Efficient Privacy-Preserving Facial Expression Classification

Yogachandran Rahulamathavan, *Member, IEEE*, and Muttukrishnan Rajarajan, *Senior Member, IEEE*

Abstract—This paper proposes an efficient algorithm to perform privacy-preserving (PP) facial expression classification (FEC) in the client-server model. The server holds a database and offers the classification service to the clients. The client uses the service to classify the facial expression (FaE) of subject. It should be noted that the client and server are mutually untrusted parties and they want to perform the classification without revealing their inputs to each other. In contrast to the existing works, which rely on computationally expensive cryptographic operations, this paper proposes a lightweight algorithm based on the randomization technique. The proposed algorithm is validated using the widely used JAFFE and MUG FaE databases. Experimental results demonstrate that the proposed algorithm does not degrade the performance compared to existing works. However, it preserves the privacy of inputs while improving the computational complexity by 120 times and communication complexity by 31 percent against the existing homomorphic cryptography based approach.

Index Terms—Privacy, security, facial expression, classification, encrypted domain.

1 Introduction

Facial expression classification (FEC) forms a critical capability desired by human-interacting systems that aim to be responsive to variations in the human's emotional state [1]–[3]. Automatic recognition of FaEs can be an important component in human-machine interfaces, human emotion analysis and medical care. However, the task of automatically recognizing the various FaEs is challenging [4]–[6].

The FEC could be automated by establishing a computing infrastructure capable of executing machine learning algorithms and building a database of facial images with different expressions. These requirements may not achievable by small organizations. Even if we assume that the requirement can be met by organizations, then they need a dedicated team to maintain the infrastructure which may not be necessary when cheaper computational power and services can be obtained from cloud providers. Hence, it is reasonable to assume that any organization can obtain the FEC as a service from cloud provider or any other modern computing infrastructure.

If the FEC is outsourced to modern computing infrastructures then it involves third party servers who deliver FEC as a service via the Internet. Initially, the server sets up a database with training facial images from various expression classes and extracts the discriminative features of these images that correspond to each class in order to simplify the classification. A client requesting the service would supply a test image whose expression it desires to recognize. In response the server extracts the discriminative features from the test image and compares them against the features of training images in order to find the test image's expression.

However, due to privacy concerns, the client requesting the service may not wish to disclose the contents of the facial image or the classification results (i.e., which expression is present in the current test facial image) to the server; while the server desires that the service does not leak out information to the client on the reference facial images that it holds in its database. As an example, a doctor may wish for her patients' emotions to be automatically recognized (rather than manually as this will be time-consuming), through the aid of a FaE database hosted on a remote server. In order to address any privacy concerns that may result from revealing patient images without explicit consent, this should be performed without leaking the contents of the patients' face images nor even the classification results to the server.

The challenge hence becomes the question of how to perform the classification collaboratively between two mutually distrusting parties without revealing the private contents of the images held by either party; and without revealing the result to the server. In literature this challenge was addressed by exploiting state-of-the-art cryptographic techniques such as homomorphic encryption and secure multi-party computations. However, these techniques preserve the privacy at the cost of heavy computational and communication complexities to the clients. In this paper, we propose an efficient solution to address the above challenge. The contributions of this paper are

• *Y. Rahulamathavan and M. Rajarajan are with the Information Security Group, School of Engineering and Mathematical Sciences, City University London, EC1V 0HB, London, U.K. (e-mails: yogachandran.rahulamathavan.1@city.ac.uk and R.Muttukrishnan@city.ac.uk).*

C_1 : develop a lightweight solution for FEC without using computationally intensive cryptographic techniques

C_2 : at the same time achieve classification accuracy equivalent to the accuracy of the conventional non-PP algorithm

The remainder of this paper is organized as follows: We review the related works in Section 2. Notation and mathematical tools required for our algorithm are given in Section 3. In Section 4, we describe the conventional FEC algorithm (i.e., without privacy requirements). In Section 5, we propose the efficient algorithm to achieve privacy during the classification followed by security and privacy analysis. Performance of the proposed algorithm is analyzed in Section 6. Conclusions are discussed in Section 7.

2 Related works

There are several classification algorithms developed in pattern recognition and machine learning for various applications [7]. Few of them in the literature have been redesigned for PP classification ([8]–[15], [29] and references therein). Majority of these are developed for distributed setting where different parties hold parts of the training database and securely train a common classifier without each party needing to disclose its own training data to other parties [9], [10].

The work in [9] proposed for the first time a strongly privacy-enhanced protocol for support vector machine (SVM) using cryptographic primitives whereas the authors assumed that the training data is distributed. Hence, in order to preserve the privacy they developed a protocol to perform secure kernel sharing, prediction and training using secret sharing and homomorphic encryption techniques. At the end of the training, each party will hold a share of the secret. In the testing phase, all parties collaboratively perform the classification using their shared secrets. At the end of the protocol, each party will hold the share of the predicted class label.

PP data classification algorithms suitable for client-server model were studied in [8], [11]–[15], [29]. The client-server model substantially reduces the computational and communication overhead to the client since s/he needs to interact with only one server compared to the distributed setting. The work in [8] proposes a PP face recognition algorithm using cryptographic techniques. Outsourcing the clinical decision support system to the third-party server without violating the client's privacy was studied in [15]. Similarly several plain domain classification algorithms were redesigned in [11]–[14] to perform PP classifications.

The PP algorithm for FEC algorithm was studied in [29]. The work in [29] is a variant of [8]. Even though, at high level, there are similarities between [8] and [29] in the PP computation part, there are significant differences between both the schemes. The scheme in [8] uses principal component analysis (PCA) for feature extraction and then nearest neighbor (NN) classifier. One problem that can be associated with PCA is that it is very sensitive to variations in expression, illumination, occlusion etc. In order to overcome this drawback, Belhumeur et. al proposed a technique called Fisher Linear Discriminant Analysis (FLDA) in [23] and showed that the method in [23] has lower error rates that than PCA based technique. However, FLDA tends to give undesired results if image samples in a particular class have many local means (i.e., a multimodal class). In that case FLDA cannot preserve the discriminative features of that particular class. Local Fisher discriminant analysis (LFDA) has been proposed to overcome drawbacks of FLDA [25]. Due to this, [29] considers LFDA instead of PCA.

The work in [29] achieves the privacy requirements by masking the test image and results using Paillier homomorphic encryption [16]. Homomorphic cryptosystems such as Paillier are public key cryptosystems and use large keys for encryption and decryption [16]. This involves computationally intensive operations such as exponentiation of very large numbers. On the other hand, the homomorphic encryption used for data classification schemes only perform limited number of operations in the encrypted domain (either addition or multiplication). Hence, more number of interactions between the client and server is required to obtain the result.

In contrast to all of these works, we propose a lightweight PP FEC protocol based on randomization technique. The protocol relies only on multiplication and addition. However, we show later in this paper that the new protocol achieves the privacy requirements without degrading the classification performance while improving the computational and communication complexity compared to the existing work.

3 PRELIMINARIES

We use boldface upper and lower case letters for matrices and vectors; $(.)^T$ denotes the transpose operator; $\|.\|_2$ the Euclidean norm; $\lfloor . \rceil$ the nearest integer approximation; $\otimes$ denotes the Kronecker product. The modular reduction operator is denoted by $mod.$ vec denotes vectorization operation which converts the matrix into a column vector.

3.1 Kronecker product and identities

In mathematics, the Kronecker product, denoted by $\otimes$, is an operation on two matrices of arbitrary size resulting in a block matrix. This product satisfies the following two identities [17] (notations $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$, $\mathbf{D}$, and $\mathbf{X}$ to be used to define matrices with appropriate sizes):

1. $vec(\mathbf{AXB}) = (\mathbf{B}^T \otimes \mathbf{A})vec(\mathbf{X})$

2. $(\mathbf{A} \otimes \mathbf{B})(\mathbf{C} \otimes \mathbf{D}) = (\mathbf{AC}) \otimes (\mathbf{BD})$

3.2 Information-theoretic security

A security algorithm is information-theoretically secure if its security is derived purely from information theory. The concept of information-theoretically secure communication was introduced in 1949 by American mathematician Claude Shannon, who used it to prove that the one-time pad system archives perfect security subject to the following two conditions [18]:

1. the key which randomizes the data should be random and should be used only once
2. the key length should be at least as long as the length of the data

If any algorithm randomizes its parameter and satisfies the above conditions then the parameters cannot be unmasked by an adversary even when the adversary has unlimited computing power e.g., if the message space and random space are equal to $1024-$bits then *prior* probability (probability for a particular message out of 2^{1024} possible messages) and *posterior* probability (probability of inferring/mapping a message in random domain to a message domain) are equal i.e., there is no advantage for an adversary to get higher posterior probability than prior probability.

