Encryption-Free Framework of Privacy-Preserving Image Recognition for Photo-Based Information Services

Kazuaki Nakamura, *Member, IEEE,* Naoko Nitta, *Member, IEEE,* and Noboru Babaguchi, *Senior Member, IEEE*

***Abstract*—Nowadays mobile devices such as smartphones have been widely used all over the world. In addition, the performance of image recognition has drastically increased with deep learning technologies. From these backgrounds, some photo-based information services provided in a client-server architecture are getting popular: client users take a photo of a certain spot and send it to a server, while the server identifies the spot with an image recognizer and returns its related information to the users. However, this kind of client-server image recognition can cause a privacy issue because image recognition results are sometimes privacy-sensitive. To tackle the privacy issue, in this paper, we propose a framework of privacy-preserving image recognition called EnfPire, in which the server cannot uniquely determine the recognition result but client users can do so. An overview of EnfPire is as follows: First client users extract a visual feature from their taken photo and transform it so that the server cannot uniquely determine the recognition result. Then the users send the transformed feature to the server, which returns a set of candidates of the recognition result to the users. Finally, the users compare the candidates to the original visual feature for obtaining the final result. Our experimental results demonstrate that EnfPire successfully degrades the server's spot-recognition accuracy from 99.8% to 41.4% while keeping 86.9% of the spot-recognition accuracy on the user side.**

***Index Terms*—privacy protection, image recognition, client-server architectures, feature transformation, photo-based information services**

I. Introduction

SINCE the development of deep belief networks by Hinton et al. [1], deep learning technologies [2] have been widely studied and introduced in various research areas with many successful results. One of such research areas is image recognition including object recognition [3], [4], [5] and scene recognition [6], [7], [8], [9], whose performance has drastically increased in the past decade. Moreover, nowadays mobile devices such as smartphones are very common and widely used all over the world. There are a lot of free or inexpensive services and applications working on smartphones. From these backgrounds, image recognition-based information services working on mobile devices are getting popular. In fact, several prototypes of such services have been already developed at non-commercial level. A typical example is non-marker-based mobile Augmented Reality [10], [11], in which the object captured by a built-in camera on a smartphone is recognized by image recognition techniques and some visual contents (e.g. virtual 3D model or advertising texts) of the recognized object are overlaid on the camera image for providing information to users. Another example is a tourist assistance system proposed by Zeng et al. [12], in which users can get guide information by taking a photo of a landmark, street, building, and so on and sending it to a cloud server that hosts image recognition services.

Client-server-based information services like the one developed in Zeng's work [12] are advantageous in that they can provide the latest information only by updating the server's information database and recognition criteria. However, they can also cause a privacy issue because image recognition results are sometimes privacy-sensitive, whose situation is described in detail in the following subsections.

A. Assumed Information Service

To describe the privacy issue in detail, we first assume the scenario of the information service. As Zeng's system does, we focus on a photo-based information service employing a client-server architecture. The service is provided in a certain public space where only a limited number of *spots* are included. We refer to such a space as a *field*. A service provider knows how many and what kind of *spots* exist in the *field*. A typical example of such spaces is a shopping mall that has various kinds of stores. In this example, each store in the shopping mall is a *spot*, and the shopping mall itself is a *field*. Other examples include a theme park consisting of a group of entertainment attractions and a city that has a lot of places for sightseeing.

In the *field*, the service provider creates a server system consisting of a database and an image recognizer. In the database, related information for each *spot* such as a product list, bargain products, customer evaluations (e.g. tweets for the *spot*), and congestion level is stored and updated in real-time. To get the information, client users take a photo of the *spot*, extract a visual feature from the photo, and send it to the server using their own smartphone. When receiving the visual feature from the users, the server recognizes the *spot* in the photo using the image recognizer and returns the corresponding information in the database to the users. Fig. 1 shows the overview of the assumed information service.

K. Nakamura, N. Nitta, and N. Babaguchi are with the Division of Electrical, Electronic, and Information Engineering, Graduate School of Engineering, Osaka University, Yamadaoka 2-1, Suita, Osaka, 565-0871 Japan. E-mail: (see http://www2c.comm.eng.osaka-u.ac.jp/en/staff.html). This work was supported by JSPS KAKENHI Grant Numbers JP16H06302 and JP17K00235.

Manuscript received February 27, 2018; revised October 10, 2018.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2018.2876752, IEEE Transactions on Information Forensics and Security

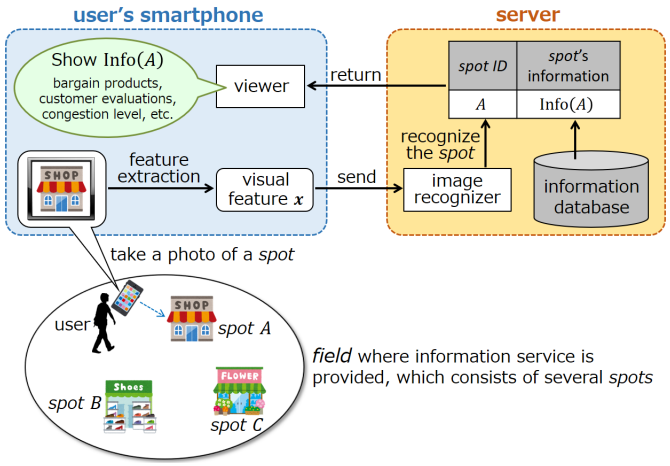

Fig. 1. Photo-based information service assumed in this study.

B. Privacy Issue

Client users of the above information service have to stay at a *spot* when taking its photo. This means the users' current location is disclosed to the server in terms of *spot* ID at the *spot*-recognition stage on the server side. Moreover, when the users send a visual feature of the photo to the server, some identifiers of the users' smartphone are also sent automatically, which can be used by the server for making a correspondence between current and past results of *spot*-recognition. This means not only the users' current location but also their location history is disclosed to the server. Because the location history can be viewed as the users' privacy information that reflects their interests and preference, it should be protected so that the server cannot get.

The above privacy issue is avoidable by making the server unable to uniquely determine the result of *spot*-recognition. However, if we simply degrade the server's recognition ability, the users are also unable to get the correct recognition result and may receive unnecessary information, i.e., information of another *spot*. Hence, it is desirable to establish a privacy-preserving recognition framework in which the server cannot uniquely determine the result of *spot*-recognition but the users can get the correct result. This can be achieved by a cryptographic technique as follows: the users send the server an encrypted version of a feature vector and the server runs its recognition process in the encrypted domain. However, this strategy can only work with a specific recognition algorithm. Moreover, encryption cost is computationally high in general, which is undesirable for mobile devices.

To solve the problem, in this paper, we propose an ENcryption-Free framework for Privacy-preserving Image REcogniton, which we call EnfPire, inspired by a privacy-preserving framework for image retrieval proposed by Weng et al. [13]. EnfPire does not impose any restrictions on the server's recognition algorithm because it does not rely on cryptographic techniques. In EnfPire, the server cannot identify the client users' current location; its candidates can only be presented. This can successfully abstract the location information, but is not enough for protecting its history. When the users continually use this kind of photo-based information service several times, their location history could be guessed by reducing the candidates of the user locations based on the geographical relationship between *spots*, which can be typically obtained from a map of the *field*. We refer to this kind of attack as map-based attack (MBA). To make EnfPire robust to MBA, we also propose its countermeasure. Our contributions in this paper are summarized as follows: First, we bring to light the privacy sensitiveness of image recognition results, which is a problem involved in not only photo-based information services but also general client-server image recognition. Second, we provide a novel framework for solving the problem, which is independent of recognition algorithms due to the encryption-freeness. Third, focusing on MBA, we further propose its countermeasure, which will be a basis of future work in the related research area.

The remainder of this paper is organized as below. In Section II, we review some previous work on privacy protection, mainly focusing on the aspect of visual information processing. Next, we describe EnfPire in detail in Section III, which is extended by introducing a countermeasure against MBA in Section IV. Moreover, we discuss EnfPire's resistance to attacks other than MBA in Section V. Then we experimentally evaluate the performance of EnfPire in Section VI and finally conclude this paper in Section VII.

Note that this paper is an extended version of our previous conference paper [14] in which we did not focus on MBA and only reported some preliminary results of simulation experiments. The differences from the conference paper are summarized as follows: First, we deeply discuss a possible procedure of the server's MBA and propose its countermeasure in Section IV. Second, we discuss EnfPire's resistance to other attacks in Section V. Third, we conduct not simulation experiments but actual experiments, in which EnfPire was compared to a state-of-the-art method and a lot of new results including evaluation results of the proposed MBA countermeasure were provided. These are reported in Section VI. In addition, we also theoretically and experimentally discuss EnfPire's computational complexity, which is to be described in Sections III-D and VI-C, and review the related work more widely in Section II.

II. Related Work

Visual contents such as images and video generally have two types of privacy-sensitive information: visual data itself and processing results of the visual data. Examples of the former include human faces, entire bodies, car license plates, and so on. The latter examples are results of content-based image retrieval (CBIR) and those of image recognition. In CBIR, not only a query image but also the gallery images retrieved with the query are sometimes privacy-sensitive because they reflect users' interest or preference. Hence, systems that can perform CBIR without disclosing such information to a retrieval server have been studied, which are called privacy-preserving CBIR (PCBIR) systems. Since CBIR and image recognition have a lot of common processes, research on privacy-preserving image recognition is often inspired by that on PCBIR.

In this section, we first focus on privacy protection methods for visual data itself in Section II-A. Then, we move our focus to privacy protection in CBIR and that in image recognition in Sections II-B and II-C, respectively. Finally, in Section II-D, we review some methods of location privacy protection, which are closely related to this work.

A. Privacy Protection for Visual Data

Methods for protecting privacy-sensitive regions in images and video have been widely studied in the past decade. Some of them visually abstract the privacy-sensitive regions by blocking out, silhouetting, pixelization, complete removal, and so on [15]. Chinomi et al. [16] proposed a system called PriSurv, which adaptively applies such operations to surveillance video based on the relationship between people in the video and its viewer. Mitsugami et al. [17] also focused on surveillance video and attempted to replace people in the video with rod-like symbols for protecting their privacy. To this end, they proposed a method of removing foreground regions in the surveillance video. Similar abstraction techniques are also used for dealing with the privacy issue of Google Street View [18], [19]. More recently, Zhang et al. [20] proposed an *anonymous camera* consisting of an infrared camera, a RGB camera, and a liquid crystal on silicon (LCoS) device, which can abstract face regions in video at the capturing phase by optical masking techniques. Inai et al. [21] focused on car license plates and proposed a method for concealing them in images using a character region detection technique.

Visual abstraction techniques such as pixelization and blurring are also used for the purpose of face de-identification [22], [23], which is a process of removing individual information from face regions in images and video. However, visual abstraction-based face de-identification are not necessarily effective for preventing automated face recognition systems. Moreover, visual abstraction often causes unnatural appearance. To cope with these problems, another de-identification approach has been studied in recent years: to replace the face regions with other face images. For a given face region, Bitouk et al. [24] proposed to select a face image similar to the given one from a pre-constructed face library and replace the given region with the selected face image by seamlessly blending their colors. Gross et al. proposed a method named k-Same [25] and its extension named k-Same-Select [26], by which an input face image is replaced with an average of its k-closest face images in a pre-constructed face dataset. Nakashima et al. [27] proposed a patch-based replacing approach for face de-identification without missing facial expressions.

As seen above, a lot of privacy protection methods for visual contents have been proposed. However, in the context of image retrieval and recognition, not only visual data itself but also its processing results are sometimes privacy-sensitive and should be protected.

