the well-known Learning with Errors (LWE) hard problem [17]. Thus, given the ciphertexts  $\mathbf{C}_{i,L1}, \mathbf{C}_{i,KL}, \mathbf{C}_{s,L1}, \mathbf{C}_{s,KL}$  only, it is computational infeasible for the cloud server or outside adversaries to recover  $\mathbf{V}_{i,L1}, \mathbf{V}_{i,KL}, \mathbf{V}_{s,L1}, \mathbf{V}_{s,KL}$ .

1) *Security of Outsourcing  $\mathbf{S}_{L1}^T \mathbf{S}_{s,L1}$  and  $\mathbf{S}_{KL}^T \mathbf{S}_{s,KL}$* : As  $\mathbf{S}_{L1}^T \mathbf{S}_{s,L1}$  and  $\mathbf{S}_{KL}^T \mathbf{S}_{s,KL}$  are used in the same manner, we use  $\mathbf{S}^T \mathbf{S}_s$  to denote them for expression simplicity. Different from the original Encryption algorithm of IVE, the user in CAPIA also outsources  $\mathbf{S}^T \mathbf{S}_s$  to the cloud besides ciphertexts  $\mathbf{C}_{i,L1}, \mathbf{C}_{i,KL}, \mathbf{C}_{s,L1}, \mathbf{C}_{s,KL}$ . As all elements in  $\mathbf{S}$  and  $\mathbf{S}_s$  are randomly selected, elements in their multiplication  $\mathbf{S}^T \mathbf{S}_s$  have the same distribution as these elements in  $\mathbf{S}$  and  $\mathbf{S}_s$  [18]. Thus, given  $\mathbf{S}^T \mathbf{S}_s$ , the cloud server is not be able to extract  $\mathbf{S}$  or  $\mathbf{S}_s$  directly and use them to decrypt  $\mathbf{C}_{i,L1}, \mathbf{C}_{i,KL}, \mathbf{C}_{s,L1}, \mathbf{C}_{s,KL}$ .

By combining  $\mathbf{S}^T \mathbf{S}_s$  with ciphertexts  $\mathbf{C}_{i,L1}$  and  $\mathbf{C}_{s,L1}$  (same as that for  $\mathbf{C}_{i,KL}$  and  $\mathbf{C}_{s,KL}$ ), the cloud can obtain

$$\begin{aligned} \mathbf{S}^T \mathbf{S}_s \mathbf{C}_{i,L1} &= \mathbf{S}^T \mathbf{S}_s \mathbf{S}_s^{-1} (w \mathbf{V}_{i,L1} + \mathbf{e}_i)^T \\ \mathbf{S}^T \mathbf{S}_s \mathbf{C}_{s,L1} &= \mathbf{S}^T \mathbf{S}_s \mathbf{S}_s^{-1} (w \mathbf{V}_{s,L1} + \mathbf{e}_i)^T = \mathbf{S}^T (w \mathbf{V}_{s,L1} + \mathbf{e}_i)^T \end{aligned}$$

From the above two equations, it is clear that the combination of  $\mathbf{S}^T \mathbf{S}_s$ ,  $\mathbf{C}_{i,L1}$  and  $\mathbf{S}^T \mathbf{S}_s$ ,  $\mathbf{C}_{s,L1}$  only transfer them to the ciphertexts of  $\mathbf{V}_{i,L1}$  and  $\mathbf{V}_{s,L1}$  that encrypted using the IVE scheme with new keys  $\mathbf{S}^T \mathbf{S}_s \mathbf{S}_s^{-1}$  and  $\mathbf{S}^T$  respectively. As  $\mathbf{S}^T \mathbf{S}_s \mathbf{S}_s^{-1}$  and  $\mathbf{S}^T$  are random keys and unknown to the cloud, recovering  $\mathbf{V}_{i,L1}$ ,  $\mathbf{V}_{s,L1}$  from  $\mathbf{S}^T \mathbf{S}_s \mathbf{C}_{i,L1}$ ,  $\mathbf{S}^T \mathbf{S}_s \mathbf{C}_{s,L1}$  still become the *LWE* problem as proved in ref [12]. To this end,  $\mathbf{S}^T \mathbf{S}_s$  only helps the cloud to perform distance comparison in CAPIA, but does not bring advantages to recover feature vectors compared with the given ciphertexts only scenario.