3.3 Privacy-preserving secure two-party computation

This computation will be exploited to perform secure operation between two parties. Let us assume one party i.e., client knows a vector $\mathbf{a}$ and another party i.e., server knows a vector $\mathbf{b}$ and scalar r. Both want to interact with each other to compute $\mathbf{a}^T\mathbf{b}+r$. However, the outcome $\mathbf{a}^T\mathbf{b}+r$ must be known only to the client. To compute $r + \mathbf{a}^T\mathbf{b}$ in private, we extend the PP scalar multiplication algorithm in [19] . The complete algorithm is presented in Table 1.

Let us assume that the client's input $\mathbf{a}$ is composed of integers while the server's input $\mathbf{b}$ is composed of floating points. Since we use integer random numbers, the server converts the elements in $\mathbf{b}$ into integers by scaling and nearest integer approximation operation. This is a valid operation since we scale the vector, $\mathbf{b}$, and r by a large scalar, s, before computation (i.e., $s.r + \mathbf{a}^T\lfloor s\mathbf{b}\rceil$) and divide the outcome by the same scaling factor s (i.e., $\frac{1}{s}(s.r + \mathbf{a}^T\lfloor s\mathbf{b}\rceil) \approx r + \mathbf{a}^T\mathbf{b}$). Initially, the client adds different random values to each of the elements in $\mathbf{a}$ as shown in Steps 7 to 10 in Table 1. Now the server executes the Steps from 14 to 21 and returns the outcome to the client. In Step 21, the server adds random number $s.r.\alpha^2$ to $\sum_{i=1}^{l} D_i$.

Theorem 1: The above two-party protocol is information-theoretically secure i.e., the server cannot infer the elements in client's input vector $\mathbf{a}$ and the client will only learn the final result but not the elements in sever's input vector $\mathbf{b}$ nor r.

TABLE 1
PP Secure Two-party Computation Algorithm to Compute $r + \mathbf{a}^T\mathbf{b}$ Privately.

1.	**Input by Client:** $\mathbf{a} = [a_1, \ldots, a_l]^T$, where $l < 2^{16}$, $\{(a_i)_{\forall\ i}\} \in \mathbb{Z}_{o_1}$ with $o_1 \leq 2^{10}$ **Input by Server:** r, $\lfloor s.\mathbf{b}\rceil = [\lfloor s.b_1\rceil, \ldots, \lfloor s.b_l\rceil]^T$, where $\{s.r,\ (b_i)_{\forall\ i}\} \in \mathbb{Z}_{o_2}$ with $o_2 \leq 2^{40}$
2.	**Output:** $r + \frac{1}{s}(\lfloor s\mathbf{a}\rceil^T\mathbf{b})$ (known only to the client)
	Client first performs the following operations
3.	choose primes α, β, where $\lvert\alpha\rvert = 80$ bits and $\beta > (l.o_1.o_2 + o_2 + 1).\alpha^2$, e.g., $\lvert\beta\rvert > 226$ bits if $l = 2^{16}$
4.	set $K = 0$ and choose l positive random integers $(c_1, \ldots, c_l)$ such that $o_2.\sum_{i=1}^{l} c_i < \alpha - o_2.l$ and $\lvert c_i\rvert < 24$ bits
5.	**for** each element $b_i \in \mathbf{b}$ **do**
6.	choose random integer z_i, compute $z_i.\beta$ such that $\lvert z_i.\beta\rvert \approx 1024$ bits, and calculate $k_i = z_i.\beta - c_i$
7.	**if** $b_i > 0$ **then** compute
8.	$C_i = a_i.\alpha + c_i + z_i.\beta$, $K = K + k_i$
9.	**else if** $b_i = 0$ **then** compute
10.	$C_i = c_i + z_i.\beta$, $K = K + k_i$
11.	**end if**
12.	**end for**
13.	send $(\alpha, C_1, C_2, \ldots, C_n)$ to the server
	Now server executes the following operations
14.	**for** each element $b_i \in \mathbf{b}$ **do**
15.	**if** $\lfloor s.b_i\rceil > 0$ **then** compute
16.	$D_i = \lfloor s.b_i\rceil.\alpha.C_i$ i.e., if $a_i > 0$ $\rightarrow D_i = \lfloor s.b_i\rceil.a_i.\alpha^2 + \lfloor s.b_i\rceil.\alpha.c_i + \lfloor s.b_i\rceil.\alpha.z_i.\beta$ if $a_i = 0 \rightarrow D_i = \lfloor s.b_i\rceil.\alpha.c_i + \lfloor s.a_i\rceil.\alpha.z_i.\beta$
17.	**else if** $b_i = 0$ **then** compute
18.	$D_i = C_i$ i.e., if $a_i > 0 \rightarrow D_i = a_i.\alpha + c_i + z_i.\beta$ if $a_i = 0 \rightarrow D_i = c_i + z_i.\beta$
19.	**end if**
20.	**end for**
21.	compute $D = s.r.\alpha^2 + \sum_{i=1}^{l} D_i$ and return D to the client
	Now client performs the following operations
22.	compute $E = D + K\ mod\ \beta$
23.	**return** $\frac{E-(E\ mod\ \alpha^2)}{s.\alpha^2}$ which is equal to $r + \frac{1}{s}(\lfloor s\mathbf{b}\rceil^T\mathbf{a}) \approx \mathbf{a}^T\mathbf{b} + r$
24.	**end procedure**

Proof: To validate the security we consider both the client and server are honest-but-curious i.e., they will follow the procedures but try to learn about each other's input, intermediate values and result. Let us show that the algorithm in Table 1 is information-theoretically secure for the following two cases.

Case I–Honest-but-curious Server

In Table 1, the client randomizes his inputs $a_i,\ i = 1, \ldots, l$ by 712-bits of freshly generated random integers $z_i,\ i = 1, \ldots, l$ and sends $C_i, i = 1, \ldots, l$ to the server (see Steps 3 to 13). The server is curious to infer the clients data a_i from C_i. The server can compute the *priori* probabilities of random integer z_i and data a_i based on the underlying stochastic process as well as the *posteriori* probabilities of the various possible a_i and keys $c_i + z_i\beta$ which might have produced C_i. According to information theory, as long as the size of the client's data is less than the random integer (i.e., $\lvert c_i + z_i\beta\rvert \approx 1024$-bits in this case) and the random

integers are fresh, then the server can only compute identical priory and posteriori probabilities, hence the client's data is information-theoretically secure. In Section 5.3 we show that the size of the client's input data is always less than $1024-$bits.

Case II–Honest-but-curious Client

As shown in the Steps 14 to 21 in Table 1, the server's input data $s.b_i,\ i = 1, \ldots, l$ is multiplied with corresponding a_i and then the results were added to get $\sum_{i=1}^{l} s.b_i.a_i$ in the randomized domain. The curious client wants to infer $\sum_{i=1}^{l} s.b_i.a_i$ or the servers input from the result. The outcome is randomized by $s.r$ where r is freshly generated random value (we avoided α^2 in here for brevity). In fact the server hides $\sum_{i=1}^{l} b_i.a_i$ using $s.r$ from the curious client. As explained previously, the server can secure $\sum_{i=1}^{l} b_i.a_i$ information-theoretically if the size of the random integer r is greater than the size of the outcome $\sum_{i=1}^{l} b_i.a_i$. In Section 5.3, we show that the server can determine the size of $\sum_{i=1}^{l} b_i.a_i$ so that it can generate larger integer $s.r$ in order to achieve the information-theoretic security. □

Theorem 2: The algorithm in Table 1 is not vulnerable to overflow errors.