B. Privacy Protection in CBIR

The purpose of PCBIR is to hide users' query image from a retrieval server as well as make the server unable to identify which gallery images are matched to the query image. This can be achieved by cryptographic techniques; that is, the users encrypt a visual feature extracted from their query image before sending it to the server and the server calculates the similarity between the visual feature of the query image and that of each gallery image in the encrypted domain. To realize this protocol, Lu et al. [28] proposed to use order preserving encryption (OPE). Instead of OPE, Zhang et al. [29] employed homomorphic encryption (HE), which is also used for privacy-preserving video retrieval [30]. There are several cryptography-based PCBIR methods focusing on specific features and metrics. Ferreira et al. [31] focused on color and texture features and proposed a PCBIR method that can efficiently encrypt such features. Xia et al. [32] proposed a PCBIR method specialized for Bag-of-words features and Earth Mover's Distance metric.

Instead of encrypting visual features extracted from original images, some studies proposed a method for directly extracting visual features from encrypted images. Its typical examples include privacy-preserving SIFT [33], [34] and its extensions [35], [36], which extract SIFT descriptors from an input encrypted image based on HE technique. Wang et al. [37] focused on shape-based features such as lines and circles and proposed a Hough Transform-based method to extract them from an input encrypted image.

Cryptography-based PCBIR methods, especially HE-based methods, are disadvantageous in computational efficiency. This is a serious problem for mobile applications. To solve the problem, several previous studies developed a PCBIR method that are not based on cryptographic techniques. In the method proposed by Weng et al. [13], a query image is first transformed into a hash code and a part of bits in the hash code are masked on the user side. Next, the partially masked hash code is sent to a retrieval server. The server compares the unmasked bits in the sent hash code with that of each gallery image in a database, and returns the user a set of images containing the same bits with the unmasked ones. Finally, the user screens the results returned from the server using the original hash code of the query for removing mismatched images. Fanti et al. [38] also proposed a similar framework. These studies inspired us to develop an encryption-free framework for privacy preserving image recognition, i.e., EnfPire.

C. Privacy Protection in Image Recognition

Compared to PCBIR, there are relatively fewer studies addressing privacy issues in image recognition, one of whose examples is privacy-preserving face recognition (PFR) firstly studied by Erkin et al. [39]. They assume a client-server architecture in which client users send a face image as a request to a recognition server, which only has a single face image per person as gallery images and compares the request face image with each gallery image to recognize the ID of the person in the request. Finally, the recognized ID is returned to the users. The purpose of PFR is to successfully perform the above protocol without disclosing the request face image and its recognition result to the server. To this end, Erkin et al. employed HE, restricting the server's face recognition

algorithm to nearest neighbor-based ones such as Eigenface [40]. This method was followed by several studies proposing its extensions [41], [42], [43]. Because the problem of image recognition using nearest neighbor algorithms is quite similar to that of image retrieval, Erkin's PFR method successfully protects the users' privacy as HE-based PCBIR methods do. However, it also has two disadvantages: high computational cost and the restriction on recognition algorithms. The former is due to the use of HE but partly solved in the extended methods. The latter is more serious and almost unavoidable because it is due to the use of a cryptography technique. Although there are a lot of cryptosystems that has additive and multiplicative homomorphism, most of them do not have other kinds of homomorphic properties. Therefore, only addition and multiplication operators can be used in HE-based methods, which makes it difficult to realize a cryptography-based recognition framework that can allow the use of various kinds of recognition algorithms. Several cryptography-based frameworks focusing on a specific recognizer (e.g. secure k-NN [44] and privacy-preserving SVM [45]) have been proposed, but they also have the same restriction. For instance, we can only use a polynomial kernel in privacy-preserving SVM.

Liu et al. [46] addressed a privacy issue in image recognition from another viewpoint. They proposed an image recognition framework that can use image data dispersed in a network as training samples without violating the privacy of data holders. In their framework, each data holder trains an image recognizer only using their own data and external users use each data holder's recognizer as a weak classifier, whose recognition results are integrated into the final result. This framework differs from ours in that they do not focus on the privacy of the users who want to get a recognition result.

D. Protection of Location Privacy

In our scenario, the client users' locations are represented as a *spot*-ID. Unlike this, in the mobile services based on GPS, the users' locations are represented as a numerical coordinate. There have been proposed a lot of methods for protecting the numerical location data. These methods can be classified into three types: cloaking area-based, transformation-based, and dummy-based.

The cloaking area-based methods [47], [48] assume a trusted third party, which mediates interactions between the server and the users. Based on a set of the exact locations sent from each user, the third party clusters it into several subsets and creates a cloaking area for each subset. The cloaking area includes all the exact locations in the corresponding subset. Then the party sends the information of each cloaking area to the server. Since the users' exact locations are spatially abstracted, the server cannot find them. However, this methods are disadvantageous in that it is difficult to prepare such a third party in practice.

In the transformation-based methods [49], [50], the users transform their exact location coordinate into another coordinate before sending it to the server. A typical approach is to replace the exact location coordinate with that of a certain near-by landmark such as an intersection or a building. However, these methods seriously degrade the quality of mobile services if there are no appropriate landmarks around the users' exact location.

In the dummy-based methods [51], [52], the users generate dummy locations differing from their actual location and send them all together to the server. After receiving the locations, the server prepares the information related to each received location and returns it to the users. Finally, the users choose the truly useful information related to their actual location. These methods are currently the mainstream since it is simple and practical.

The above methods, especially the cloaking area-based ones, cannot be directly applied to our assumed photo-based information service, since the location data is not a numerical coordinate in our case. However, transformation-based and dummy-based methods are suggestive, which are a basis of our work.

III. The Proposed Framework Named EnfPire

Inspired by the PCBIR method of Weng et al. [13], we propose an encryption-free framework for privacy-preserving image recognition, i.e., EnfPire, whose details are described in this section.

A. Notation

Let a positive integer set $\mathcal{S} = \{1, 2, \cdots, N\}$ be a set of *spots* in a *field*, where each integer i indicates the ID of the corresponding *spot* and N is the total number of the *spots*. We assume that, for each *spot* i, a set of its typical visual features $\mathcal{F}_i = \{\boldsymbol{f}_{ij} \in \mathbb{R}^n | j = 1, 2, \cdots, J\}$ is available for both a server and client users, where n is the dimension of the feature vectors and J is the number of available typical features for a *spot*. The whole feature set $\mathcal{F} = \bigcup_{i=1}^{N} \mathcal{F}_i$ can be provided in several ways. The easiest way is that the server provides, but some group of the client users or a certain third party can also provide. Images on the Web are also usable as $\mathcal{F}$ in some specific cases; for instance, in the case that the *field* is a city and the *spots* are famous places for sightseeing inside the city, there are a lot of license-free images of the places on the Web (e.g. Flicker), whose feature vectors can be used as $\boldsymbol{f}_{ij}$. In addition, let P be a *spot*'s photo taken by the client users with their smartphone, and let $\boldsymbol{x} \in \mathbb{R}^n$ be a visual feature extracted from the photo P.

B. Overview

The client users' current location is unavoidably leaked to the server if it can uniquely recognize the ID of the *spot* in P. Therefore, EnfPire tries to transform the feature vector $\boldsymbol{x}$ into $\boldsymbol{y} \in \mathbb{R}^n$ before sending it to the server in order to degrade the performance of *spot*-recognition. The overview of EnfPire is as follows (see also Fig. 2):

(1) A client user extracts a visual feature $\boldsymbol{x}$ from her taken photo P. Note that $\boldsymbol{x}$ is assumed to be effective enough for successfully recognizing the *spot*-ID of P by the server's image recognizer; in other words, we assume that the server's *spot*-recognition ability is very

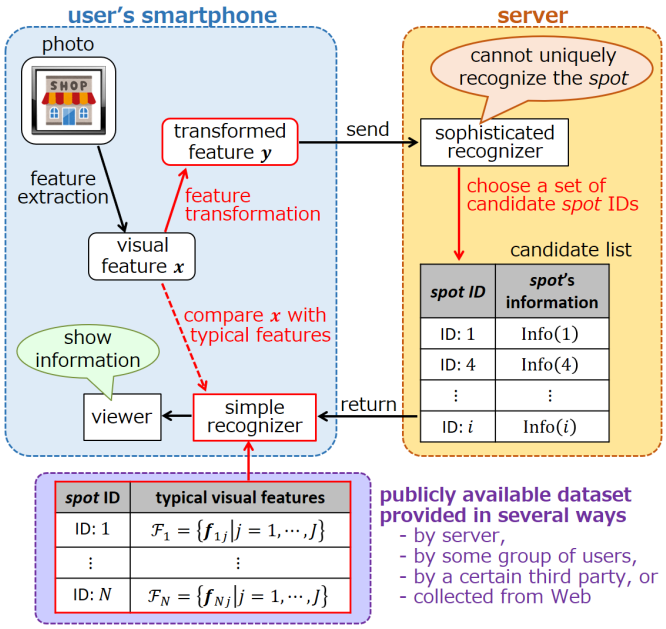

Fig. 2. Overview of the proposed framework named EnfPire.

high in this paper. This is because the server having low recognition ability cannot guess the user's current location well, which does not cause the privacy issue. In this case, EnfPire would not be needed.

(2) The user transforms the extracted feature $\boldsymbol{x}$ into $\boldsymbol{y}$ on the user's smartphone and sends it to the server. With the transformation, the effectiveness of the original feature $\boldsymbol{x}$ is degraded so that the server cannot uniquely recognize the *spot*-ID of P from $\boldsymbol{y}$. Note that $\boldsymbol{x}$ is not disclosed to the server because the transformation is done on the user side.

(3) Because $\boldsymbol{y}$ is less effective, the server does not uniquely recognize the *spot*-ID of P. Instead, a set of its candidates is chosen. More specifically, the server returns the user a set of candidates of correct *spot* ID $\hat{\mathcal{S}} \subset \mathcal{S}$ as well as the information of each candidate *spot* $i \in \hat{\mathcal{S}}$.

(4) For each candidate $i \in \hat{\mathcal{S}}$ returned from the server, the user compares the original feature $\boldsymbol{x}$ with $\boldsymbol{f}_{ij}$ $(j = 1, \cdots, J)$ and determines the final recognition result uniquely. Since this process is also done on the user side, the final result is not disclosed to the server. This means the server can only get several candidates of the user's current location.

In step (4), the users have to use a relatively simple recognizer that involves no training phase (such as 1-nearest neighbor algorithm) because of their limited computational resources. On the other hand, the server's recognizer is not restricted; many kinds of sophisticated algorithms including SVM, Neural Networks, AdaBoost, Random Forests, and naïve Bayes can be used unlike the existing PFR methods [39], [41].

C. Feature Transformation using Linear Subspace

A visual feature transformation from $\boldsymbol{x}$ into $\boldsymbol{y}$ is a key component of EnfPire. In this section, we describe how to design the transformation in detail.

In the PCBIR method of Weng et al. [13], they mask a part of bits in a retrieval query for preserving client users' privacy. This is a similar process with feature selection in the context of image recognition; that is, a part of dimensions in a feature vector are masked on the user side and the server runs its recognition process ignoring the masked dimensions. However, this process is not suitable to our assumed problem because the server can identify which dimensions were masked on the user side, and therefore can re-train a new recognizer that is specialized for the remaining dimensions. The new recognizer increases the server's *spot*-recognition performance, which is not desirable from the aspect of privacy protection. Therefore, we should employ a transformation method that makes the server unable to judge whether visual features sent from the users are original version or transformed version. To this end, we focus on a linear subspace of the original feature space of $\boldsymbol{x}$. Let L be an $n \times m$ matrix for projecting $\boldsymbol{x}$ onto a certain m-dimensional subspace, where $m < n$. Note that L satisfies $L^{\mathrm{T}}L = I_m$, where I_m is the m-dimensional unit matrix. By the matrix L, the original feature $\boldsymbol{x}$ is projected onto the linear subspace as $L^{\mathrm{T}}\boldsymbol{x} \in \mathbb{R}^m$, which can be back-projected to the original space as $LL^{\mathrm{T}}\boldsymbol{x} \in \mathbb{R}^n$. Since $\boldsymbol{x}$ and $LL^{\mathrm{T}}\boldsymbol{x}$ has the same dimension, $LL^{\mathrm{T}}\boldsymbol{x}$ seems to be an original feature vector for a person who has no knowledge of the L. Hence, we employ LL^{T} as an operator of the feature transformation and use $LL^{\mathrm{T}}\boldsymbol{x}$ as $\boldsymbol{y}$, namely $\boldsymbol{y} = LL^{\mathrm{T}}\boldsymbol{x}$.