2) *Known Ciphertext-Image Pairs*: We now consider that the cloud server gets a set of ciphertext-image pairs from the background analysis as  $\{\mathbf{V}_{i,L1}, \mathbf{C}_{i,L1}\}$  ( $\{\mathbf{V}_{s,L1}, \mathbf{C}_{s,L1}\}$ ,  $\{\mathbf{V}_{i,KL}, \mathbf{C}_{i,KL}\}$ ,  $\{\mathbf{V}_{s,KL}, \mathbf{C}_{s,KL}\}$  respectively). In ref [19], a linear analysis attack based on ciphertext-image pairs is introduced to recover vectors from their distance comparison result. In particular, instead of trying to recovering feature vectors or secret keys directly from ciphertexts, such an attack attempts to recover the vectors from the distance comparison result by constructing and solving enough number (i.e., greater than the dimension of vector) of linear equations. To launch this kind of linear analysis attack to CAPIA, there are two necessary requirements that need to be fulfilled simultaneously: 1) The cloud obtains at least  $m$  ciphertext-image pairs, where  $m$  is the dimension of feature vectors; 2) The cloud has access to the exact  $L_1$  distance and KL-Divergence comparison results. As shown in Eq.4 and Eq.7, CAPIA only provides scaled and obfuscated comparison results by adding noise terms  $\epsilon_i$  and random scaling factor  $r_c$ . As a result, the cloud cannot fulfill the second requirement to launch a successful linear analysis attack to CAPIA. To this end, CAPIA is secure even a set of ciphertext-image pairs are obtained by the cloud server.

3) *Request Unlinkability*: The request unlinkability in CAPIA is guaranteed by the randomization for each request. Specifically, each query request  $\mathbf{V}_{s,L1}$ ,  $\mathbf{V}_{s,KL}$  is element-wise obfuscated with different random error terms  $\mathbf{e}_s$  and random number  $r_s$  during the encryption, which makes the obfuscated  $\mathbf{V}_{s,L1}$ ,  $\mathbf{V}_{s,KL}$  have the same distribution as these random values in  $\mathbf{e}_s$  and  $r_c$  [18]. Thus, by changing  $\mathbf{e}_s$  and  $r_c$  during the encryption of different requests, CAPIA outputs different random ciphertexts, even for requests generated from the same image.

## V. EVALUATION

To evaluate the performance of CAPIA, we implemented a prototype using Python 2.7. In our implementation, Numpy [20] is used to support efficient multi-dimension array operations. OpenCV [21] is used to extract the color-space features of the images and build the filter kernels to generate the Gabor filter results. Pywt [22] is adopted to perform Haar wavelet and get the corresponding Haar results. Sklearn [23] is used

to perform the PCA transformation. We use the well-known IAPR TC-12 [24] as the pre-annotated dataset, which contains 20,000 annotated images and the average number of keywords for each image is 5.7. All tests are performed on a 3.1 GHz Intel Core i7 Macbook Pro with OS X 10.11.6 installed.

In the rest of this section,  $n$  is the total number of images in the pre-annotated dataset;  $m_{L1}$  and  $m_{KL}$  are dimensions of vectors  $\mathbf{V}_{i,L1}$  and  $\mathbf{V}_{i,KL}$  after pre-processing respectively;  $PCA - X$  is used to denote the strength of *PCA* transformation applied to  $\mathbf{V}_{i,H}$  and  $\mathbf{V}_{i,HQ}$  in  $\mathbf{V}_{i,L1}$ , which compresses their dimensions from 4096 to  $\frac{4096}{X}$ .  $PCA - 128$ ,  $PCA - 64$ ,  $PCA - 32$ ,  $PCA - 16$ , and  $PCA - 8$  are evaluated in our experiments to balance the efficiency and accuracy of CAPIA. We also use  $\mathit{DOT}_m$  to denote a dot product operation between to two  $m$ -dimensional vectors.