Proof: Let us assume that elements in vectors $\mathbf{a}$, $\lfloor s.\mathbf{b} \rceil$, and $s.r_i$ take maximum possible values. Let us denote elements in $\mathbf{a}$ can take maximum value of o_1 and elements in $\lfloor s.\mathbf{b} \rceil$ and $s.r_i$ can take maximum values of o_2. Hence, if number of elements in $\mathbf{a}$ and $\lfloor s.\mathbf{b} \rceil$ are equal to l, then output of the algorithm in Table 1 should be at most $\frac{o_2+l.o_1.o_2}{s}$. In order to verify this, let us go through the Steps 8, 16, 21, 22, and 23 in Table 1: Step 8 $\Rightarrow C_i = o_1.\alpha + c_i + z_i.\beta,\ i = 1, \ldots, l$. Step 16 $\Rightarrow D_i = o_2\alpha C_i = o_2.o_1.\alpha^2 + o_2.\alpha.c_i + o_2.\alpha.z_i.\beta,\ i = 1, \ldots, l$. Step 21 $\Rightarrow D = o_2.\alpha^2 + \sum_{i=1}^{l} D_i = o_2.\alpha^2 + l.o_1.o_2.\alpha^2 + o_2.\alpha.\sum_{i=1}^{l} c_i + o_2.\sum_{i=1}^{l} \alpha.z_i.\beta$. Since $K = \sum_{i=1}^{l}(z_i.\beta - c_i)$, and from Step 22 $\Rightarrow E = D + K \ mod\ \beta = o_2.\alpha^2 + l.o_1.o_2.\alpha^2 + o_2.\alpha.\sum_{i=1}^{l} c_i + o_2.\sum_{i=1}^{l} \alpha.z_i.\beta + \sum_{i=1}^{l}(z_i.\beta - c_i)\ mod\ \beta = o_2.\alpha^2 + l.o_1.o_2.\alpha^2 + (o_2.\alpha - 1).\sum_{i=1}^{l} c_i$. There is no further modulo reduction since $\beta > (l.o_1.o_2 + o_2 + 1).\alpha^2$ (Step 3), $o_2.\sum_{i=1}^{n} c_i < \alpha - o_2.l$ (Step 4), and

$$\begin{aligned} E = &\ o_2.\alpha^2 + l.o_1.o_2.\alpha^2 + (o_2.\alpha - 1).\sum_{i=1}^{l} c_i, \\ < &\ o_2.\alpha^2 + l.o_1.o_2.\alpha^2 + o_2.\alpha.\sum_{i=1}^{l} c_i \\ < &\ o_2.\alpha^2 + l.o_1.o_2.\alpha^2 + (\alpha^2 - \alpha.o_2.l) \\ < &\ o_2.\alpha^2 + l.o_1.o_2.\alpha^2 + \alpha^2 \\ = &\ l.o_1.o_2.\alpha^2 + (o_2 + 1).\alpha^2 < \beta. \end{aligned}$$

From Step 23, $E\ mod\ \alpha^2 = o_2.\alpha^2 + l.o_1.o_2.\alpha^2 + (o_2.\alpha - 1).\sum_{i=1}^{l} c_i\ mod\ \alpha^2 = (o_2.\alpha - 1).\sum_{i=1}^{l} c_i$. There is no further modulo reduction since $o_m.\sum_{i=1}^{n} c_i < \alpha - o_m.l$ (Step 4) and $E\ mod\ \alpha^2 = (o_2.\alpha - 1).\sum_{i=1}^{l} c_i < o_m.\alpha.\sum_{i=1}^{l} c_i < \alpha^2$. Hence, from Step 23, $\frac{E-(E\ mod\ \alpha^2)}{s.\alpha^2} = \frac{o_2.\alpha^2 + l.o_1.o_2.\alpha^2}{s.\alpha^2}$. The two-party computation algorithm always output correct result. □

4 Facial Expression Classification

A FEC involves two different phases: facial feature extraction and classification. Facial feature extraction involves extracting features of facial images; the resulting feature vectors can then be used to project the facial image from the higher dimensional image space into a lower dimensional feature space while preserving the discriminative features. Discriminative features separate the facial images of one class from facial images of other classes in the lower dimensional feature space [20]. Better separation among classes in the lower dimensional space leads to higher classification rate.

PCA is one of the most widely used feature extraction (i.e., dimensionality reduction) methods in image recognition and compression [20]–[22]. PCA aims to obtain a set of mutually orthogonal bases that describe the global information of the data points in terms of variance. The drawback in PCA is that the scatter being maximized is due not only to the between-class scatter that is useful for classification, but also to the within-class scatter. Maximization of within-class scatter includes unwanted information to the classification process [23].

The Fisher linear discriminant analysis (FLDA) [7], [23], [24] has been proposed as alternative method to overcome the drawbacks of PCA. In fact FLDA preserves the discriminative features of images while reducing dimension on the image space. FLDA obtains the transformation matrix by maximizing the between-class scatter matrix while minimizing the within-class scatter matrix. However, it tends to give undesired results if image samples in a particular class have many local means (i.e., a multimodal class). In that case FLDA cannot preserve the discriminative features of that particular class. Local Fisher discriminant analysis (LFDA) has been proposed to overcome drawbacks of FLDA [25]–[28]. Within this context, we consider LFDA in the paper for feature extraction.

4.1 Local Fisher Discriminant Analysis

Suppose we have C number of expression classes, let n_c be the number of facial images for cth class, where $c \in \{1, \ldots, C\}$ denotes the class index (refer Table 2 for more notations). The total number of facial images can thus be denoted as $N = \sum_{c=1}^{C} n_c$. Let us assume that each facial image is a real valued grayscale image and can be represented as a matrix, where each element corresponds to the pixel value of a point within the image. A two-dimensional facial image matrix can be converted to a one-dimensional vector by stacking each column (or row) of the matrix

TABLE 2
Variables and their descriptions.

Variable	Description
C	Number of different classes in the training database
n_c	Number of images in the c^{th} class
N	Total number of images in the training database
n	Number of pixels per image
$\widetilde{\mathbf{x}_i}$	ith facial image (original)
$\bar{\mathbf{x}}$	Mean of the training database
$\mathbf{x}_i$	Difference training data sample of $\widetilde{\mathbf{x}_i}$ i.e., $\mathbf{x}_i = \widetilde{\mathbf{x}_i} - \bar{\mathbf{x}}$
$\mathbf{u}_i$	This will be used to project the images onto lower dimensional space
$\mathbf{y}_i$	Projected training images of $\mathbf{x}_i$ in lower dimension
$\mathbf{t}$	Difference test image i.e., $\mathbf{t} = \tilde{\mathbf{t}} - \bar{\mathbf{x}}$
$\mathbf{w}$	Low dimensional vector corresponding to $\mathbf{t}$
d_i	Euclidean distance between $\mathbf{w}$ and $\mathbf{y}_i$

into a long vector. Denote $\widetilde{\mathbf{x}_i} \in \mathbb{Z}^{n\times 1}$ as a vector representation of the ith facial image.

Let us start with a training set of facial images $\widetilde{\mathbf{x}_i} = [\widetilde{x_{1,i}}, \ldots, \widetilde{x_{n,i}}]^T \in \mathbb{Z}^{n\times 1}$, $i = 1, \ldots, N$. A classifier can be trained using the training data samples to classify an unlabeled test sample. To do this, let us denote the difference training data samples as $\mathbf{x}_i \in \mathbb{R}^{n\times 1}$, $i = 1, \ldots, N$ where,

$$\mathbf{x}_i = \tilde{\mathbf{x}}_i - \bar{\mathbf{x}}, \forall i, \tag{1}$$

where $\bar{\mathbf{x}} = \frac{1}{N}\sum_{i=1}^{N} \tilde{\mathbf{x}}_i$ denotes the mean of the training data samples.

Let us denote the projected image in the lower dimensional feature space corresponding to the image $\mathbf{x}_i = [x_{1,i}, \ldots, x_{n,i}]^T \in \mathbb{R}^{n\times 1}$ as $\mathbf{y}_i = [y_{1,i}, \ldots, y_{m,i}]^T \in \mathbb{R}^{m\times 1}$, where $m \ll n$. Hence, $\mathbf{y}_i$ can be obtained by the following linear projection using the LFDA transformation matrix $\mathbf{U}$:

$$\mathbf{y}_i = \mathbf{U}^T\mathbf{x}_i, \ i = 1, \ldots, N, \tag{2}$$

$$y_{k,i} = \mathbf{u}_k^T\mathbf{x}_i, \ k = 1, \ldots, m. \tag{3}$$

In LFDA, the transformation matrix $\mathbf{U}$ maximizes the local-between-class-scatter-matrix $\mathbf{S_B}$ while minimizing the local-within-class-scatter-matrix $\mathbf{S_W}$:

$$\mathbf{U} = \underset{\mathbf{V}}{\operatorname{argmax}} \frac{|\ \mathbf{V}^T\mathbf{S_B}\mathbf{V}\ |}{|\ \mathbf{V}^T\mathbf{S_W}\mathbf{V}\ |}, \tag{4}$$

where, $\mathbf{V}$ represents the possible transformation matrices i.e., the optimal $\mathbf{V}$ which maximizes the argument is equal to $\mathbf{U}$,

$$\mathbf{S_B} = \frac{1}{2}\sum_{i,j=1}^{N} B_{i,j}(\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T, \tag{5}$$

$$\mathbf{S_W} = \frac{1}{2}\sum_{i,j=1}^{N} W_{i,j}(\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T, \tag{6}$$

where $B_{i,j}$ and $W_{i,j}$ are tuning parameters for LFDA. Interesting readers can refer [25], [29] for more details.