The above feature transformation using the matrix L has the following two effects: (a) It makes the distribution of the transformed features different from that of the original features for each *spot* $i \in \mathcal{S}$. (b) It makes the distribution of the transformed features of a certain *spot* i overlap with that of another *spot* j. Because of these two effects, the server's *spot*-recognition performance is expected to be degraded. Note that the effect (a) is exerted on all the *spots* in $\mathcal{S}$ whereas the effect (b) is not necessarily exerted on all the *spot*-pairs; that is, there would be some *spots* whose feature distribution is not overlapped with that of any other *spots* even after the feature transformation. These *spots* can be accurately recognized by the server if it knows L. This is because the server knowing L can transform all the features in $\mathcal{F}$ by the L and re-train a new recognizer based on a set of the transformed features. Hence, it is very important for the users to hide their L from the server. Here, we discuss the method of Weng et al. [13] again. In fact, their method can be viewed as a limited version of our method; it is equivalent to using L each of whose column vector is a one-hot vector. Moreover, in Weng's method, L should be disclosed to the server, because the server has to specify which dimensions of the original features are masked in order to run its retrieval process ignoring the masked dimensions. Hence, the users cannot hide their L in Weng's method, and the server's *spot*-recognition performance is insufficiently degraded; only the effect (b) is exerted. In contrast, since our method allows the users to hide L from the server, the effect (a) is still exerted. This is why the proposed method is more suitable to the problem of privacy-preserving image recognition than Weng's method.

Now our focus is on how to design the matrix L. To degrade

the *spot*-recognition performance on the server side, for each *spot* i, its transformed feature vectors should not be separable from those of several other *spots*. However, if too many *spots* are not separable from each other, the performance of the *spot*-recognition and its computational cost on the user side become unacceptable. Based on this consideration, we design the L using $\mathcal{F}$ as follows:

(1) Randomly partition the *spot* set $\mathcal{S}$ into K disjoint clusters so that each cluster has at least two *spots*, which we refer to as $\mathcal{C}_k$ $(k = 1, \cdots, K)$ in the remainder.
(2) Find L that maximizes $\mathrm{tr}\left(L^{\mathrm{T}}\Sigma_{\mathrm{b}}(\mathcal{S})L\right)$ and $\mathrm{tr}\left(L^{\mathrm{T}}\Sigma_{\mathrm{w}}(\mathcal{C}_k)L\right)$ $(k = 1, \cdots, K)$ as well as minimizes $\mathrm{tr}\left(L^{\mathrm{T}}\Sigma_{\mathrm{w}}(\mathcal{S})L\right)$ and $\mathrm{tr}\left(L^{\mathrm{T}}\Sigma_{\mathrm{b}}(\mathcal{C}_k)L\right)$ $(k = 1, \cdots, K)$, where $\Sigma_{\mathrm{w}}(\mathcal{S})$ and $\Sigma_{\mathrm{b}}(\mathcal{S})$ are the within- and between-cluster scatter matrices computed with $\mathcal{F}$, while $\Sigma_{\mathrm{w}}(\mathcal{C}_k)$ and $\Sigma_{\mathrm{b}}(\mathcal{C}_k)$ are the within- and between-*spot* scatter matrices computed with $\bigcup_{i\in\mathcal{C}_k}\mathcal{F}_i$.

The above method is based on the idea of Linear Discriminant Analysis (LDA) [53]. Simultaneously minimizing the between-*spot* variance and maximizing the within-*spot* variance for each cluster $\mathcal{C}_k$ in step (2), two *spots* i and j are expected to be hardly separable in the transformed feature space if both of them belong to the same cluster, i.e., $i, j \in \mathcal{C}_k$. At the same time, simultaneously maximizing the between-cluster variance and minimizing the within-cluster variance for $\mathcal{S}$, two *spots* i and j are expected to be easily separable in the transformed feature space if they are not in the same cluster.

The optimal matrix $\hat{L}$ satisfying the above property can be obtained by finding L that maximizes $\mathrm{tr}\left(L^{\mathrm{T}}\Phi L\right)$ subject to the constraint $L^{\mathrm{T}}L = I_m$, where

$$\Phi = \alpha\Sigma_{\mathrm{b}}(\mathcal{S}) + \sum_{k=1}^{K}\Sigma_{\mathrm{w}}(\mathcal{C}_k) + \gamma\left\{\beta\Sigma_{\mathrm{w}}(\mathcal{S}) + \sum_{k=1}^{K}\Sigma_{\mathrm{b}}(\mathcal{C}_k)\right\}^{-1} \quad (1)$$

is a positive semi-definite matrix and α, β, and γ are positive constants for appropriately weighting each term in Eq. (1). This maximization problem can be solved by finding m-largest eigenvalues of Φ and their eigenvectors $\boldsymbol{u}_1, \cdots, \boldsymbol{u}_m$. Using the eigenvectors, we can obtain the optimal $\hat{L}$ as

$$\hat{L} = (\boldsymbol{u}_1 \ \cdots \ \boldsymbol{u}_m) \ . \quad (2)$$

Note that the server cannot get any knowledge of $\hat{L}$ because the users freely determine a partition for $\mathcal{S}$, i.e., $\mathcal{C}_1, \cdots, \mathcal{C}_K$, in step (1) and the partition is not disclosed to the server. Moreover, if needed, the users can also change the partition for $\mathcal{S}$ anytime they want. This means $\hat{L}$ used by each client user is not always same, which makes EnfPire more robust to the server's statistical analysis of $\hat{L}$.

D. Computational Complexity

EnfPire requires the client users to perform the following three processes:

(i) Computing L based on the features set $\mathcal{F}$,
(ii) Transforming a feature vector $\boldsymbol{x}$ into $\boldsymbol{y} = LL^{\mathrm{T}}\boldsymbol{x}$ before sending it to the server, and
(iii) Finding the final recognition result from the candidates returned by the server.

In this section, we analyze whether the computational complexity of these processes is acceptable or not for the users who have limited computational resources.

First, to compute L, the client users have to calculate Φ and its eigenvectors. This requires a computational cost of $O(JNn^2+n^3)$, where $O(JNn^2)$ is required for the construction of Φ and $O(n^3)$ is required for its eigen-decomposition. Although this cost is generally high, the users do not have to compute L in real-time when they use photo-based information services under EnfPire. This is because the users can preliminarily download $\mathcal{F}$ and compute L before actually using the service. Hence, the computational cost of the process (i) is not a problem. Note that the preliminary computation of L is not necessarily need to be done on the users' mobile device; the users are allowed to compute L on a desktop PC and move it to their own mobile device if they want. In addition, by preliminarily downloading $\mathcal{F}$, the users do not need to download any image data *spot*-by-*spot* in real-time when they are using the service. This greatly improves the feasibility of EnfPire.

Second, to transform $\boldsymbol{x}$ into $\boldsymbol{y}$, a computational cost of $O(mn)$ is required. More specifically, a cost of $O(mn)$ is required for the multiplication of $\boldsymbol{x}$ by L^{T} and another cost of $O(mn)$ is required for the multiplication of $L^{\mathrm{T}}\boldsymbol{x}$ by L. Because m is a constant that can be controlled by each user, the total cost of the process (ii) is $O(n)$. This is generally much less than the cost of feature extraction from an image, which requires at least a cost of $O(WH)$, where W and H are the width and height of the image. Therefore, the computational cost of the process (ii) is negligible.

The computational cost of the process (iii) highly depends on the recognition algorithm employed on the user side. Accordingly, we will experimentally evaluate it in Section VI-C.

IV. Countermeasure against Map-Based Attack

As mentioned in Section I, the original EnfPire proposed in the previous section could suffer from the server's map-based attack (MBA), whose procedure is briefly described as follows: First, the travel time distribution between two *spots* i and j is guessed for all pairs $(i, j) \in \mathcal{S}^2$ based on their geographical distance calculated with a map of the *field*. Next, each client user's actual travel time between *spots* is obtained by recording the arrival interval between two consecutive requests. Finally, the obtained actual travel time is compared with the guessed distribution in order to find the most likely sequence of the user's locations. To make EnfPire robust to MBA, we describe the details of the MBA procedure in Section IV-A and propose its countermeasure in Section IV-B.

A. Possible Procedure of Map-Based Attack

Suppose the case that a client user uses the assumed information service M times continually. Let $\boldsymbol{y}_l$ $(l = 1, \cdots, M)$ be a transformed visual feature sent from the user with her l-th use of the service and define $\mathcal{Y}$ as $\mathcal{Y} = \{\boldsymbol{y}_l \mid l = 1, \cdots, M\}$. Next, let t_l $(l = 2, \cdots, M)$ be the interval between the arrival time of $\boldsymbol{y}_{l-1}$ and that of $\boldsymbol{y}_l$ to the server, and let $\mathcal{T}$ be a

set of the time intervals, i.e., $\mathcal{T} = \{t_l \mid l = 2, \cdots, M\}$. As previously mentioned, the travel time distribution between any *spot* pair (i, j) is guessed in MBA. Let $\theta(i, j)$ denote the parameter of the distribution and let $\mathbf{\Theta}$ denote a set of the parameters, i.e., $\mathbf{\Theta} = \{\theta(i, j) \mid i, j \in \mathcal{S}\}$. In addition, let a variable o_l $(l = 1, \cdots, M)$ denote the server's recognition result for the visual feature $\boldsymbol{y}_l$ and define $\mathcal{O}$ as $\mathcal{O} = \{o_l \mid l = 1, \cdots, M\}$. Under the above setting, the server can estimate the user's location history by finding

$$\hat{\mathcal{O}} = \underset{\mathcal{O}}{\operatorname{argmax}}\, p(\mathcal{O}|\mathcal{Y}, \mathcal{T}, \mathbf{\Theta}) \ . \tag{3}$$

Based on the assumption that o_l only depends on the information at time steps l and $l-1$, the above probability can be decomposed as

$$p(\mathcal{O}|\mathcal{Y}, \mathcal{T}, \mathbf{\Theta}) = p(o_1|\boldsymbol{y}_1) \prod_{l=2}^{M} p(o_l|o_{l-1}, \boldsymbol{y}_l, t_l, \mathbf{\Theta}) \ , \tag{4}$$

where

$$p(o_l|o_{l-1}, \boldsymbol{y}_l, t_l, \mathbf{\Theta}) = \frac{p(\boldsymbol{y}_l, t_l, o_l|o_{l-1}, \mathbf{\Theta})}{\sum_{o_l} p(\boldsymbol{y}_l, t_l, o_l|o_{l-1}, \mathbf{\Theta})} \ . \tag{5}$$

Here we introduce several additional assumptions: First, $\boldsymbol{y}_l$ and t_l are conditionally independent given o_l and o_{l-1}. Second, $\boldsymbol{y}_l$ only depends on o_l. Third, prior distributions $p(o_l)$ and $p(o_l|o_{l-1}, \mathbf{\Theta})$ are uniform and can be represented by certain constants $\frac{1}{\eta}$ and ζ, respectively. Under these assumptions, $p(\boldsymbol{y}_l, t_l, o_l|o_{l-1}, \mathbf{\Theta})$ can be represented as

$$\begin{aligned} p(\boldsymbol{y}_l, t_l, o_l|o_{l-1}, \mathbf{\Theta}) &= \zeta p(\boldsymbol{y}_l, t_l|o_l, o_{l-1}, \mathbf{\Theta}) \\ &= \zeta p(\boldsymbol{y}_l|o_l) p(t_l|o_l, o_{l-1}, \mathbf{\Theta}) \\ &= \zeta\eta p(\boldsymbol{y}_l) p(o_l|\boldsymbol{y}_l) p(t_l|\theta(o_l, o_{l-1})). \end{aligned} \tag{6}$$