### A. System Setup and Dataset Encryption

To perform the one-time setup in CAPIA, the user pre-processes feature vectors of each image in the pre-annotated image dataset. Specifically, the user first performs JL-Lemma based approximation over  $\mathbf{V}_{i,L1}$  to make  $\mathbf{V}_{i,L1}$  compatible with our PLIC. As discussed in Section IV-A, there is a trade-off between the approximation accuracy of  $L_1$  distance and length of the approximated vector that determines efficiency of follow up privacy-preserving operations. To balance such a trade-off, we evaluate different parameters for approximation as shown in Fig.2 (a)-(d). According to our results, we suggest to set  $\alpha = 1$  and  $\gamma = 100$  which introduces 3.61% error rate for  $L_1$  distance computation, and extends the dimension of  $\mathbf{V}_{i,L1}$  from 864 to 1296 under the setting of  $PCA - 32$ . The selection of *PCA* strength will be discussed and evaluated in Section V-B. Specifically, the error rate drops fast when  $\alpha < 1$  and becomes relative stable when  $\alpha > 1$ . Meanwhile, the dimension of the approximated vector increases linearly to the value of  $\alpha$ . With regard to  $\gamma$ , the dimension of the approximated vector becomes relative stable when  $\gamma > 100$ , however, the error rate still increases when  $\gamma > 100$ . As shown in Fig.3 (a), such an approximation setting makes the pre-processing procedure cost 1471ms to 118ms for each image with *PCA* setting from *No PCA* to  $PCA - 128$ .

After the pre-processing,  $\{\mathbf{V}_{i,L1}, \mathbf{V}_{i,KL}\}_{1 \leq i \leq n}$  will be encrypted using the *Data Encryption* procedures of our PLIC and PKLC schemes respectively. As shown in Eq.3 and Eq.6, the encryption of each  $\mathbf{V}_{i,L1}$  and  $\mathbf{V}_{i,KL}$  requires  $(m_{L1})\mathit{DOT}_{m_{L1}}$  and  $(m_{KL})\mathit{DOT}_{m_{KL}}$  operations respectively. Fig.3 (a) shows the total encryption cost for  $\mathbf{V}_{i,L1}$  and  $\mathbf{V}_{i,KL}$  of a pre-annotated image decreases from 1436ms to 4.7ms by increasing the strength of *PCA* from *No PCA* to  $PCA - 128$ . This is because the dimension of  $\mathbf{V}_{i,L1}$ , i.e.,  $m_{L1}$ , is determined by the strength of *PCA*, which is directly correlated to the encryption cost of  $\mathbf{V}_{i,L1}$ . Same as the system setup, encrypting feature vectors is also a one-time cost, which does not impact the performance of later on real-time privacy-preserving image annotation.

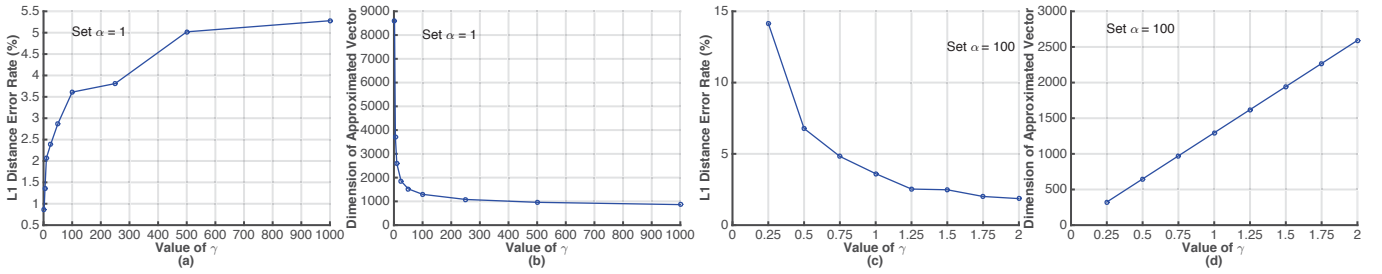


Fig. 2. Error rate of Approximation and Dimension of Approximated Vector (*PCA* – 32)