4.2 Classification

In this paper, we consider 1-NN [31]. However, our algorithm can be extended to k-NN and Hamming distance based classifiers since these techniques and 1-NN are mathematically similar. 1-NN classifier predicts the matching class of the test image based on the closest training image. Squared Euclidean distance calculation can be used to obtain the distances between the test image and training images (i.e., in the lower dimension). In order to explain the classification phase, let us denote a test image as $\tilde{\mathbf{t}} = [\widetilde{t_1}, \ldots, \widetilde{t_n}]^T \in \mathbb{Z}^{n\times 1}$ and difference test image as $\mathbf{t} = [t_1, \ldots, t_n]^T \in \mathbb{R}^{n\times 1}$, where

$$\mathbf{t} = \tilde{\mathbf{t}} - \bar{\mathbf{x}}. \tag{7}$$

Denote the low dimensional vector corresponding to $\mathbf{t}$ as $\mathbf{w} = [w_1, \ldots, w_m]^T \in \mathbb{R}^{m\times 1}$ where $m \ll n$. Now $\mathbf{t}$ is projected by the projection matrix as

$$\mathbf{w} = \mathbf{U}^T\mathbf{t}, \tag{8}$$

$$w_k = \mathbf{u}_k^T\mathbf{t}, \ k = 1, \ldots, m. \tag{9}$$

The squared Euclidean distance, d_i, between $\mathbf{w}$ and $\mathbf{y}_i$, $i = 1, \ldots, N$ is

$$\begin{aligned} d_i = \ & \|(\mathbf{w} - \mathbf{y}_i)\|_2^2 = \|(\mathbf{U}^T\mathbf{t} - \mathbf{U}^T\mathbf{x}_i)\|_2^2, \\ = \ & \sum_{k=1}^{m}\left(\mathbf{u}_k^T\mathbf{t}\mathbf{t}^T\mathbf{u}_k - 2\mathbf{u}_k^T\mathbf{t}\mathbf{u}_k^T\mathbf{x}_i + \mathbf{u}_k^T\mathbf{x}_i\mathbf{x}_i^T\mathbf{u}_k\right). \end{aligned} \tag{10}$$

The decision rule of the 1-NN classifier is that the training image $\mathbf{x}_*$ is said to have same expression of the test image if

$$d_* = \min\{d_1, \ldots, d_N\}, \tag{11}$$

where d_* is smaller than a given threshold T. So far we have discussed the traditional approach in which both the training phase (i.e., feature extraction) and the testing phase (i.e., classification) are carried out by the same party. In the next section, we propose an efficient protocol where the training phase is performed by one-party (i.e., server) whereas the testing phase is collaboratively carried out between two-parties (i.e., between the client and server).

5 Privacy-preserving Facial Expression Classification

In this section, we extend the algorithm explained in Section 4 to perform PP FEC. The new algorithm satisfies the following three requirements:

R_1 : without using highly computationally intensive public-key homomorphic encryption schemes such as Paillier cryptographic system [16]

R_2 : hide the client's input data sample and the classification result from the server

R_3 : hide the server side classification parameters such as feature vectors and mean of the database from the client

In order to explain the new algorithm, let us split the testing phase of the traditional approach in Section 4 into four steps as follows:

S_1 : obtain difference test image i.e., (7)

S_2 : projecting the difference test image onto lower dimension i.e., (8)

S_3 : Euclidean distance calculation i.e., (10)

S_4 : minimum distance calculation to match the test image to a known class i.e., (11)

TABLE 3
Variables and their description. We use ✓ (X) to denote if the corresponding variable is known (unknown) to one party.

Variables	Known only to the Client	Known only to the Server
$\widetilde{\mathbf{x}_i}$, $\bar{\mathbf{x}}$, $\mathbf{x}_i$, $\mathbf{y}_i$, $\mathbf{u}_i$	X	✓ (see Table 2)
$\mathbf{t}$	✓ (see Table 2)	X
$\tilde{\boldsymbol{\eta}}$	✓ (Noise vector, same dimension as $\tilde{\mathbf{t}}$)	✓
$\boldsymbol{\eta}$	X	✓ (Difference noise vector)
$\boldsymbol{\gamma}$	✓ (difference vector)	X
$\bar{\mathbf{w}}$	X	✓ (Lower dimensional vector corresponding to $\boldsymbol{\eta}$)
$\bar{d}_i$	X	✓ (Euclidean distance between between $\bar{\mathbf{w}}$ and $\mathbf{y}_i$
$m_i = \bar{d}_i - d_i$	X	X
r_i	X	✓ (Random integer generated by the server to hide $\bar{d}_i$)
$\widetilde{\bar{d}_i} = \bar{d}_i + r_i$	✓	✓
$r^k_{i,1}$, $r^k_{i,2}$, $r^k_{i,3}$, $r^k_{i,4}$	X	✓ (Random integers generated by the server to use during the secure two-party computations)

In the following subsections, we explain how each of these steps can be computed without violating the above three requirements (R_1 – R_3). Since our method preserves the privacy, we add the term "private" to those four steps in order to distinguish our method from the traditional approach. Definitions for the variables used in the following subsections are provided in Table 3.

5.1 Obtain the difference test image in private

Initially, the difference test image needs to be obtained for the classification. However, the client cannot send the test image, $\tilde{\mathbf{t}}$, due to the privacy concerns. Hence, the client only sends noise vector, $\tilde{\boldsymbol{\eta}} \in \mathbb{Z}^{n\times 1}$, with same dimension as test image. Since server receives only the noise vector, it cannot get any information about the test image vector. As shown in (7), the server obtain the difference noise vector, $\tilde{\boldsymbol{\eta}}$, instead of difference noise vector, $\boldsymbol{\eta}$, as follows:

$$\boldsymbol{\eta} = \tilde{\boldsymbol{\eta}} - \bar{\mathbf{x}}. \tag{12}$$

However, only the client knows the difference between the test image and the noise image. Let us denote the difference as $\boldsymbol{\gamma}$ where

$$\boldsymbol{\gamma} = \tilde{\boldsymbol{\eta}} - \tilde{\mathbf{t}} = \boldsymbol{\eta} - \mathbf{t} \in \mathbb{Z}^{n\times 1}. \tag{13}$$

5.2 Projecting the difference noise vector onto the lower dimension space in private

As shown in (8), the server projects the difference noise vector (instead of difference test image) and obtains lower dimension vector, $\bar{\mathbf{w}} = [\bar{w_1}, \bar{w_2}, \ldots, \bar{w_m}] \in \mathbb{R}^{m\times 1}$, as follows:

$$\bar{\mathbf{w}} = \mathbf{U}^T\boldsymbol{\eta}, \tag{14}$$

$$\bar{w_k} = \mathbf{u}_k^T\boldsymbol{\eta},\ k = 1, \ldots, m. \tag{15}$$

However, using (12), (13), $\mathbf{t} = \tilde{\mathbf{t}} - \bar{\mathbf{x}}$, (9) and (15), we can derive $\bar{w_k}$ in terms of w_k, $k = 1, \ldots, m$ as follows:

$$\begin{aligned} \bar{w_k} &= \mathbf{u}_k^T(\tilde{\boldsymbol{\eta}} - \bar{\mathbf{x}}) = \mathbf{u}_k^T[(\tilde{\mathbf{t}} + \boldsymbol{\gamma}) - \bar{\mathbf{x}}], \\ &= \mathbf{u}_k^T[(\tilde{\mathbf{t}} - \bar{\mathbf{x}}) + \boldsymbol{\gamma}] = \mathbf{u}_k^T(\mathbf{t} + \boldsymbol{\gamma}), \\ &= w_k + \mathbf{u}_k^T\boldsymbol{\gamma},\ k = 1, \ldots, m. \end{aligned} \tag{16}$$

The scalar $\mathbf{u}_k^T\boldsymbol{\gamma}$ in (16) is unknown to the server. Hence, the server uses only the $\bar{w_k},\ k = 1, \ldots, m$ obtained in (15) for the distance calculation step instead of $w_k,\ k = 1, \ldots, m$. We show in Section 5.5 that the server cannot infer w_k from (16).