From Eq. (5) and (6), we can derive

$$p(o_l|o_{l-1}, \boldsymbol{y}_l, t_l, \mathbf{\Theta}) = p(o_l|\boldsymbol{y}_l) q(t_l|\theta(o_l, o_{l-1})) \ , \tag{7}$$

where

$$q(t_l|\theta(o_l, o_{l-1})) = \frac{p(t_l|\theta(o_l, o_{l-1}))}{\sum_{o_l} p(o_l|\boldsymbol{y}_l) p(t_l|\theta(o_l, o_{l-1}))} \ . \tag{8}$$

Using Eq. (4) and (7), we formulate $p(\mathcal{O}|\mathcal{Y}, \mathcal{T}, \mathbf{\Theta})$ as

$$p(\mathcal{O}|\mathcal{Y}, \mathcal{T}, \mathbf{\Theta}) = \prod_{l=1}^{M} p(o_l|\boldsymbol{y}_l) \prod_{l=2}^{M} q(t_l|\theta(o_l, o_{l-1})) \ . \tag{9}$$

Due to the monotonicity of the logarithm, we can obtain $\hat{\mathcal{O}}$ by maximizing

$$\begin{aligned} &\log p(\mathcal{O}|\mathcal{Y}, \mathcal{T}, \mathbf{\Theta}) \\ &\quad = \sum_{l=1}^{M} \log p(o_l|\boldsymbol{y}_l) + \sum_{l=2}^{M} \log q(t_l|\theta(o_l, o_{l-1})) \end{aligned} \tag{10}$$

instead of $p(\mathcal{O}|\mathcal{Y}, \mathcal{T}, \mathbf{\Theta})$, which can be efficiently computed with Viterbi algorithm.

In the above procedure, $p(o_l|\boldsymbol{y}_l)$ can be provided by the server's image recognizer as a recognition likelihood for the transformed feature $\boldsymbol{y}_l$, while $p(t|\theta(i, j))$ is modeled as follows in this paper: Assuming that the user's walking speed v follows a truncated normal distribution as

$$p(v) = \begin{cases} Z \exp\left(-\frac{(v-\mu)^2}{2\sigma^2}\right) & (v > 0) \\ 0 & (v \leq 0) \end{cases} \ , \tag{11}$$

where μ and σ are, respectively, the mean and standard deviation of the normal distribution and Z is a normalization constant, we model the distribution of travel time t between a *spot* pair (i, j) as

$$p(t|\theta(i, j)) = \begin{cases} Z \frac{d_{ij}}{t^2} \exp\left(-\frac{(d_{ij}-\mu t)^2}{2\sigma^2 t^2}\right) & (t > 0) \\ 0 & (t \leq 0) \end{cases} \ , \tag{12}$$

where $\theta(i, j) = d_{ij}$ is the geographical distance between *spot* i and *spot* j.

B. Countermeasure

Suppose the case that the user's $(l-1)$-th location is *spot* i and l-th location is *spot* j. In this case, the time interval t_l tends to be well matched to the distribution $p(t|d_{ij})$ unless the user consciously makes a significant change in her walking speed. This is why the MBA procedure presented in the previous section would work well even when the *spot*-recognition likelihood $p(o_l = j|\boldsymbol{y}_l)$ is not so high. Hence, if we make the server mistakenly obtain incorrect t_l, the probability $p(t_l|d_{ij})$ becomes low and therefore the MBA procedure cannot work well. To achieve this, we propose to automatically send a dummy request to the server as a countermeasure against MBA. Due to the dummy feature, the time interval t_l is not well matched to the distribution $p(t|d_{ij})$ for any $j \in \mathcal{S}$, which significantly degrades the performance of MBA. The concreate interaction between the server and the users for dummy requests is as follows: First, a dummy request is automatically sent to the server from the users' mobile device. For the sent request, the server next runs its recognition process as usual. Then the server's recognition result, i.e., a set of candidate *spot*-IDs, is returned to the users, which is finally removed in the users' mobile device automatically. Note that the users themselves are not aware of the dummy request and its recognition result.

To protect client users' privacy, we have to carefully choose the dummy feature: First, automatically generated features are not appropriate because such features tend to be dissimilar with the actual features of any *spot*, which can be easily guessed as dummy. Second, the features contained in $\mathcal{F}$ are not appropriate because $\mathcal{F}$ is available for not only the users but also the server, which can easily judge whether the sent feature is in $\mathcal{F}$ or not. Third, client users who are first-time visitors to the *field* would have no actual images of the *spots*. Considering these constraints, we propose to use the transformed version of any $\boldsymbol{f} \in \mathcal{F}$, i.e., $\boldsymbol{g} = LL^{\mathrm{T}}\boldsymbol{f}$, as the dummy feature. Due to the lack of knowledge of L, the server cannot find that $\boldsymbol{g}$ was obtained by transforming $\boldsymbol{f}$. Moreover, the recognition likelihood for $\boldsymbol{g}$ has the same tendency with that for the actual requests $\boldsymbol{y}$ due to the use of the same L. Hence, it is difficult for the server to distinguish which of the sent features are dummy.

We determine the sending time of dummy requests using an exponential distribution, which is often employed to describe a distribution of the time intervals between events. Whenever an actual request or a dummy request is sent to the server from the user side, we generate a positive real number τ as

$$\tau \sim \text{Exponential}(\lambda) = \lambda e^{-\lambda\tau} \ , \tag{13}$$

where λ is the parameter of the exponential distribution. Then we send a randomly chosen dummy request to the server after τ seconds of the last request. If the user manually sends the next actual request within τ seconds, we do not send a dummy feature and generate a new τ. Note that the countermeasure proposed above is not needed for the users who do not continually use the assumed information service.

V. Resistance to Other Attacks

In this section, we discuss EnfPire's resistance to some attacks other than MBA.

A. Reconstruction of the original features

If the server can reconstruct the original feature $\boldsymbol{x}$ from its transformed version $\boldsymbol{y}$, the client users' location information is easily leaked to the server. Therefore, we first discuss the possibility of the reconstruction of the original features.

Basically, it is impossible to exactly reconstruct $\boldsymbol{x}$ from $\boldsymbol{y} = LL^{\mathrm{T}}\boldsymbol{x}$ because two or more different original features are transformed to the same $\boldsymbol{y}$ by the linear projection. Suppose the case that $\boldsymbol{x}$ is represented as

$$\boldsymbol{x} = \boldsymbol{q} + \sum_{j=1}^{m} \xi_j \boldsymbol{u}_j \ , \tag{14}$$

where $\boldsymbol{u}_j$ is the j-th column vector of L given by Equation (2), ξ_j is a coefficient for $\boldsymbol{u}_j$, and $\boldsymbol{q}$ is the residual component that is normal to the subspace spanned by L. In this case, the transformed feature $\boldsymbol{y}$ is computed as

$$\boldsymbol{y} = \sum_{j=1}^{m} \xi_j \boldsymbol{u}_j \tag{15}$$

regardless of $\boldsymbol{q}$. This means that the server cannot get any information about $\boldsymbol{q}$ from $\boldsymbol{y}$, and therefore it cannot exactly reconstruct the original $\boldsymbol{x}$. However, if the server knows each user's L, it can obtain a set of pairs of original features and their transformed version by transforming all $\boldsymbol{f} \in \mathcal{F}$ by the L, and statistically estimate the probability $p(\boldsymbol{x}|\boldsymbol{y})$ using the set. If the estimated $p(\boldsymbol{x}|\boldsymbol{y})$ is not so ambiguous for some $\boldsymbol{y}$, its original version can be approximated with sufficient accuracy. Hence, it is quite important again to hide each user's L from the server.

When a client user continues to use the same L for a long time, the server can statistically guess it. This is because all the features sent from the user lie on the same subspace. The less the dimension of the subspace, i.e., m, is, the easier the statistical estimation of the corresponding L is. Hence, the users should compute their own L with relatively large m and regularly change it like a password.

B. Distinguishing Dummy Requests

The MBA countermeasure proposed in Section IV-B does not work well if the server can correctly distinguish which of the sent requests are dummy and which are not. Therefore, we next discuss how realistic it is for the server to distinguish dummy requests.

The most naïve way to distinguish dummy requests is to compare the time interval data t_l with the travel time distribution $p(t|d_{ij})$ for every *spot*-pair (i, j). If the probability $p(t = t_l|d_{ij})$ is high for a certain *spot*-pair, then both the l-th and $(l-1)$-th requests are guessed as actual requests. However, the reverse is not always true; in other words, $p(t = t_l|d_{ij})$ could be low even if neither the l-th request nor the $(l-1)$-th requests is dummy. This is because the client users do not necessarily send a request to the server in all *spots* they visited. For instance, suppose the case that a client user visits *spots* i, j, and k in this order but she sends a request to the server only in *spots* i and k. In this case, the time interval between the arrivals of the first and second requests does not necessarily match the distribution $p(t|d_{ik})$. This is because she does not take a direct route from the *spot* i to the *spot* k, while $p(t|d_{ik})$ modeled by Equation (12) is based on the geographical distance between i and k. More importantly, the server has no way to judge whether each user is sending a query in all *spots* s/he visited or not.

Another way to distinguish dummy requests is brute-force search, in which the server iteratively performs the following process: temporarily removing a subset of the users' requests as dummies and applying the MBA procedure presented in Section IV-A to obtain the score defined by Equation (10). Performing this process for all the possible subsets and finding the one having the maximum score, the server could be able to distinguish the dummy requests. However, since the number of the possible subsets is 2^M when a user sends M requests in total, the brute-force search requires a computational cost of $O(2^M)$. This cost is sufficiently low when and only when M is very small, but the MBA itself is not so powerful with small M.

The above discussions show that it is difficult for the server to distinguish dummy requests in EnfPire.

C. Collusion Attack

Finally, we discuss the possibility of a collusion attack in this section. In most of the client-server systems, the server suffers from a malicious client user. Sometimes two or more users try to attack the server in collusion. In this context, the server is an attacked side that has privacy information and the (malicious) users are an attacking side who try to steal the information. However, in the photo-based information services, the situation is different; the users are an attacked side who have privacy information, namely their locations, and the (malicious) server is an attacking side that tries to steal the information. Hence, in our context, we do not have to consider the collusion of multiple users. That is why, we consider the other two kinds of collusion. One is the collusion of two or more servers covering different *fields*, and the other is the collusion of the server and its confederate users. Note

that we use the term "confederate user" to refer to a malicious user who helps a malicious server to steal other users' location information.

1) Collusion of two or more servers: If a client user uses the same feature transformation matrix L for two or more servers covering different *fields*, they can collude to estimate the user's L. More specifically, the servers can share their received features and conduct a statistical analysis on a set of the shared features to estimate the user's L. However, this problem can be easily solved; each client user just has to use different L for different servers.

2) Collusion of the server and its confederate users: The confederate users can provide two kinds of information to the server. The first one is a set of pairs of original features and their transformed version computed by a certain L. Statistically analyzing the set, the server can estimate $p(\boldsymbol{x}|\boldsymbol{y})$ under the L. However, since the confederate users' L is generally different from the normal users' L, the estimated $p(\boldsymbol{x}|\boldsymbol{y})$ is not helpful for obtaining an accurate approximation of the original version of the normal users' requested features. Of course it is almost impossible for the confederate users to provide the above information for all the possible L, because different L are obtained by differently partitioning the *spot* set $\mathcal{S}$ into clusters and the number of ways to partition $\mathcal{S}$ is very huge in general.

The second information that can be provided from the confederate users is a set of precise time interval data; they can provide such a set by sending a request to the server in all *spots* they visited and setting L as $L = I_n$. This information can be exploited to make the travel time distribution $p(t|d_{ij})$ more reliable for every *spot*-pair (i, j), which improves the performance of the MBA procedure presented in Section IV-A. However, as long as the normal users use the countermeasure proposed in Section IV-B, i.e., dummy requests, the MBA procedure does not work well even when $p(t|d_{ij})$ is perfectly estimated.