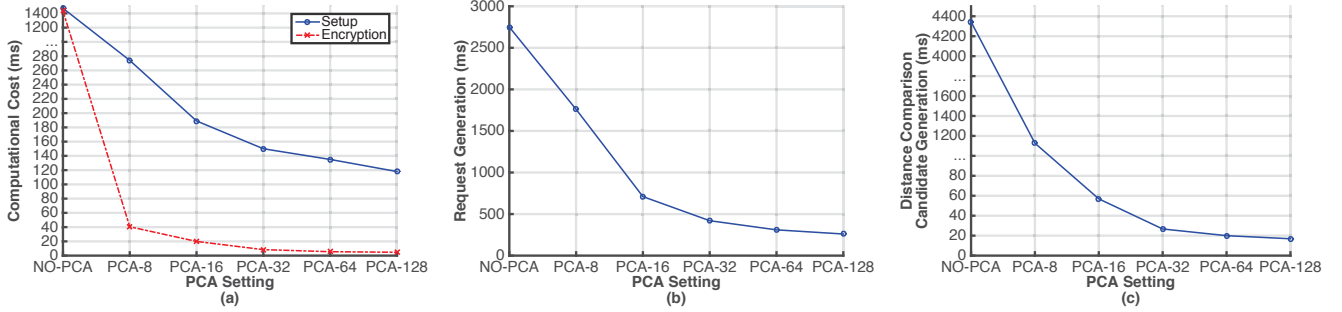


Fig. 3. (a) System Setup and Encryption Cost (b) Request Generation Cost (c) Distance Comparison Candidate Generation Cost on Cloud

### B. Real-time Image Annotation

**Efficiency:** To annotate a new image in a privacy-preserving manner, the user generates an encrypted request by pre-processing and encrypting feature vectors of the requested image. By varying the PCA strength from *No PCA* to *PCA* – 128, Fig.3 (b) shows that the request generation spends from 2775ms to 268ms. On receiving the encrypted request, the cloud first computes distance comparison candidate  $Comp_i$  for each image  $I_i, 1 \leq i \leq n$  in the pre-annotated dataset, which requires a  $(m_{L1} + 1)DOT_{m_{L1}}$  operation and a  $(m_{KL} + 1)DOT_{m_{KL}}$  operation as shown in Eq.8 and Eq.9. By changing the strength of PCA from *No PCA* to *PCA* – 128, the computational cost for  $Comp_i$  changes from 4334ms to 16.9ms as shown in Fig.3 (c). This is because the dimension of  $V_{i,L1}$ , i.e.,  $m_{L1}$ , is determined by the strength of PCA and  $m_{L1} \gg m_{KL}$  (e.g., 1298 v.s. 98 in *PCA* – 32). Afterwards, the cloud selects encrypted keywords according the ranking of  $Comp_i$  as Eq.10. It is worth to note that the annotation process on cloud can be easily parallelized for performance optimization. In particular, computation of  $Comp_i$  for different pre-annotated images are independent with each other, and thus can be easily parallelized in the cloud computing environment.

**Accuracy:** We now evaluate the accuracy of CAPIA. In our evaluation, we use the standard average *precision* and *recall* rates to measure the accuracy of keywords annotation as that in automatic annotation using plaintext images. We use 50 images as annotation requests, and each image will be assigned ten keywords after automatic annotation. Each request has two or more related images in the pre-annotated dataset. We use set  $[K_1, K_2, \dots, K_x]$  to denote distinct keywords annotated for all 50 requested images. The annotation precision and recall rate for a keyword  $K_j, 1 \leq j \leq x$  in these 50 requests are

defined as

- $precision_{K_j}$ : number of images assigned  $K_j$  correctly in CAPIA divided by the total number of images assigned  $K_j$  in CAPIA.
- $recall_{K_j}$ : number of images assigned  $K_j$  correctly in CAPIA divided by the number of images assigned  $K_j$  in the ground-truth annotation.

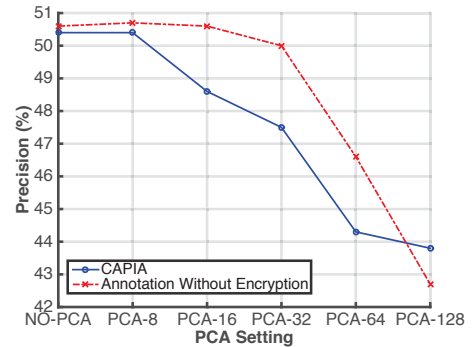


Fig. 4. Precision of CAPIA and Annotation without Encryption

To compare the annotation accuracy of CAPIA, we also evaluate the no-privacy-preserving annotation using the same 50 requests. As shown in Fig.4 and Fig.5, while providing strong privacy guarantee, CAPIA introduces less than 2.5% and 7.5% accuracy loss in terms of average precision and recall rates with PCA setting from *No PCA* to *PCA* – 128. In addition, Fig.4 and Fig.5 also demonstrate that the increasing of PCA strength reduces the annotation accuracy of CAPIA to some extent, especially from *PCA* – 32 to *PCA* – 64. Taking the efficiency enhancement brought by PCA together into consideration, we suggest to use *PCA* – 32 as an appropriate setting for practical usage. Specifically, Fig.3 demonstrates the efficiency improvement from PCA becomes relative stable