5.3 Euclidean distance calculation in private

Since the server has only computed $\bar{\mathbf{w}}$, let us denote the Euclidean distance between $\bar{\mathbf{w}}$ and the low dimensional training image $\mathbf{y}_i$ as $\bar{d}_i,\ i = 1, \ldots, N$. Similar to (10), we can compute the Euclidean distances $\bar{d}_i,\ i = 1, \ldots, N$ as

$$\begin{aligned} \bar{d}_i &= \|(\bar{\mathbf{w}} - \mathbf{y}_i)\|_2^2 = \|(\mathbf{U}^T\boldsymbol{\eta} - \mathbf{U}^T\mathbf{x}_i)\|_2^2, \\ &= \sum_{k=1}^{m} \left(\mathbf{u}_k^T\boldsymbol{\eta}\boldsymbol{\eta}^T\mathbf{u}_k - 2\mathbf{u}_k^T\boldsymbol{\eta}\mathbf{u}_k^T\mathbf{x}_i + \mathbf{u}_k^T\mathbf{x}_i\mathbf{x}_i^T\mathbf{u}_k\right). \end{aligned} \tag{17}$$

Since, $\boldsymbol{\eta} = \mathbf{t} + \boldsymbol{\gamma}$, we can write $\bar{d}_i$ in terms of $d_i,\ i = 1, \ldots, N$ as

$$\begin{aligned} \bar{d}_i &= \sum_{k=1}^{m} \big[\mathbf{u}_k^T(\mathbf{t} + \boldsymbol{\gamma})(\mathbf{t} + \boldsymbol{\gamma})^T\mathbf{u}_k - 2\mathbf{u}_k^T(\mathbf{t} + \boldsymbol{\gamma})\mathbf{u}_k^T\mathbf{x}_i \\ &\qquad + \mathbf{u}_k^T\mathbf{x}_i\mathbf{x}_i^T\mathbf{u}_k\big], \\ &= d_i + \sum_{k=1}^{m} \left[\mathbf{u}_k^T\left(2\mathbf{t}\boldsymbol{\gamma}^T + \boldsymbol{\gamma}\boldsymbol{\gamma}^T - 2\boldsymbol{\gamma}\mathbf{x}_i^T\right)\mathbf{u}_k\right]. \end{aligned} \tag{18}$$

It is shown in Section 5.5 that the server cannot infer d_i from (18). In the next subsection, we elaborate how the distances obtained in (18) can be used to obtain the matching training image.

5.4 Minimum distance calculation to match the test image to a known class in private

Let us denote the difference between d_i and $\bar{d}_i$ in (18) as $m_i,\ i = 1,\ldots,N$ (i.e., $m_i = \sum_{k=1}^{m}\left[\mathbf{u}_k^T(2\mathbf{t}\boldsymbol{\gamma}^T + \boldsymbol{\gamma}\boldsymbol{\gamma}^T - 2\boldsymbol{\gamma}\mathbf{x}_i^T)\mathbf{u}_k\right]$). Let us rewrite (18) as follows:

$$\bar{d}_i = d_i + m_i,\ i = 1,\ldots,N. \tag{19}$$

In order to find the minimum distance, the server needs to remove m_i from $\bar{d}_i$. Since the server does not know the vectors $\boldsymbol{\gamma}$ and $\mathbf{t}$, it is infeasible for the server to obtain a true distance value d_i from $\bar{d}_i,\ i = 1,\ldots,N$. Hence, the server cannot find the matching training image corresponding to the test image using (19). In order to obtain the matching image, the server needs to interact with the client by sending all the $\bar{d}_i,\ i = 1,\ldots,N$. Before sending $\bar{d}_i,\ i = 1,\ldots,N$, the server generates random value r_i for each $\bar{d}_i$ and send $\widetilde{d}_i$ to the client where

$$\widetilde{d}_i = \bar{d}_i + r_i = d_i + m_i + r_i,\ i = 1,\ldots,N. \tag{20}$$

Now the client must interact with the server to compute $m_i + r_i,\ i = 1,\ldots,N$. Using $m_i + r_i,\ i = 1,\ldots,N$, the client can compute the actual Euclidean distances $d_i,\ i = 1,\ldots,N$ as follows:

$$d_i = \widetilde{d}_i - (m_i + r_i),\ i = 1,\ldots,N, \tag{21}$$

where

$$m_i + r_i = \sum_{k=1}^{m}\mathbf{u}_k^T(2\mathbf{t}\boldsymbol{\gamma}^T + \boldsymbol{\gamma}\boldsymbol{\gamma}^T - 2\boldsymbol{\gamma}\mathbf{x}_i^T)\mathbf{u}_k + r_i.$$

Since $\mathbf{t} = \tilde{\mathbf{t}} - \bar{\mathbf{x}}$,

$$\begin{aligned} m_i + r_i &= \sum_{k=1}^{m}\mathbf{u}_k^T[2(\tilde{\mathbf{t}} - \bar{\mathbf{x}})\boldsymbol{\gamma}^T + \boldsymbol{\gamma}\boldsymbol{\gamma}^T - 2\boldsymbol{\gamma}\mathbf{x}_i^T]\mathbf{u}_k + r_i, \\ &= \sum_{k=1}^{m}\mathbf{u}_k^T(2\tilde{\mathbf{t}}\boldsymbol{\gamma}^T - 2\bar{\mathbf{x}}\boldsymbol{\gamma}^T + \boldsymbol{\gamma}\boldsymbol{\gamma}^T - 2\boldsymbol{\gamma}\mathbf{x}_i^T)\mathbf{u}_k + r_i, \\ &= \sum_{k=1}^{m}\mathbf{u}_k^T[(2\tilde{\mathbf{t}} + \boldsymbol{\gamma})\boldsymbol{\gamma}^T]\mathbf{u}_k \\ &\quad - \sum_{k=1}^{m}\mathbf{u}_k^T(2\bar{\mathbf{x}}\boldsymbol{\gamma}^T + 2\boldsymbol{\gamma}\mathbf{x}_i^T)\mathbf{u}_k + r_i. \end{aligned} \tag{22}$$

It should be noted that $m_i,\ i = 1,\ldots,N$ should only be known to the client. Otherwise, if they are known to the server, then the server can compute the actual Euclidean distances between the test and training images, and eventually the server can obtain the expression corresponding to the test image which violates the client's privacy. In (22), the vectors $\tilde{\mathbf{t}}$ and $\boldsymbol{\gamma}$ are only known to the client and the vectors $\mathbf{u}_k$, $\mathbf{x}_i$, $\bar{\mathbf{x}}$ and the random scalar r_i are known only to the server. The vectors and scalars known to the server and client are coupled with each other in (22) and it requires large number of interactions between the client and server to privately compute $m_i + r_i$. If we reorder (22) such that, the variables known to the server to be on one side while the variables known to the client to other side then the algorithm in Table 1 can be used to compute (22).

5.4.1 Solve (22) using the Algorithm in Table 1

In order to exploit the algorithm in Table 1, we must rearrange variables in (22). In order to do this, let us now process the first term in (22) (i.e., $\sum_{k=1}^{m}\mathbf{u}_k^T[(2\tilde{\mathbf{t}} + \boldsymbol{\gamma})\boldsymbol{\gamma}^T]\mathbf{u}_k$) using Kronecker identities as follows:

$$\begin{aligned} &\mathbf{u}_k^T[(2\tilde{\mathbf{t}} + \boldsymbol{\gamma})\boldsymbol{\gamma}^T]\mathbf{u}_k = vec\left\{\mathbf{u}_k^T[(2\tilde{\mathbf{t}} + \boldsymbol{\gamma})\boldsymbol{\gamma}^T]\mathbf{u}_k\right\}, \\ &= (\mathbf{u}_k^T \otimes \mathbf{u}_k^T)[\boldsymbol{\gamma} \otimes (2\tilde{\mathbf{t}} + \boldsymbol{\gamma})] = (\mathbf{u}_k^T\boldsymbol{\gamma}) \otimes \left[\mathbf{u}_k^T(2\tilde{\mathbf{t}} + \boldsymbol{\gamma})\right], \\ &= (\mathbf{u}_k^T\boldsymbol{\gamma}) \times \left[\mathbf{u}_k^T(2\tilde{\mathbf{t}} + \boldsymbol{\gamma})\right]. \end{aligned} \tag{23}$$

Let us assume that the server generates random integers $r_{i,1}^k$, $r_{i,2}^k$, $r_{i,3}^k$, and $r_{i,4}^k$, $\forall k$ to randomize the scalar product output. If we incorporate these random variables within (23) then (23) becomes equal to (24) (shown in the top of the next page).

In (24), the server knows $\mathbf{u}_k$, $r_{i,1}^k$, $r_{i,2}^k$, $r_{i,3}^k$, and $r_{i,4}^k$ while the client knows $\boldsymbol{\gamma}$ and $2\tilde{\mathbf{t}} + \boldsymbol{\gamma}$. Hence, the client and server collaboratively compute $\left[\mathbf{u}_k^T\boldsymbol{\gamma} + r_{i,1}^k\right]$,$\left[\mathbf{u}_k^T(2\tilde{\mathbf{t}} + \boldsymbol{\gamma}) + r_{i,2}^k\right]$, $\left[\left(r_{i,2}^k\mathbf{u}_k\right)^T\boldsymbol{\gamma} + r_{i,3}^k\right]$, and $\left[\left(r_{i,1}^k\mathbf{u}_k\right)^T(2\tilde{\mathbf{t}} + \boldsymbol{\gamma}) + r_{i,4}^k\right]$ using the algorithm in Table 1. In order to balance the equation, the server subtracts $\left(r_{i,1}^k r_{i,2}^k - r_{i,3}^k - r_{i,4}^k\right)$ in (24) in the next computation. Now let us process the remaining terms in (22) (i.e., $-\sum_{k=1}^{m}\mathbf{u}_k^T(2\bar{\mathbf{x}}\boldsymbol{\gamma}^T + 2\boldsymbol{\gamma}\mathbf{x}_i^T)\mathbf{u}_k + r_i$) as follows: [second term in (22)-$\left(r_{i,1}^k r_{i,2}^k - r_{i,3}^k - r_{i,4}^k\right)$] = $-\sum_{k=1}^{m}\mathbf{u}_k^T(2\bar{\mathbf{x}}\boldsymbol{\gamma}^T + 2\boldsymbol{\gamma}\mathbf{x}_i^T)\mathbf{u}_k + r_i - \left(r_{i,1}^k r_{i,2}^k - r_{i,3}^k - r_{i,4}^k\right)$ can be transformed into (25) using Kronecker identities (shown in the top of the next page).