Based on the above discussion, the information provided by the confederate users is not helpful for the server to estimate the normal users' L and their location information.

VI. Experimental Evaluation

We conducted two kinds of experiments in order to evaluate the privacy protection performance of EnfPire. In the first experiment, we evaluated how much the *spot*-recognition accuracy on the server side and that on the user side are degraded by the proposed feature transformation method. Note that we did not use the MBA procedure presented in Section IV-A on the server side in the first experiment. On the other hand, in the second experiment, we used the MBA procedure and evaluated how much the server's *spot*-recognition accuracy is degraded by the countermeasure proposed in Section IV-B.

A. Experimental Setup

The above two experiments were conducted at the EXPO '70 Commemorative Park located in Osaka, Japan. The park itself was assumed as a *field* and 16 places inside the park were assumed as *spots*. To collect photos of these 16 places, we recruited 14 people as participants of the experiments and instructed them to visit the park and take photos in three divided days between Feb. 19th and Apr. 30th in 2017. On the first and the second day, we instructed the participants to visit all of the 16 places and take at least 5 photos per place. As the result, we collected 310 photos per place, 260 of which were used as training data for training the server's image recognizer as well as constructing a set of typical visual features $\mathcal{F}_i$ $(i = 1, \cdots, 16)$. The remaining 50 photos were used as test images for the first experiment. On the third day, we instructed the participants to walk in the park freely and take a photo per place that they visited. As the result, we collected 252 photos altogether, which were used as test images for the second experiment. Moreover, for each of the 252 photos, we extracted its shooting time from its EXIF tag and used a set of the extracted shooting time as the time interval data $\mathcal{T}$. Fig. 3 shows the ID, the name, and an example photo of each place assumed as a *spot* in the experiments.

Fig. 3. 16 places assumed as *spots* in the experiments, which are located in EXPO '70 Commemorative Park, Osaka, Japan.

For each of the collected photos, we extracted a visual feature using AlexNet CNN model [3] pre-trained on ILSVRC 2012 ImageNet dataset. More specifically, we applied the CNN model to the collected photos using a deep learning framework *Caffe* [54] and extracted 4096-dimensinal feature vectors by choosing the output of the fc7 layer, which were then compressed to 256-dimensional vectors by Principal Component Analysis (PCA). We employed nonlinear SVM with a RBF kernel as a recognition method on the server side, whereas we employed 1-nearest neighbor algorithm for a recognition method on the user side. The SVM parameters of the server's recognizer were tuned by 2-fold cross validation with the grid search strategy.

In the remainder of this section, we use $A_{\mathrm{s}}(r)$ to represent the server's top-r recognition accuracy, i.e., the percentage that the correct *spot* ID of a test image is within its top-r candidates chosen by the server. Similarly, we use A_{u} to represent the recognition accuracy on the user side.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2018.2876752, IEEE Transactions on Information Forensics and Security

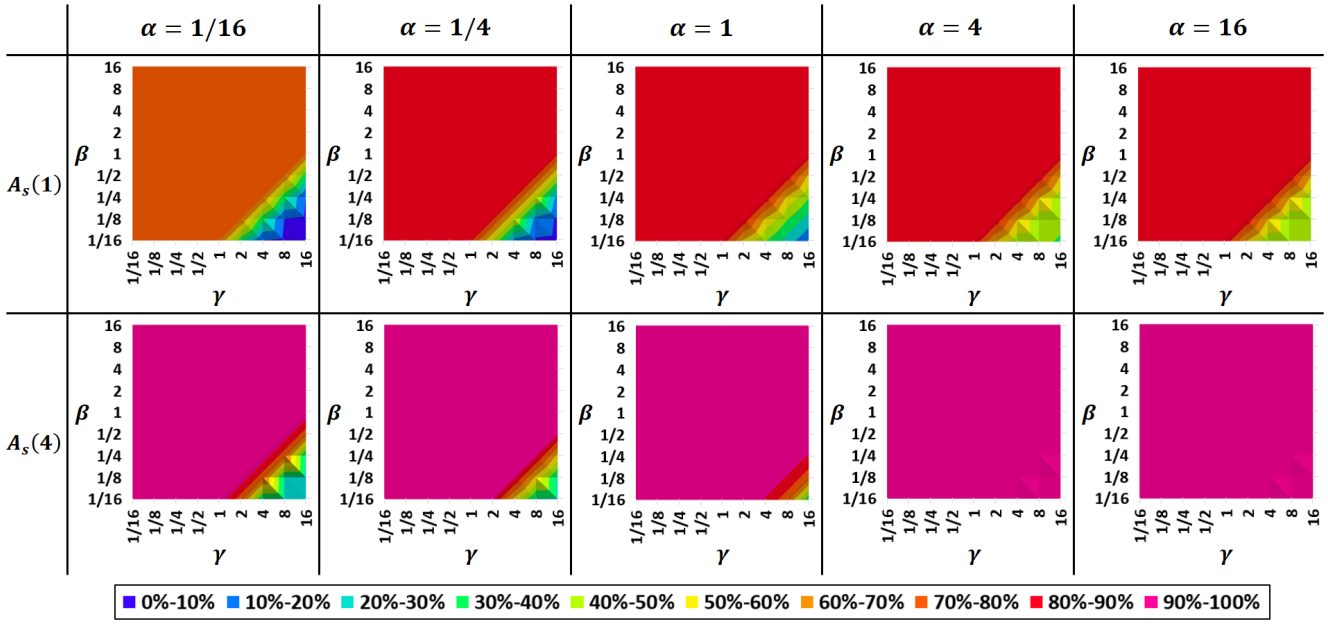

Fig. 4. Server's *spot*-recognition accuracy with various combinations of weight parameters α, β, and γ. The other parameters are fixed as $J = 200$, $m = 16$.

B. Parameter Setting

EnfPire has several parameters that can affect both $A_{\mathrm{s}}(r)$ and A_{u}. Before conducting the above two main experiments, we examined the effect of these parameters in order to tune them, using the photos collected for the first experiment.

1) Weight parameters α, β, and γ: First, we evaluated $A_{\mathrm{s}}(r)$ with various combinations of α, β, and γ, fixing the other parameters as $J = 200$ and $m = 16$. The way of partitioning a *spot* set $\mathcal{S} = \{1, 2, \cdots, 16\}$ was also fixed; we partitioned $\mathcal{S}$ into $\mathcal{C}_1 = \{1, 5, 8, 12\}$, $\mathcal{C}_2 = \{2, 6, 7, 15\}$, $\mathcal{C}_3 = \{4, 9, 10, 11\}$, and $\mathcal{C}_4 = \{3, 13, 14, 16\}$. Under this partition, each *spot* $i \in \mathcal{S}$ would be hardly separable only from the other three *spots* belonging to the same cluster with i. Hence, A_{u} would become sufficiently high when the server returns four candidates per test image. In this case, $A_{\mathrm{s}}(4)$ indicates an upper bound of A_{u}. Therefore, we mainly checked $A_{\mathrm{s}}(4)$ as well as $A_{\mathrm{s}}(1)$. Lower $A_{\mathrm{s}}(1)$ is desirable from the aspect of privacy protection, whereas higher $A_{\mathrm{s}}(4)$ is desirable because client users should be able to recognize the correct *spot* ID for getting appropriate *spot*-information. Fig. 4 shows the result of the evaluation.

We can see from Fig. 4 that the *spot*-recognition accuracy tends to be degraded when β is small and γ is large. This is especially pronounced with $A_{\mathrm{s}}(1)$. Smaller β and larger γ give much weight to $\Sigma_{\mathrm{b}}(\mathcal{C}_k)^{-1}$ for each $k \in \{1, \cdots, K\}$, which decreases the separability of the *spots* in the same cluster. Hence $A_{\mathrm{s}}(1)$ is remarkably degraded with small β and large γ. Basically, $A_{\mathrm{s}}(4)$ can also be degraded for the same reason. However, when α is large, a *spot* is rarely misrecognized as another *spot* belonging to a different cluster; most of the misrecognition cases occur between two *spots* belonging the same cluster. This is because larger α gives much weight to $\Sigma_{\mathrm{b}}(\mathcal{S})$, which increases the separability of the clusters. As the result, $A_{\mathrm{s}}(4)$ is not so degraded even with small β and large γ under the condition that α is large, as seen in Fig. 4. Conversely, smaller α much decreases the separability of the clusters, which often causes misrecognition between two *spots* belonging to different clusters. As the result, $A_{\mathrm{s}}(4)$ is simultaneously degraded with $A_{\mathrm{s}}(1)$ in this case, which is not

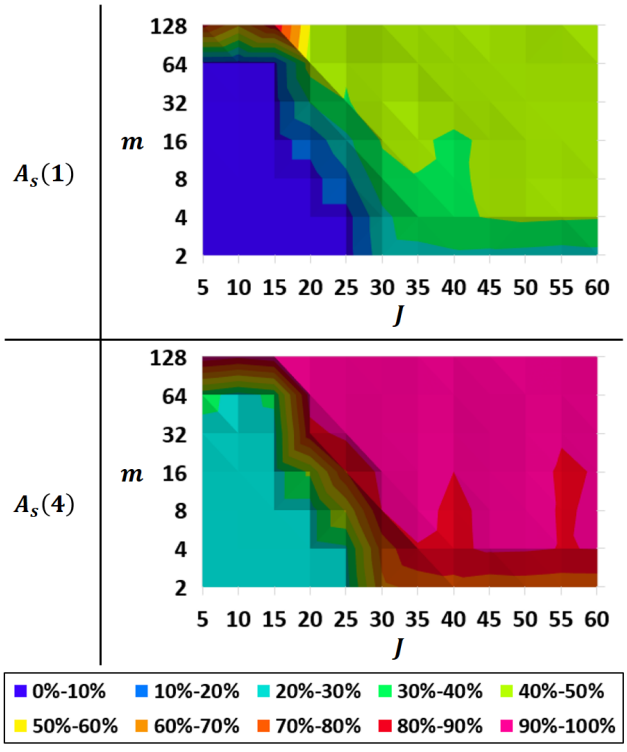

Fig. 5. Server's *spot*-recognition accuracy with various combinations of J and m. Weight parameters are fixed as $\alpha = 16$, $\beta = 1/16$, and $\gamma = 16$.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2018.2876752, IEEE Transactions on Information Forensics and Security

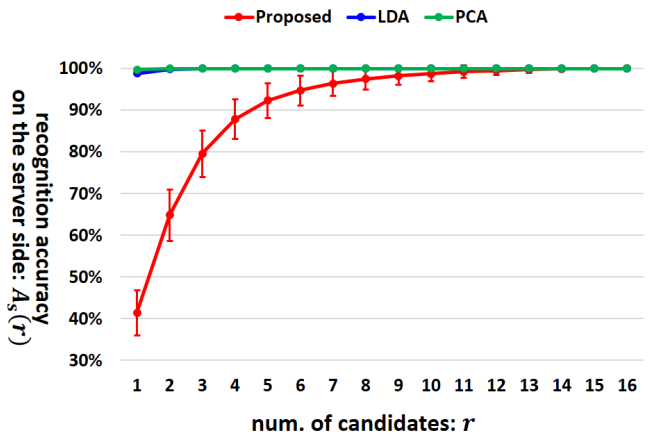

Fig. 6. Server's *spot*-recognition accuracy $A_s(r)$ in the cases of using PCA, LDA, and the proposed feature transformation method.