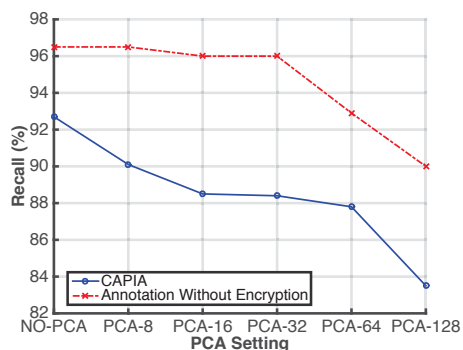


Fig. 5. Recall of CAPIA and Annotation without Encryption

after  $PCA - 32$ . Meanwhile, the accuracy loss of CAPIA still increases quickly after  $PCA - 32$ .

In Table I, we present samples of automatically annotated images using CAPIA. On one hand, CAPIA is highly possible to assign correct keywords to images compared with human annotation. This observation also confirms the high average recall rate of CAPIA, since these ground-truth annotations are likely to be covered in CAPIA. On the other hand, CAPIA also introduces additional keywords that frequently appear together with these accurate keywords in top related images. These additional keywords are typically not directly included in human annotations, but are potentially related to correct keywords. Such a fact also explains why the average precision rate of CAPIA is relatively low compared with the average recall rate. Overall, our evaluation results demonstrate that although CAPIA cannot provide perfect keywords selection all the time compared with human annotation, it is still promising for automatically assigning keywords to images, and hence fulfilling the fundamental gap between SE schemes and images.





| Image   | CAPIA Annotation  | Human Annotation                               |
|---|---|--|
|  | floor-tennis-court, man, woman  | floor-tennis-court, man                        |
|  | sky-blue, highway, vegetation, ground, bush, trees, lake, ocean                           | highway, sky-blue, trees, vegetation           |
|  | cloud, sky-blue, ground, mountain, horse, man, road, grass                                | ground, cloud, sky-blue, mountain, snow, grass |
|  | group-of-persons, sky-blue, ground, trees, mountain, ruin-archeological, hat, cloud, hill | trees, ground, man, sky-blue, group-of-persons |

TABLE I  
SAMPLE ANNOTATION RESULTS

### C. Communication Cost and Storage Overhead

The communication cost in CAPIA comes from two major parts: annotation request and encrypted results returned from the cloud server. The encrypted request consists of a  $m_{L1}$ -dimensional vector  $C_{s,L1}$  and a  $m_{KL}$ -dimensional vector  $C_{s,KL}$ . In the  $PCA - 32$  setting, the total communication

cost for a request is 26KB. Meanwhile, the returned result contains encrypted keywords and distance comparison candidates  $Comp_i$  of top 10 related images. Using AES-256 for keywords encryption, the total size for the returned result is 488 Bytes with the average number of keywords for each pre-annotated image as 5.7. With regard to the storage overhead of CAPIA, it includes two parts for each pre-annotated image  $I_i$ : 1) encrypted feature vectors  $C_{i,L1}$  and  $C_{i,KL}$ , which are 26KB in total. 2) Encrypted keywords, which are 480 Bytes as average using AES-256 encryption.

## VI. RELATED WORKS

To solve the problem of how to search over encrypted data, the idea of keywords-based searchable encryption (SE) was first introduced by Song et.al in ref [4]. Later on, with the widespread use of cloud storage services, the idea of SE received increasing attention from researchers. In ref [5], [6], search efficiency enhanced SE schemes are proposed based on novel index constructions. After that, SE with the support of multiple keywords and conjunctive keywords are investigated in ref [7], [8], and thus making the search more accurate and flexible. Recently, fuzzy keyword is considered in ref [9], which enables SE schemes to tolerate misspelled keyword during the search process. While these SE schemes offer decent features for keywords-based search, their application to images are limited given the question that how keywords of images can be efficiently extracted with privacy protection. It is impractical for cloud storage users to manually annotate their images. To automate the keywords extraction process for images, a number of research works have been proposed with the concept of “automatic image annotation” [11], [25]–[27]. In automatic image annotation, keywords of a new image can be learned from a large-scale images that have already been annotated. Nowadays, mobile devices have become the major platform for taking and outsourcing images, however, deploying automatic image annotation with large-scale image datasets on mobile devices is clearly inefficient in terms of energy, storage, and computation. Although outsourcing image annotation tasks to public cloud servers is a potential solution to release the burden of resource constrained mobile devices, it also raises privacy issues since unencrypted images need to be delegated to the cloud. Therefore, this paper proposes CAPIA, which utilizes the power of cloud computing to perform automatic image annotation for users, while only providing encrypted image information to the cloud.