In (25), the server knows $2\left[(\mathbf{u}_k^T \otimes \mathbf{u}_k^T\bar{\mathbf{x}}) + (\mathbf{u}_k^T\mathbf{x}_i \otimes \mathbf{u}_k^T)\right]$ and $r_i + (r_{i,1}^k r_{i,2}^k - r_{i,3}^k - r_{i,4}^k)$ and the clients knows $\boldsymbol{\gamma}$. Hence, the algorithm in Table 1 can be exploited by both the client and server in order to compute the $[-\sum_{k=1}^{m}\mathbf{u}_k^T(2\bar{\mathbf{x}}\boldsymbol{\gamma}^T + 2\boldsymbol{\gamma}\mathbf{x}_i^T)\mathbf{u}_k + r_i - \left(r_{i,1}^k r_{i,2}^k - r_{i,3}^k - r_{i,4}^k\right)]$. Hence, using (24), (25), and Table 1, the client can obtain $(m_i + r_i)$ $\forall i$. This will enable the client to compute the true Euclidean distances between the test image and the training images using (21). Hence, the client can find the smallest Euclidean distance using (11). Only task now left is to find the expression of the training image corresponding to the smallest Euclidean distance. This task must be achieved without leaking the index of the training image corresponding to the smallest Euclidean distance to the server.

$$(\mathbf{u}_k^T\boldsymbol{\gamma})\times\Big[\mathbf{u}_k^T(2\tilde{\mathbf{t}}+\boldsymbol{\gamma})\Big]+\Big(r_{i,1}^k r_{i,2}^k - r_{i,3}^k - r_{i,4}^k\Big) = \Big[\mathbf{u}_k^T\boldsymbol{\gamma}+r_{i,1}^k\Big]\times\Big[\mathbf{u}_k^T(2\tilde{\mathbf{t}}+\boldsymbol{\gamma})+r_{i,2}^k\Big]-\Big[\Big(r_{i,2}^k\mathbf{u}_k\Big)^T\boldsymbol{\gamma}+r_{i,3}^k\Big]-\Big[\Big(r_{i,1}^k\mathbf{u}_k\Big)^T(2\tilde{\mathbf{t}}+\boldsymbol{\gamma})+r_{i,4}^k\Big]. \quad (24)$$

$$\begin{aligned}-\mathbf{u}_k^T(2\bar{\mathbf{x}}\boldsymbol{\gamma}^T+2\boldsymbol{\gamma}\mathbf{x}_i^T)\mathbf{u}_k + r_i - (r_{i,1}^k r_{i,2}^k - r_{i,3}^k - r_{i,4}^k) &= -2vec(\mathbf{u}_k^T\bar{\mathbf{x}}\boldsymbol{\gamma}^T\mathbf{u}_k) - 2vec(\mathbf{u}_k^T\boldsymbol{\gamma}\mathbf{x}_i^T\mathbf{u}_k) + r_i - (r_{i,1}^k r_{i,2}^k - r_{i,3}^k - r_{i,4}^k),\\ &= -2\Big[(\mathbf{u}_k^T\otimes\mathbf{u}_k^T\bar{\mathbf{x}})+(\mathbf{u}_k^T\mathbf{x}_i\otimes\mathbf{u}_k^T)\Big]\,vec(\boldsymbol{\gamma}) + r_i - (r_{i,1}^k r_{i,2}^k - r_{i,3}^k - r_{i,4}^k). \quad (25)\end{aligned}$$

5.4.2 Privacy-preserving Expression Finding

Let us define a binary vector $\mathbf{d}_b \in \{0,1\}^{N\times 1}$. If nth Euclidean distance is the smallest distance then, the client generates a binary vector $\mathbf{d}_b$ by setting nth element to 1 while setting all other elements to 0. Let us denote the expression of nth training image as id_n and define another vector called expression vector as $\mathbf{d}_{exp} = [exp_1,\ exp_2\ , \ldots, exp_N]^T$. Client must keep the binary vector $\mathbf{d}_b$ away from the server in order to protect the privacy of the test image. However, the client could exploit the algorithm in Table 1 to obtain the expression of the test image without revealing the $\mathbf{d}_b$ as follows: The expression of the training image which is corresponding to the smallest Euclidean distance (i.e., lets assume nth training image) could be obtained by computing $\mathbf{d}_b^T\mathbf{d}_{exp}$ (i.e., $[0\,0\,\ldots 0\,1\,0\,\ldots\,0].[exp_1\,\ldots, exp_n, exp_{n+1}, \ldots, exp_N]^T = exp_n$). It should be noted that the algorithm in Table 1 but with different inputs can be used to obtain the correct expression of the matching training image. If the client feeds the binary vector $\mathbf{d}_b$ instead of $\mathbf{a}$ and if the server feeds the expression vector $\mathbf{d}_{exp}$ instead of $s.\mathbf{b}$ and 0 for r to the algorithm in Table 1, then the output of the algorithm should be equivalent to the expression of the matching training image.

5.5 Privacy Analysis

In this section, we analyse whether our algorithm is vulnerable to any privacy leakage. Our algorithm is based on two-party computation and the only possibility that the privacy leakage can happen is during the interaction between two-parties. We use the following O. Goldreich's privacy definition to proof that our method doesn't leak any unintended information to client or server:

Privacy definition for the secure two-party computation: A secure two-party protocol should not reveal more information to a semi-honest party than the information that can be induced by looking at that partys input and output. The formal proof of the definition can be found in [40].

Let us verify whether the proposed two-party computation satisfies the privacy definition. As described in Section 5, the proposed algorithm is composed of four sub-algorithms. In the following we show what are the inputs and outputs to and from the client and server, respectively. This will clearly highlight what is already known to client and server. Hence if we can prove that nothing else can be inferred other than the known input and output with higher posterior probability than prior probability then the proposed algorithm satisfies the privacy definition.

The ultimate aim for client is to keep the test image and the classification result away from the server while the server wants to keep the classification parameters away from the client. Initially (Section 5.1, Section 5.2, and Section 5.3), the client was just sending the noise vector $\tilde{\boldsymbol{\eta}} \in \mathbb{Z}^{n\times 1}$ to the server instead the true test image $\tilde{\mathbf{t}} \in \mathbb{Z}^{n\times 1}$. From these inputs, the server only know the size of the test image. This is not a privacy leakage since the server knows the size of the images when training the classifier.

In return the server sends the randomized Euclidean distances back to the client. As shown in Section 5.4, the server hides the Euclidean distance $\bar{d}_i$ by random integer r_i where $|r_i| > |\bar{d}_i|$ in order to achieve information-theoretic security. The upper bound of $|\bar{d}_i|$ for an 256×256 image is $256\times 256\times(2^8)^2 < 2^{48}$, hence $|r_i| = 48$-bits is sufficiently enough to achieve the information-theoretic security.

In Section 5.4.1, the client and server interact with each other using the algorithm in Table 1 in order to compute the true Euclidean distances. The clients inputs are $\boldsymbol{\gamma}$ and $(2\tilde{\mathbf{t}}+\boldsymbol{\gamma})$. As proved in Theorem 2, information-theoretic security can be achieved when the client's input data is less than 1024-bits. Hence, the server cannot learn any unindented knowledge.

Similarly for the server, if $|r_{i,1}^k| > |\mathbf{u}_k^T\boldsymbol{\gamma}|\ \forall i,k$, $|r_{i,2}^k| > |\mathbf{u}_k^T(2\tilde{\mathbf{t}}+\boldsymbol{\gamma})|\ \forall i,k$, $|r_{i,3}^k| > |\left(r_{i,2}^k\mathbf{u}_k\right)^T\boldsymbol{\gamma}|\ \forall i,k$, $|r_{i,4}^k| > |\left(r_{i,1}^k\mathbf{u}_k\right)^T(2\tilde{\mathbf{t}}+\boldsymbol{\gamma})|\ \forall i,k$, and $|r_{i,1}^k r_{i,2}^k - r_{i,3}^k - r_{i,4}^k| > |-2\left[(\mathbf{u}_k^T\otimes\mathbf{u}_k^T\bar{\mathbf{x}})+(\mathbf{u}_k^T\mathbf{x}_i\otimes\mathbf{u}_k^T)\right]vec(\boldsymbol{\gamma}),\ \forall i,k|$ then the client cannot learn any additional knowledge from individual scalar products. Below we show that client and sever knows the required size of random number to randomize the inputs.