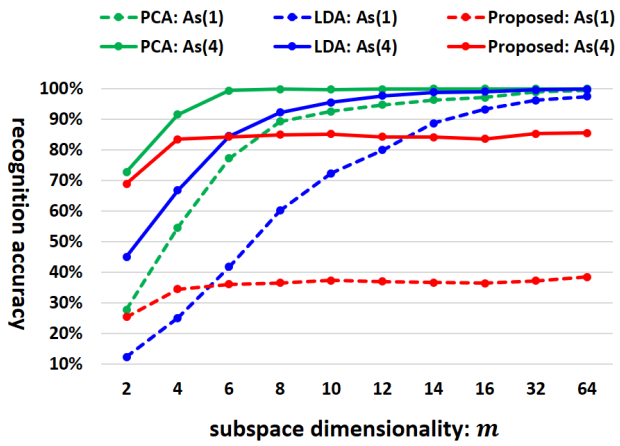

Fig. 7. $A_s(1)$ and $A_s(4)$ with much smaller m.

desirable. Hence, we should use large α. In addition, large α allows us to degrade $A_s(1)$ without degradation of $A_s(4)$ by employing small β and large γ. Based on the above discussion, we set $\alpha = 16$, $\beta = \frac{1}{16}$, and $\gamma = 16$ in the subsequent experiments.

2) The other parameters: Next, we evaluated the effect of the other two parameters, the number of visual features per *spot* J and the dimension of the linear subspace m, under the above setting of α, β, and γ. The *spot* set $\mathcal{S}$ was partitioned by the same way with the previous paragraph. Fig. 5 shows $A_s(1)$ and $A_s(4)$ with various combinations of J and m.

EnfPire considers the *spots*' feature distribution when constructing the feature transformation matrix L. However, the feature distribution cannot be well estimated with the limited number of sample feature vectors. This means the constructed matrix L does not reflect the actual feature distribution if J is too small. The linear subspace defined by such L is almost same to a randomly chosen one from the original feature space. Hence, both $A_s(1)$ and $A_s(4)$ are remarkably degraded with small m but remain high with large m when J is small, as seen in Fig. 5. On the other hand, if J is large enough, EnfPire can estimate the feature distribution well and construct a good L, which leads lower $A_s(1)$ and higher $A_s(4)$ desirably. In the result shown in Fig. 5, $J \geq 50$ seems enough. Considering an advantage of smaller J on computational efficiency, we set $J = 50$ in the subsequent experiments.

When J is large enough, m gives a limited influence to $A_s(1)$ and $A_s(4)$. This indicates the choice of m is not a critical matter from the aspect of the recognition accuracy. However, $m = \frac{n}{2}$ is desirable for protecting the information of L from the server, where n is the dimension of the original feature space. This is because the diversity of L is maximized with $m = \frac{n}{2}$, which makes the server's statistical analysis of L difficult even when the client users always use the same L. Therefore, we set $m = \frac{n}{2} = 128$ in the subsequent experiments.

C. Performance of Feature Transformation Method

Now we proceed to the main experiments.

1) Privacy protection performance: First, we evaluated $A_s(1)$ without any feature transformation. The result was 99.8%, which means almost all of the client users' location information can be unavoidably disclosed to the server. This indicates the importance of a framework for privacy-preserving image recognition like EnfPire.

Next, we evaluated how much $A_s(1)$ is degraded by the proposed feature transformation method, where the parameters α, β, γ, J, and m were set as described in the previous section. The *spot* set $\mathcal{S}$ was equally partitioned into four clusters so that each cluster has four *spots* selected randomly. Under the above condition, we evaluated $A_s(r)$ 100 times and calculated their average. This is because $A_s(r)$ can vary depending on the partitioning way of $\mathcal{S}$. For comparison, we also tested PCA and LDA as a method for constructing the feature transformation matrix L and also evaluated $A_s(r)$ in these two cases. Note that we also set $m = 128$ in the cases of PCA and LDA. Fig. 6 shows the result. The error bars on the curve of the proposed method indicate the standard deviation of its $A_s(r)$. We can see from Fig. 6 that $A_s(1)$ falls to 41.4% with the proposed feature transformation method. This indicates that it can successfully prevent the server from uniquely recognizing the client users' current location. In contrast, PCA and LDA can hardly preserve the users' location information with $m = 128$ because they do not degrade $A_s(1)$ at all; it keeps more than 98% after the feature transformation. This is because PCA and LDA are originally used for obtaining a compact linear subspace that keeps a high level of recognition ability. Therefore, a very low-dimensional subspace should be used in the cases of PCA and LDA to degrade $A_s(1)$. To further consider this point, we compared the proposed method with PCA and LDA using much smaller m, especially focusing on $A_s(1)$ and $A_s(4)$. Fig. 7 shows the result. As seen in Fig. 7, $A_s(1)$ of PCA and that of LDA are degraded only with very small m, but $A_s(4)$ is also tends to be degraded with such small m. This is undesirable because $A_s(4)$ means an upper bound of A_u when the server returns four candidate *spots* as a response to the users' request. Very narrow range of m can satisfy both low $A_s(1)$ and high $A_s(4)$ in the cases of PCA and LDA. This makes the tuning process of the parameter m

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TIFS.2018.2876752, IEEE Transactions on Information Forensics and Security

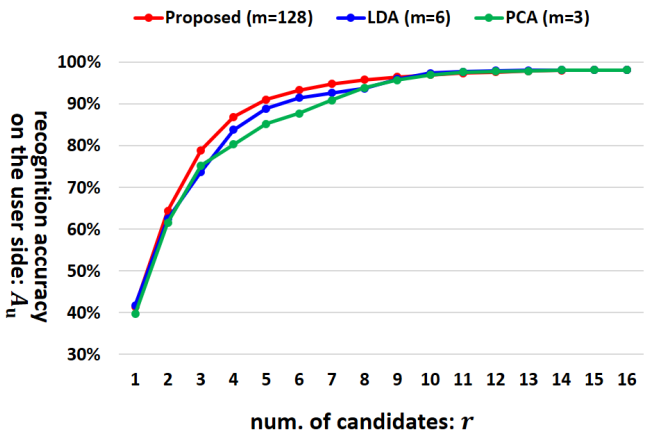

Fig. 8. *Spot*-recognition accuracy on the user side A_u.

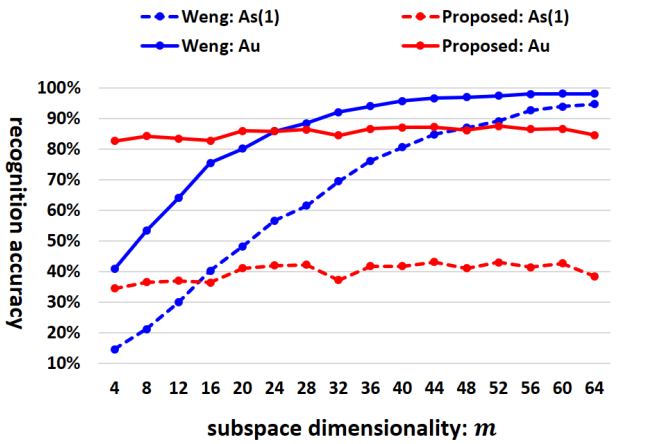

Fig. 9. Comparison of feature transformation method of EnfPire with that of Weng's framework.

quite difficult. Moreover, since PCA and LDA always provide the same linear subspace whose optimal dimension is very low, their robustness to the server's statistical analysis of L is limited. In contrast, the proposed method can provide various L that satisfies both low $A_s(1)$ and high $A_s(4)$, where the parameter m does not need to be finely tuned.

Ideally, $A_s(4)$ is expected to be close to 100% by the proposed method because each cluster has at most four *spots*. However, only 87.8% of $A_s(4)$ is achieved in the result shown in Fig. 6. This is because two *spot*-clusters $\mathcal{C}$ and $\mathcal{C}'$ are not always linearly separable. Because EnfPire employs a linear subspace for the feature transformation, the distribution of the transformed features of $\mathcal{C}$ and that of $\mathcal{C}'$ overlap with each other regardless of L if $\mathcal{C}$ and $\mathcal{C}'$ are not linearly separable. In this case, misrecognition between two *spots* $i \in \mathcal{C}$ and $j \in \mathcal{C}'$ can occur. This problem would be solvable by employing a nonlinear subspace for the feature transformation. To realize the solution without largely increasing the computational complexity is an important feature work.

Finally, we evaluated A_u. For fair comparison, we employed $m = 3$ for PCA, $m = 6$ for LDA, and $m = 128$ for the proposed method so that $A_s(1)$ of these three methods will be almost same. As we discussed above, it is not always sufficient for the users to receive only four candidate *spots*. Hence, we supposed the case that the server returns top r *spot*-IDs having the highest recognition likelihood to the users, and tested A_u with various r. Fig. 8 shows the result. $A_s(1)$ of PCA, LDA, and the proposed method are 39.6%, 41.7%, and 41.4%, respectively. The proposed method achieves $A_u = 86.9\%$ with $r = 4$ and $A_u = 91.1\%$ with $r = 5$, which outperforms the other two. This result indicates that it can provide a good recognition power to the user side. Of course the desired recognition power would be different depending on the type of information services and each user's personality. Thus the accuracy of around 90% might be sometimes insufficient. In such case, r should be adaptively controlled by each user. Since r affects the processing time of the final recognition process as we discussed later, the users need to consider the balance of the recognition accuracy and the processing time when they decide the optimal value of r.

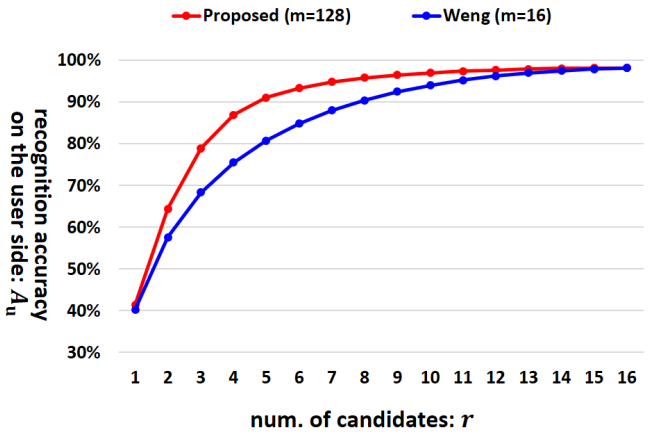

Fig. 10. A_u of proposed feature transformation method and that of Weng's one with various r.

The above experimental results are summarized as follows: EnfPire can degrade the server's recognition accuracy from almost 100% to around 40% while keeping 90% of recognition accuracy on the user side. This means that EnfPire can make the server unable to recognize users' current location as well as appropriately provide its information to the users. In other words, EnfPire successfully protects the users' location information without highly degrading the service quality in photo-based information services.

For further discussion, we compared EnfPire's feature transformation method with Weng's one [13]. In Weng's method, the feature transformation matrix L is disclosed to the server and it can re-train a new recognizer, as mentioned in Section III-C. Therefore, when measuring $A_s(1)$ and A_u, we used the re-trained recognizer on the server side in Weng's method. The comparison result is shown in Fig. 9. As shown in this figure, both $A_s(1)$ and A_u are highly degraded in Weng's method. Because their method is equivalent to using L each of whose column vector is a one-hot vector, the transformed features tend to become less informative. Therefore their method requires relatively large m to keep sufficiently high A_u. However, if we use large m, $A_s(1)$ becomes also high.

TABLE I
AVERAGE (AVG) AND STANDARD DEVIATION (SD) OF PROCESSING TIMES

Process	AVG	SD
(i) computation of L	0.605 [sec.]	0.0129 [sec.]
(ii) feature transformation	0.00339 [sec.]	0.000136 [sec.]
(iii) final recognition process	0.0398 [sec.]	0.0035 [sec.]

This is because the re-trained recognizer works better than the server's original recognizer. In fact, in the case of $m = 24$, A_{u} of the proposed method and that of Weng's method are almost same whereas $A_{\mathrm{s}}(1)$ of Weng's method is much higher than that of the proposed method. On the other hand, if we use smaller m to make $A_{\mathrm{s}}(1)$ low, then A_{u} becomes low. We can see this fact in Fig. 10, which shows A_{u} of the proposed method and that of Weng's one with various r. We employed $m = 128$ for the proposed method and $m = 16$ for Weng's method so that $A_{\mathrm{s}}(1)$ of the two methods will be almost same. The above results indicate that the privacy protection performance of EnfPire outperforms that of Weng's method.