Another line of research that is related to this work is privacy-preserving image retrieval [28]–[31]. While these schemes also investigate similarity measurement between images, none of them considers how to transfer keywords to new images. In addition, these existing privacy-preserving image retrieval schemes [28]–[30] are designed based on powerful but expensive homomorphic encryption schemes, which require frequent user (or a fully trusted key agent) involvement during the image similarity measurement process. Differently, CAPIA enables the user to fully outsource the privacy-preserving image similarity measurement task to the cloud without any

interaction. In ref [31], the performance of privacy-preserving image similarity measurement has been greatly enhanced on both user side and cloud server side. Unfortunately, the security of this scheme against the linear analysis attack [19] is based on the assumption that PCA transformation parameters cannot be learned by the cloud server. In CAPIA, such an assumption is not necessary thanks to our design based on the LWE hard problem and our approximated distance comparison.

## VII. CONCLUSION

In this work, we proposed a cloud assisted privacy-preserving automatic image annotation scheme (CAPIA), which supports efficient and accurate keywords extraction. In CAPIA, lightweight privacy-preserving  $L_1$  distance (PLIC) and KL-Divergence (PKLC) comparison schemes are carefully designed, which enable key steps of automatic image annotation to be performed in a privacy-preserving manner. Our PLIC and PKLC schemes can also be utilized as independent tools for other related fields, especially for similarity measurement of data. To improve the annotation accuracy, we also investigate a real-time weight design and integrate it into CAPIA. Thorough security analysis is provided to demonstrate that CAPIA is secure in the defined threat model. Our prototype implementation over the well-known IAPR TC-12 dataset validates the practical performance of CAPIA in terms of computational cost, communication cost, storage cost, and accuracy.