The client's inputs are $\boldsymbol{\gamma}$ and $2\tilde{\mathbf{t}}+\boldsymbol{\gamma}$ where $|\boldsymbol{\gamma}| = 8$-bits and $|2\tilde{\mathbf{t}}+\boldsymbol{\gamma}| = 10$-bits. Since the client's inputs i.e., $|\boldsymbol{\gamma}\alpha| = 88-$bits and $|(2\tilde{\mathbf{t}}+\boldsymbol{\gamma})\alpha| = 90-$bits, are substantially smaller than 1024-bits, the clients data is information-theoretically secure. Since the server knows the size of the clients input data as well as the sizes of the elements in $\mathbf{u}_k$, and $\bar{\mathbf{x}}$, the server can compute the required sizes of the random integers $r_{i,1}^k$, $r_{i,2}^k$, $r_{i,3}^k$, and $r_{i,4}^k$ to information-theoretically secure its parameters. Based on the experiment we obtained that $|r_{i,1}^k| = 45 > |\mathbf{u}_k^T\boldsymbol{\gamma}|\ \forall i,k$, $|r_{i,2}^k| = 45 > |\mathbf{u}_k^T(2\tilde{\mathbf{t}}+\boldsymbol{\gamma})|\ \forall i,k$,

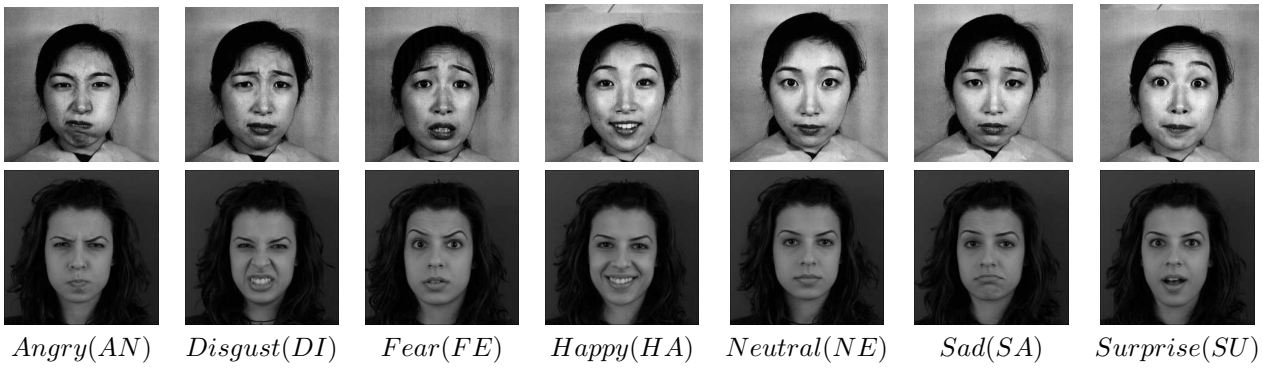

Fig. 1. Example FaE images from JAFFE (1st row) and MUG (2nd row) databases. The 3rd row depicts the expression classes.

$|r^k_{i,3}| = 100 > |\left(r^k_{i,2}\mathbf{u}_k\right)^T\boldsymbol{\gamma}| \ \forall i,k,\ |r^k_{i,4}| = 100 > |\left(r^k_{i,1}\mathbf{u}_k\right)^T(2\tilde{\mathbf{t}}+\boldsymbol{\gamma})| \ \forall i,k,$ and $|r^k_{i,1}r^k_{i,2} - r^k_{i,3} - r^k_{i,4}| > |-2\left[(\mathbf{u}^T_k \otimes \mathbf{u}^T_k\bar{\mathbf{x}}) + (\mathbf{u}^T_k\mathbf{x}_i \otimes \mathbf{u}^T_k)\right] vec(\boldsymbol{\gamma}),\ \forall i,k|$.

In Section 5.4.2, the client and server interact with each other using the algorithm in Table 1 in order to identify the matching sample. The size of the client's input data is $1-$bit which is smaller than the required $1024-$bits of input data, hence the data is information-theoretically secure. Since the server knows the sizes of the client's input data, it can compute the appropriate size of random integers in order to protect the expression vector from the client.

At high level, the client's inputs are element wise randomized before sending them to the server, hence, according to information theory, the client's inputs are secure. At the end client obtain the Euclidean distance between client's test image and the server's training image. However, the client cannot infer any additional information compromise the server's classification parameters from the Euclidean distance for the following reasons: Euclidean distance is just a positive scalar which was obtained by from feature vectors. From (9) and (10), it is obvious that the scalar value is an aggregated value obtained from large number of variables. Inferring those variables (classification parameters) from a scalar is impossible. Even if the adversary send different inputs, due to the inherent properties of (9) and (10), the outcome will always be a scalar version of previous values.

6 Performance Analysis

The proposed FEC method was evaluated using two FaE databases: JAFFE [32] and MUG [33]. Fig. 1 shows sample images and expression categories from both the databases. The JAFFE database contains facial images of ten Japanese females, where each has two to four samples for each expression. In total, there are 213 greyscale FaE images in this database, each of pixel resolution 256×256. The MUG database contains image sequences of FaEs belonging to 86 subjects comprising of 35 women and 51 men. Each image is of resolution 896×896. In our experiments we use images of 52 subjects, where two to four images were extracted from image sequences per subject per expression, totalling 1022 images. For the purpose of computational efficiency, all the images were resized to 51×51 pixels. The number of images used in our experiments and their resolutions are listed in Table 4. In our experiments we convert all the images into $8-$bit grayscale images, where each pixel value is 8-bit long.

TABLE 4
Details of utilized images from JAFFE and MUG.

Database	Subjects	Images per Class	Resolution
JAFFE	10	28∼32	51×51
MUG	52	110∼120	51×51

In this section we attempt to show that the proposed method doesn't degrade the classification accuracy due to the randomization. Hence, in this paper, without loss of generality, we use leave-one-out [35] strategy. More precisely, one image is removed from the database and all the remaining images are used for training, while the removed image is used as a test image. This procedure is repeated for a different left out test image each time until all the images are tested. Then we represented the results as a confusion matrix (as shown in Table 5 and Table 7). The accuracy can be inferred in different ways from the confusion matrix i.e., precision, recall, specificity, false positive rate, Matthews correlation coefficient, F1-score and accuracy etc. However the work in [29] uses precision $\times$ 100 i.e., the ratio between correctly identified expression and total images tested under the same expression times 100. The same strategy applied in this paper too.

The transformation matrix $\mathbf{U}$ in (4) can be obtained using generalized eigenvectors of matrix pairs $\{\mathbf{S_B}, \mathbf{S_W}\}$ if $\mathbf{S_W}$ is non-singular. Since the number of images used for training in this experiment is smaller than the size of the image vector (i.e., $N < n$), the matrix $\mathbf{S_W}$ becomes singular. In order to overcome

this issue, we exploit the technique used in [23], where PCA transformation matrix $\mathbf{W}_P$ has been used to reduce the dimension of the input space such that $\mathbf{S_W}$ becomes non-singular in the reduced dimensional space. Using $\mathbf{W}_P$ and (4), the LFDA based transformation matrix $\mathbf{W}_L$ can be obtained as

$$\mathbf{W}_L = \underset{\mathbf{Z}}{\operatorname{argmax}} \frac{\mid \mathbf{Z}^T \mathbf{W}_P^T \mathbf{S_B} \mathbf{W}_P \mathbf{Z} \mid}{\mid \mathbf{Z}^T \mathbf{W}_P^T \mathbf{S_W} \mathbf{W}_P \mathbf{Z} \mid}. \quad (26)$$

Note that the LFDA transformation matrix $\mathbf{U}$ is composed of generalized eigenvectors of matrices of $\{\mathbf{W}_P^T \mathbf{S_B} \mathbf{W}_P, \mathbf{W}_P^T \mathbf{S_W} \mathbf{W}_P\}$. Now the modified transformation matrix $\mathbf{U} = \mathbf{W}_P \mathbf{W}_L$ will be used in (2) and (8) for dimensionality reduction.