2) Processing time: To experimentally evaluate the computational cost of EnfPire, we measured the processing time of the three processes mentioned in Section III-D, namely, (i) Computation of L, (ii) feature transformation, and (iii) final recognition process. More specifically, for each of these processes, we measured its processing time 100 times and evaluated their average and standard deviation. The parameters were set as $\alpha = 16$, $\beta = \frac{1}{16}$, $\gamma = 16$, $J = 50$, $m = 128$, and $r = 4$. Note that we used not a mobile device but a desktop PC with Core i7-2600 CPU and 8GB DDR3 RAM for measuring the processing time, which was made in 2012 as a middle range model. TABLE I shows the result.

We can see in the table that the actual processing time for computing L is 0.605 [sec.] on an average. This would be too long for real-time applications because mobile devices require much more time than desktop PCs, but in EnfPire, it is not a problem because L can be preliminarily computed, as mentioned in Section III-D. For the preliminary computation, the processing time of around 0.6 [sec.] would be sufficiently short. In contrast, the processing time of the feature transformation should be short for real-time computation, which is actually less than 0.004 [sec.] in TABLE I. This is sufficiently negligible even if mobile devices require 10 times more computational time than desktop PCs. The computational cost of the final recognition process highly depends on the algorithm employed on the user side. In our case, we employed 1-nearest neighbor algorithm, which requires a cost of $O(rJ)$. To confirm this, we evaluated the processing time of the final recognition process with various r and J. The results are shown in Fig. 11 and Fig. 12, in which the processing time almost linearly increases with r and J; it can be approximated as $0.000187rJ+0.00217$ [sec.]. This result supports the above theoretical consideration. In the case of $J = 50$ and $r = 4$, the processing time is 0.0398 [sec.] on an average, as shown in TABLE I. This is also sufficiently negligible.

Based on the above discussion, we conclude that the computational cost of EnfPire is acceptable even for the client users having limited computational resources.

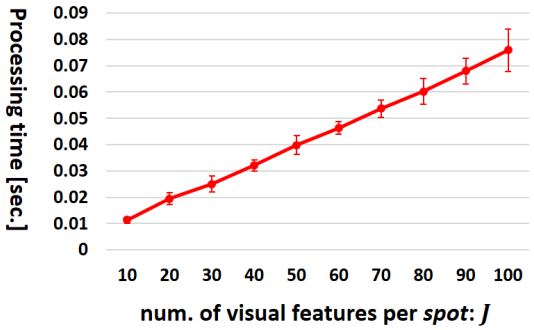

Fig. 11. Processing time of final recognition process on the client side with various J.

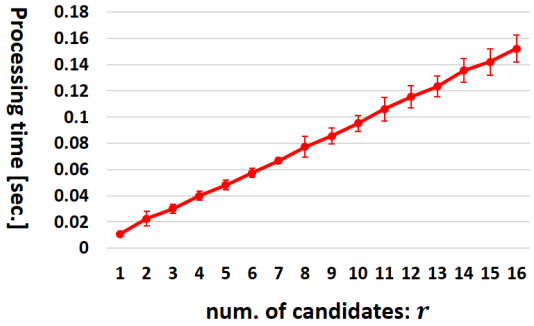

Fig. 12. Processing time of final recognition process on the client side with various r.

D. Performance of MBA Countermeasure

In the second main experiment, we evaluated the performance of the MBA countermeasure proposed in Section IV-B, using the same parameter setting with the previous section. The way of partitioning the *spot* set $\mathcal{S}$ into clusters was also same; that is, we partitioned $\mathcal{S}$ into four clusters so that each cluster has four *spots* selected randomly. Then we evaluated the *spot*-recognition accuracy 100 times and averaged them for each trial mentioned below.

First, we tested how much $A_{\mathrm{s}}(1)$ increases by the MBA procedure presented in Section IV-A. In the result, up to 77.4% of $A_{\mathrm{s}}(1)$ was achieved with $\mu = 0.97$ [m/s] and $\sigma = 0.55$ [m/s], where μ and σ are the mean and standard deviation of the distribution of client users' walking speed introduced in Eq. (11). Note that only 42.9% of $A_{\mathrm{s}}(1)$ was achieved without the MBA procedure, which is a consistent result with Fig. 6. This means more than 30% of performance improvement can be achieved with MBA. This is not desirable from the aspect of privacy protection for the users' location information.

Next, we tested the performance of the proposed countermeasure. To this end, we randomly chose several images as dummy from the training dataset $\mathcal{F}$ and virtually added them to the test dataset, simulating the process described in Section IV-B. Then we applied the server's image recognizer to the virtually generated test dataset with the MBA procedure and evaluated the recognition accuracy, i.e., $A_{\mathrm{s}}(1)$. Similarly, we also evaluated $A_{\mathrm{s}}(1)$ in the case without the MBA procedure and compared the two results. Note that we ignored the dummy images and their recognition results when measur-

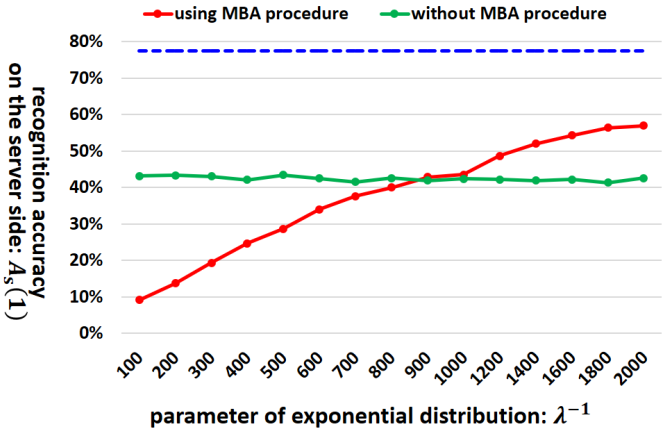

Fig. 13. $A_s(1)$ using/without MBA procedure presented in Section IV-A. Blue dashed line indicates recognition accuracy for original test dataset (without dummy requests) achieved by the MBA procedure.

ing $A_s(1)$. The performance of the proposed countermeasure highly depends on λ, which is the parameter of the exponential distribution in Eq. (13). Therefore, we separately performed the above comparison with different λ. Fig. 13 shows the result. $A_s(1)$ without the MBA procedure is always around 42% regardless of λ. This is consistent with the result shown in Fig. 6. On the other hand, in the case of using the MBA procedure, lower $A_s(1)$ is obtained with larger λ (or smaller λ^{-1} in Fig. 13). This is because more number of dummy requests were generated with larger λ. In Fig. 13, the proposed countermeasure successfully prevents the increase of $A_s(1)$ when λ is equal to or larger than $\frac{1}{900}$, where 80 dummy requests were generated on an average with $\lambda = \frac{1}{900}$. Since the total number of the actual requests was 252 in this experiment, one dummy per three actual requests would be enough for preventing the server's MBA.

The above experimental results are summarized as follows: Although the server's recognition performance is improved from around 40% to 75% by MBA, the proposed countermeasure can prevent such performance improvement successfully. This means that EnfPire can protect users' location history even when the server maliciously employs MBA.

VII. Conclusion

In this paper, we proposed a framework for privacy-preserving image recognition named EnfPire, focusing on photo-based information services. In such services, a server can easily recognize client users' current location as we experimentally showed. To protect the location information, EnfPire abstracts it using a linear subspace-based feature transformation. Because EnfPire does not rely on cryptographic techniques, it allows the use of various recognition algorithms on the server side. In our first experiment, EnfPire successfully degraded the server's recognition accuracy from 99.8% to 41.4% while keeping 86.9% on the user side, without finely tuning the parameters including the dimension of the linear subspace. This indicates the effectiveness of EnfPire. Moreover, even in the case that the server exploits map-based attack, the same degree of performance can be achieved by a dummy request-based countermeasure. Our second experiment indicates that one dummy per three actual requests would be enough. On the other hand, our experiments also show a problem of EnfPire; the recognition accuracy on the user side could degrade when the visual features of a certain *spot* are not linearly separable from those of another *spot*. This is due to the use of a linear subspace-based feature transformation. We believe that high linear-separability of the features between *spots* is generally ensured in photo-based information services. This is because a feature space whose dimension is much larger than the number of *spots* should be employed in order to make the server's original recognition ability high. This hypothesis will be demonstrated by an experiment with a larger number of *spots* in a future work. At the same time, we will attempt to develop a nonlinear extension of EnfPire's feature transformation method in another work.

In the deep learning era, it is more common for many image recognizers to receive an image itself rather than its visual feature as an input. Since we can regard an image as a high-dimensional vector, EnfPire is theoretically able to work with such recognizers. However, in practice, it is non-trivial whether subspace-based methods are suitable to such case or not. Experimental evaluation of this point is also important work.

References

[1] G. E. Hinton, S. Osindero, and Y. Teh: "A Fast Learning Algorithm for Deep Belief Nets," Neural Computation, Vol.18, No.7, pp.1527–1554, 2006.

[2] L. Deng and D. Yu: "Deep Learning: Methods and Applications," Foundations and Trends in Signal Processing, Vol.7, No.3–4, pp.197–387, 2014.

[3] A. Krizhevsky, I. Sutskever, and G. E. Hinton: "ImageNet Classification with Deep Convolutional Neural Networks," in Proceedings of the 25th International Conference on Neural Information Processing Systems, pp.1097–1105, 2012.

[4] P. Agrawal, R. Girshick, J. Malik: "Analyzing the Performance of Multilayer Neural Networks for Object Recognition," in Proceedings of the 13th European Conference on Computer Vision, pp.329–344, 2014.

[5] M. Liang and X. Hu: "Recurrent Convolutional Neural Network for Object Recognition," in Proceedings of the 28th IEEE Conference on Computer Vision and Pattern Recognition, pp.3367–3375, 2015.

[6] M. Koskela and J. Laaksonen: "Convolutional Network Features for Scene Recognition," in Proceedings of the 22nd ACM International Conference on Multimedia, pp.1169–1172, 2014.

[7] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva: "Learning Deep Features for Scene Recognition using Places Database," in Proceedings of the 27th International Conference on Neural Information Processing Systems, pp.487–495, 2014.

[8] L. Herranz, S. Jiang, and X. Li: "Scene Recognition with CNNs: Objects, Scales and Dataset Bias," in Proceedings of the 29th IEEE Conference on Computer Vision and Pattern Recognition, pp.571–579, 2016.

[9] G. Xie, X. Zhang, S. Yan, and C. Liu: "Hybrid CNN and Dictionary-Based Models for Scene Recognition and Domain Adaptation," IEEE Transactions on Circuits and Systems for Video Technology, Vol.27, No.6, pp.1263–1274, 2017.

[10] D. Kim and D. Hwang: "Non-Marker based Mobile Augmented Reality and its Applications using Object Recognition," Journal of Universal Computer Science, Vol.18, No.20, pp.2832–2850, 2012.

[11] Y. Kim and W. Kim: "Implementation of Augmented Reality System for Smartphone Advertisements," International Journal of Multimedia and Ubiquitous Engineering, Vol.9, No.2, pp.385–392, 2014.

[12] Y. Zeng, Y. Chan, T. Lin, M. Shih, P. Hsieh, and G. Chao: "Scene Feature Recognition-Enabled Framework for Mobile Service Information Query System," in Proceedings of the 17th International Conference on Human Interface and the Management of Information, 2015.

[13] L. Weng, L. Amsaleg, A. Morton, and S. Marchand-Maillet: "A Privacy-Preserving Framework for Large-Scale Content-Based Information Retrieval," IEEE Transactions on Information Forensics and Security, Vol.10, No.1, pp.152–167, 2015.