## REFERENCES

- [1] M. LLC., "How many digital photos will be taken in 2017," <http://mylio.com/true-stories/tech-today/how-many-digital-photos-will-be-taken-2017-repost>, 2016.
- [2] Boxcryptor, "Encrypt your files in your dropbox," <https://www.boxcryptor.com/en/dropbox>, [Online; accessed Aug. 2016].
- [3] Amazon Simple Storage Service, "Protecting data using encryption," <http://docs.aws.amazon.com/AmazonS3/latest/dev/UsingEncryption.html>, [Online; accessed Aug. 2016].
- [4] D. X. Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," in *Proceedings of the 2000 IEEE Symposium on Security and Privacy*, ser. SP '00. Washington, DC, USA: IEEE Computer Society, 2000, pp. 44–55.
- [5] D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano, "Public key encryption with keyword search," in *Advances in Cryptology - EUROCRYPT 2004*. Springer Berlin Heidelberg, 2004, pp. 506–522.
- [6] C. Wang, N. Cao, J. Li, K. Ren, and W. Lou, "Secure ranked keyword search over encrypted cloud data," in *Proceedings of the 2010 IEEE 30th International Conference on Distributed Computing Systems*, ser. ICDCS '10. Washington, DC, USA: IEEE Computer Society, 2010, pp. 253–262.
- [7] N. Cao, C. Wang, M. Li, K. Ren, and W. Lou, "Privacy-preserving multi-keyword ranked search over encrypted cloud data," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 25, no. 1, pp. 222–233, Jan 2014.
- [8] W. Sun, B. Wang, N. Cao, M. Li, W. Lou, Y. T. Hou, and H. Li, "Privacy-preserving multi-keyword text search in the cloud supporting similarity-based ranking," in *Proceedings of the 8th ACM SIGSAC Symposium on Information, Computer and Communications Security*, ser. ASIA CCS '13. New York, NY, USA: ACM, 2013, pp. 71–82.
- [9] B. Wang, S. Yu, W. Lou, and Y. T. Hou, "Privacy-preserving multi-keyword fuzzy search over encrypted data in the cloud," in *INFOCOM, 2014 Proceedings IEEE*, April 2014.
- [10] H. J. Escalante, C. A. Hernández, J. A. Gonzalez, A. López-López, M. Montes, E. F. Morales, L. Enrique Sucar, L. Villaseñor, and M. Grubinger, "The segmented and annotated iapr tc-12 benchmark," *Comput. Vis. Image Underst.*, vol. 114, no. 4, pp. 419–428, Apr. 2010. [Online]. Available: <http://dx.doi.org/10.1016/j.cviu.2009.03.008>
- [11] A. Makadia, V. Pavlovic, and S. Kumar, "Baselines for image annotation," *Int. J. Comput. Vision*, vol. 90, no. 1, pp. 88–105, Oct. 2010. [Online]. Available: <http://dx.doi.org/10.1007/s11263-010-0338-6>
- [12] H. Zhou and G. Wornell, "Efficient homomorphic encryption on integer vectors and its applications," in *Information Theory and Applications Workshop (ITA), 2014*. IEEE, 2014, pp. 1–9.
- [13] S. Rane, W. Sun, and A. Vetro, "Privacy-preserving approximation of l1 distance for multimedia applications," in *2010 IEEE International Conference on Multimedia and Expo*, July 2010, pp. 492–497.
- [14] B. J. William and L. Joram, "Extensions of lipschitz mappings into a hilbert space," *Contemporary mathematics*, vol. 26, pp. 189–206, 1984.
- [15] M. C. Science and A. I. Laboratory, "Labelme dataset," <http://labelme.csail.mit.edu/Release3.0/browserTools/php/dataset.php>, 2017.
- [16] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006.
- [17] Z. Brakerski, C. Gentry, and S. Halevi, "Packed ciphertexts in lwe-based homomorphic encryption," in *16th International Conference on Practice and Theory in Public-Key Cryptography (PKC)*, February 2013, pp. 1–13.
- [18] J. Katz and Y. Lindell, *Chapter 11, Introduction to Modern Cryptography*. Chapman & Hall/CRC, 2007.
- [19] B. Yao, F. Li, and X. Xiao, "Secure nearest neighbor revisited," in *Data Engineering (ICDE), 2013 IEEE 29th International Conference on*, April 2013, pp. 733–744.
- [20] N. Developers, "Numpy," *NumPy Numpy. Scipy Developers*, 2013.
- [21] G. Bradski et al., "The opencv library," *Doctor Dobbs Journal*, vol. 25, no. 11, pp. 120–126, 2000.
- [22] F. Wasilewski, "PyWavelets - Wavelet Transforms in Python," <https://pywavelets.readthedocs.io/en/latest/>, 2006, [Online; accessed 07-March-2017].
- [23] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.
- [24] G.-H. Liu and J.-Y. Yang, "Content-based image retrieval using color difference histogram," *Pattern Recognition*, vol. 46, no. 1, pp. 188–198, 2013.
- [25] X.-J. Wang, L. Zhang, F. Jing, and W.-Y. Ma, "Annosearch: Image auto-annotation by search," in *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, vol. 2. IEEE, 2006, pp. 1483–1490.
- [26] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman, "Labelme: a database and web-based tool for image annotation," *International journal of computer vision*, vol. 77, no. 1-3, pp. 157–173, 2008.
- [27] Y. Verma and C. V. Jawahar, "Image annotation using metric learning in semantic neighbourhoods," in *Proceedings of the 12th European Conference on Computer Vision - Volume Part III*, ser. ECCV'12. Berlin, Heidelberg: Springer-Verlag, 2012, pp. 836–849. [Online]. Available: [http://dx.doi.org/10.1007/978-3-642-33712-3\\_60](http://dx.doi.org/10.1007/978-3-642-33712-3_60)
- [28] W. Lu, A. Varna, and M. Wu, "Confidentiality-preserving image search: A comparative study between homomorphic encryption and distance-preserving randomization," *Access, IEEE*, vol. 2, pp. 125–141, 2014.
- [29] C.-Y. Hsu, C.-S. Lu, and S.-c. Pei, "Image feature extraction in encrypted domain with privacy-preserving sift," *Image Processing, IEEE Transactions on*, vol. 21, no. 11, pp. 4593–4607, Nov 2012.
- [30] L. Zhang, T. Jung, P. Feng, K. Liu, X. Y. Li, and Y. Liu, "Pic: Enable large-scale privacy preserving content-based image search on cloud," in *2015 44th International Conference on Parallel Processing*, Sept 2015, pp. 949–958.
- [31] J. Yuan, S. Yu, and L. Guo, "Seisa: Secure and efficient encrypted image search with access control," in *2015 IEEE Conference on Computer Communications (INFOCOM)*, April 2015, pp. 2083–2091.