[14] K. Fujii, K. Nakamura, N. Nitta, and N. Babaguchi: "A Framework of Privacy-Preserving Image Recognition for Image-Based Information Services," in Proceedings of the 23rd International Conference on Multimedia Modeling, pp.40–52, 2017.

[15] A. Cavallaro, O. Steiger, and T. Ebrahimi: "Semantic Video Analysis for Adaptive Content Delivery and Automatic Description," IEEE Transactions on Circuits and Systems for Video Technology, Vol.15, No.10, pp.1200–1209, 2005.

[16] K. Chinomi, N. Nitta, Y. Ito, and N. Babaguchi: "PriSurv: Privacy Protected Video Surveillance System Using Adaptive Visual Abstraction," in Proceedings of the 14th International Conference on Multimedia Modeling, pp.144–154, 2008.

[17] I. Mitsugami, M. Mukunoki, Y. Kawanishi, H. Hattori, and M. Minoh: "Privacy-Protected Camera for the Sensing Web," in Proceedings of the 13th International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems, pp.622–631, 2010.

[18] A. Frome, G. Cheung, A. Abdulkader, M. Zennaro, B. Wu, A. Bissacco, H. Adam, H. Neven, and L. Vincent: "Large-Scale Privacy Protection in Google Street View," in Proceedings of the 12th International Conference on Computer Vision, pp.2373–2380, 2009.

[19] A. Flores and S. Belongie: "Removing Pedestrians from Google Street View Images," in Proceedings of the 2nd International Workshop on Mobile Vision, pp.53–58, 2010.

[20] Y. Zhang, Y. Lu, H. Nagahara, and R. Taniguchi: "Anonymous Camera for Privacy Protection," in Proceedings of the 22nd International Conference on Pattern Recognition, 2014.

[21] K. Inai, M. Palsson, V. Frinken, Y. Feng, and S. Uchida: "Selective Concealment of Characters for Privacy Protection," in Proceedings of 22nd International Conference on Pattern Recognition, 2014.

[22] M. Boyle, C. Edwards, and Saul Greenberg: "The Effects of Filtered Video on Awareness and Privacy," in Proceedings of the 2000 ACM Conference on Computer Supported Cooperative Work, pp.1–10, 2000.

[23] C. Neustaedter, S. Greenberg, and M. Boyle: "Blur Filtration Fails to Preserve Privacy for Home-Based Video Conferencing," ACM Transactions on Computer-Human Interaction, Vol.13, No.1, pp.1–36, 2006.

[24] D. Bitouk, N. Kumar, S. Dhillon, P. Belhumeur, S. K. Nayar: "Face Swapping: Automatically Replacing Faces in Photographs," ACM Transactions on Graphics, Vol.27, No.3, 2008.

[25] R. Gross, E. Airoldi, B. Malin, and L. Sweeney: "Integrating Utility into Face De-identification," in Proceedings of the 5th International Conference on Privacy Enhancing Technologies, pp.227–242, 2005.

[26] R. Gross, L. Sweeney, F. de la Torre, S. Baker: "Model-Based Face De-identification," in Proceedings of the 2006 Conference on Computer Vision and Pattern Recognition Workshop, 2006.

[27] Y. Nakashima, T. Koyama, N. Yokoya, and N. Babaguchi: "Facial Expression Preserving Privacy Protection using Image Melding," in Proceedings of the 2015 IEEE International Conference on Multimedia and Expo, 2015.

[28] W. Lu, A. Swaminathan, A. L. Varna, and M. Wu: "Enabling Search over Encrypted Multimedia Databases," in Proceedings of the SPIE Conference on Media Forensics and Security, Vol.7254, pp.18–29, 2009.

[29] L. Zhang, T. Jung, P. Feng, K. Liu, X. Li, and Y. Liu: "PIC: Enable Large-Scale Privacy Preserving Content-Based Image Search on Cloud," in Proceedings of the 44th International Conference on Parallel Processing, 2015.

[30] W. Chu and F. Chang: "A Privacy-Preserving Bipartite Graph Matching Framework for Multimedia Analysis and Retrieval," in Proceeding of the 5th ACM International Conference on Multimedia Retrieval, pp.243–250, 2015.

[31] B. Ferreira, J. Rodrigues, J. Leitao, and H. Domingos: "Privacy-Preserving Content-Based Image Retrieval in the Cloud," in Proceedings of the 34th IEEE Symposium on Reliable Distributed Systems, 2015.

[32] Z. Xia, Y. Zhu, X. Sun, Z. Qin, and K. Ren: "Towards Privacy-Preserving Content-Based Image Retrieval in Cloud Computing," IEEE Transactions on Cloud Computing. (to be published)

[33] C. Hsu, C. Lu, and S. Pei: "Secure and Robust SIFT," in Proceedings of the 17th ACM International Conference on Multimedia, pp.637–640, 2009.

[34] C. Hsu, C. Lu, and S. Pei: "Image Feature Extraction in Encrypted Domain with Privacy-Preserving SIFT," IEEE Transactions on Image Processing, Vol.21, No.11, pp.4593–4607, 2012.

[35] Z. Qin, J. Yan, K. Ren, C. Chen, and C. Wang: "Towards Efficient Privacy-Preserving Image Feature Extraction in Cloud Computing," in Proceedings of the 22nd ACM International Conference on Multimedia, pp.497–506, 2014.

[36] Z. Qin, J. Yan, K. Ren, C. Chen, and C. Wang: "SecSIFT: Secure Image SIFT Feature Extraction in Cloud Computing," ACM Transactions on Multimedia Computing, Communications, and Applications, Vol.12, No.4, 2016.

[37] S. Wang, M. Nassar, M. Atallah, and Q. Malluhi: "Secure and Private Outsourcing of Shape-Based Feature Extraction," in Proceedings of the 15th International Conference on Information and Communications Security, 2013.

[38] G. Fanti, M. Finiasz, and G. Friedland: "Toward Efficient, Privacy-Aware Media Classification on Public Databases," in Proceedings of the 4th International Conference on Multimedia Retrieval, 2014.

[39] Z. Erkin, M. Franz, J. Guajardo, S. Katzenbeisser, I. Lagendijk, and T. Toft: "Privacy-Preserving Face Recognition," in Proceedings of the 9th International Symposium on Privacy Enhancing Technologies, pp.235–253, 2009.

[40] M. Turk and A. Pentland: "Eigenfaces for Recognition," Journal of Cognitive Neuroscience, Vol.3, No.1, pp.71–86, 1991.

[41] A. Sadeghi, T. Schneider, and I. Wehrenberg: "Efficient Privacy-Preserving Face Recognition," in Proceedings of the 12th International Conference on Information Security and Cryptology, pp.229–244, 2009.

[42] J. Bringer, H. Chabanne, and A. Patey: "Privacy-Preserving Biometric Identification Using Secure Multiparty Computation: An Overview and Recent Trends," IEEE Signal Processing Magazine, Vol.30, No.2, pp.42–52, 2013.

[43] C. Xiang, C. Tang, Y. Cai, and Q. Xu: "Privacy-Preserving Face Recognition with Outsourced Computation," Soft Computing, Vol.20, No.9, pp.3735–3744, 2016.

[44] M. Kantarcıoğlu and C. Clifton: "Privately Computing a Distributed k-NN Classifier," in Proceedings of the 8th European Conference on Principles of Data Mining and Knowledge Discovery, 2004.

[45] Y. Rahulamathavan, R. C. Phan, S. Veluru, K. Cumanan, and M. Rajarajan: "Privacy-Preserving Multi-Class Support Vector Machine for Outsourcing the Data Classification in Cloud," IEEE Transactions on Dependable and Secure Computing, Vol.11, No.5, pp.467–479, 2014.

[46] C. Liu, Z. Shang, and Y. Y. Tangc: "An Image Classification Method That Considers Privacy-Preservation," Neurocomputing, Vol.208, No.5, pp.80–98, 2016.

[47] R. Cheng, Y. Zhang, E. Bertino, and S. Prabhakar: "Preserving User Location Privacy in Mobile Data Management Infrastructures," in Proceedings of the 6th International Workshop on Privacy Enhancing Technologies, pp.393–412, 2006.

[48] B. Lee, J. Oh, H. Yu, and J. Kim: "Protecting Location Privacy using Location Semantics," in Proceedings of the 17th ACM International Conference on Knowledge Discovery and Data Mining, pp.1289–1297, 2011.

[49] M. Duckham and L. Kulik: "A Formal Model of Obfuscation and Negotiation for Location Privacy," in Proceedings of the 2005 International Conference on Pervasive Services, pp.152–170, 2005.

[50] C. A. Ardagna, M. Cremonini, E. Damiani, S. De Capitani di Vimercati, and P. Samarati: "Location Privacy Protection Through Obfuscation-Based Techniques," in Proceedings of the 21st IFIP WG 11.3 Working Conference on Data and Applications Security, pp.47–60, 2007.

[51] H. Kido, Y. Yanagisawa, and T. Satoh: "An Anonymous Communication Technique using Dummies for Location-based Services," in Proceedings of the 2005 International Conference on Pervasive Services, pp.88–97, 2005.

[52] T. Hara, A. Suzuki, M. Iwata, Y. Arase, and X. Xie: "Dummy-Based User Location Anonymization Under Real-World Constraints," IEEE Access, Vol.4, pp.673-687, 2016.

[53] S. Mika, G. Ratsch, J. Weston, B. Scholkopf, and K. R. Mullers: "Fisher Discriminant Analysis with Kernels," Neural Networks for Signal Processing, Vol.9, pp.41–48, 1999.

[54] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell: "Caffe: Convolutional Architecture for Fast Feature Embedding," in Proceedings of the 22nd ACM International Conference on Multimedia, pp.675–678, 2014.

Kazuaki Nakamura received the B.S. degree in Engineering from Kyoto University in 2005, and the M.S. and Ph.D. degrees in Informatics from Kyoto University in 2007 and 2011, respectively. He is currently an Assistant Professor at Graduate School of Engineering, Osaka University, from 2012. His research interests include image processing, image recognition, and video analysis. He is a member of IEEE, IEICE, IPSJ, and ITE.

Naoko Nitta received the B.E., M.E., and Ph.D. degrees in Engineering Science from Osaka University, in 1998, 2000, and 2003, respectively. She is currently an Associate Professor in Graduate School of Engineering, Osaka University. From 2002 to 2004, she was a research fellow of the Japan Society for the Promotion of Science. From 2003 to 2004, she was a Visiting Scholar at Columbia University, New York. Her research interests are in the areas of video content analysis and image/audio processing. She received Best Paper Award of 2006 Pacific-Rim Conference on Multimedia (PCM2006). She is a member of IEEE, ACM, IEICE, and ITE.

Noboru Babaguchi received the B.E., M.E. and Ph.D. degrees in communication engineering from Osaka University, in 1979, 1981 and 1984, respectively. He is currently a Professor of the Department of Information and Communications Technology, Osaka University. From 1996 to 1997, he was a Visiting Scholar at the University of California, San Diego. His research interests include image/video analysis, multimedia computing and intelligent systems. Recently, he has been engaged in privacy protection for visual information, and security and fabrication of multimedia. He has published over 250 journal and conference papers and several textbooks.

Dr. Babaguchi received Best Paper Award of PCM2006 and IAS2009. He is on the editorial board of Multimedia Tools and Applications, and New Generation Computing. He served as a Guest Editor of IEEE TIFS, Special Issue on Intelligent Video Surveillance for Public Security & Personal Privacy. He also served as a workshop Co-chair of MIR2001, a Track Co-chair of IEEE ICME2006, a General Co-chair of MMM2008, a Sub-Track Chair of APSIPA ASC 2009, a General Co-Chair of ACM Multimedia 2012, a Track Co-Chair of ICPR2012, an Area Chair of IEEE ICME2013, and an Honorary Co-Chair of ACM ICMR2018. He also served on program committee of international conferences in multimedia related fields. He is a Fellow of IEICE, a Senior Member of IEEE, and a member of ACM, IPSJ, ITE and JSAI.