In FEC literature, to the best of our knowledge, there is no theory defines how to choose the feature vectors (i.e., how many feature vectors or which feature vectors) to get optimal classification accuracy. The only way is to do this is based on trial and error. An extensive simulation was conducted in [29] by checking the accuracy against different number of combinations for size of $\mathbf{W}_P$ and $\mathbf{W}_L$. To avoid repetition and to maintain fair comparison between proposed work and [29], we use the same parameters obtained in [29] for the following experiments.

6.1 Experiments on the JAFFE Database

This subsection describes our series of experiments on the JAFFE database. Initially, we should obtain the dimensions of the $\mathbf{W}_P$ and the $\mathbf{W}_L$ corresponding to best classification accuracy. In order to compare the performance of the proposed method against the Paillier encryption based technique and the conventional technique (i.e., without privacy preservation), we exploit the parameter details used in [29]. Denote the dimension of $\mathbf{W}_P$ as p and that of $\mathbf{W}_L$ as l. Fig. 2 depicts the recognition rates for various values of p and l. In Fig. 2, top accuracy of 94.37% is achieved when $p = 90$ and $l = 40$. Table 5 shows the confusion matrix corresponding to the top recognition rate. Note that for some classes, e.g. the SA class, the recognition rate tends to be lower than other expression classes because the class is highly confused with other expressions [36]. We consider only the top recognition case (i.e., $p = 90$ and $l = 40$) to evaluate the proposed scheme. To illustrate the effect of the scalar s in Table 1, we obtained the classification accuracy of the proposed algorithm for five different scalar values and the results were plotted in Fig. 3. In the same figure, we plotted the accuracy of [29] against the scaling factor and the accuracy of conventional scheme explained in Section 4. Our algorithm performs better than [29] for the smaller scaling factors. Also our algorithm achieves the maximum accuracy faster than [29]. This is due to the fact that the work in [29] uses the scaling factor right from the beginning of the classification process (from calculating the difference

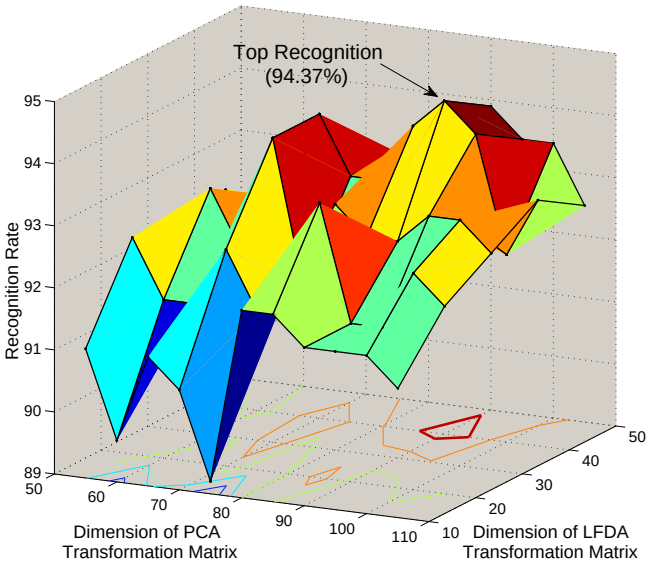

Fig. 2. Recognition rate for various dimensions of $\mathbf{W}_P$ and $\mathbf{W}_L$ for JAFFE database [29].

TABLE 5
Confusion matrix for JAFFE database.

	AN	DI	FE	HA	NE	SA	SU	Accuracy
AN	**29**	1	0	0	0	0	0	96.7%
DI	0	**27**	1	0	0	1	0	93.1%
FE	0	1	**30**	0	0	0	1	93.8%
HA	0	0	0	**29**	1	1	0	93.5%
NE	0	0	0	0	**30**	0	0	100%
SA	0	0	2	1	0	**28**	0	90.3%
SU	1	1	0	0	0	0	**28**	93.3%
Average Accuracy								**94.37%**

test vector). However, our algorithm uses the scaling factor to remove random noise from the Euclidean distances (at the very end of the classification process). This makes our algorithms to reduce the number of errors compared to [29] when the scaling factors are small. The crucial point is that the classification accuracies of both the algorithms eventually become equal to the accuracy of the conventional approach (i.e., 94.37 percent) when s is at a sufficient level, in this case $s > 10^4$. Hence, our algorithm doesn't degrade the classification accuracy due to the privacy preservation.

TABLE 6
Dimensions of PCA and LFDA used to get best performance in MUG database.

	Exp. 1	Exp. 2	Exp. 3
Total images	350	567	840
Images per class	50	81	120
Dimension of PCA	150	160	300
Dimension of LFDA	40	40	50

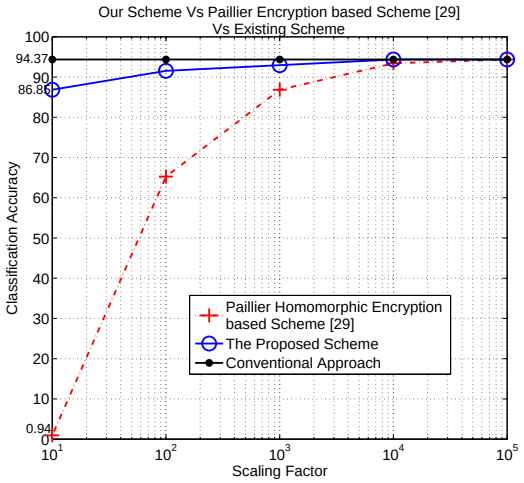

Fig. 3. An effect of scaling factor.

6.2 Experiments on the MUG Database

The MUG database contains more facial images than the JAFFE database and includes images of both men and women. Using the MUG database we studied different class sizes against accuracy, scaling factor, dimension of PCA and dimension of LFDA. In particular, we considered three experiments with different class sizes: Exp. 1: 50 images per class, Exp. 2: 81 images per class and Exp 3: 120 images per class. We then empirically obtained the dimensions of PCA and LFDA corresponding to best classification accuracy. Table 6 shows the details of these parameters. Then experiments similar to the JAFFE database were conducted for MUG database using the proposed method, Paillier encryption based method [29], and conventional scheme. Fig. 4 compares the top classification accuracies for all three methods and for all three class sizes. In Fig. 4, all three methods achieve same accuracy since scaling factor $s = 10^6$ has been used. This is due to the fact that the error caused by integer approximation is compensated by larger scaling factor.

Top recognition rate 95.24% is achieved when the number of images per class is equal to 81 while the performances of other two experiments were below 95%. Hence, the accuracy is not monotonically increasing with the class size. From Table 6 and Fig. 4, the dimension of the $\mathbf{W}_P$ used for dimensionality reduction increases with that of the class size while the dimension of the $\mathbf{W}_L$ remains nearly the same despite increase in class size. The crucial point is that the accuracy of the proposed method is same as the accuracies of existing methods regardless of class sizes. Hence the proposed method doesn't degrade the performance due to the size of the problem.

In order to evaluate the relation between scaling factor and classification accuracy, we consider the top recognition case. The confusion matrix for the case of 81 facial images per class without privacy preservation is given in Table 7 when the total number of images, dimensions of PCA, and dimensions of LFDA are 567, 160, and 40, respectively. Then we obtained the accuracies of each expression class by varying the scaling factor using the proposed method. The bar chart in Fig. 5 demonstrates that the number of correctly classified images per class increases monotonically with the scaling factor. The proposed method achieves the results in Table 7 when $s = 10^4$. Hence, the proposed method performs equally well as conventional scheme regardless of type of expression class.

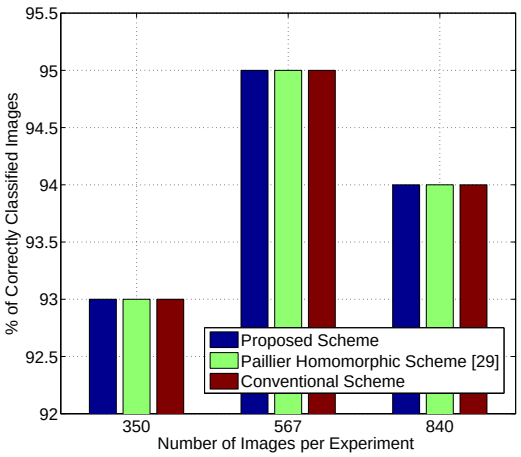

Fig. 4. Comparison of proposed method on MUG.

TABLE 7
Confusion matrix for MUG database in plain domain.

	AN	DI	FE	HA	NE	SA	SU	Accuracy
AN	**78**	0	1	0	2	0	0	96.30%
DI	0	**76**	1	2	0	1	1	93.83%
FE	2	1	**72**	1	2	0	3	88.89%
HA	1	1	2	**77**	0	0	0	95.06%
NE	1	1	0	0	**79**	0	0	97.53%
SA	0	0	0	0	0	**80**	1	98.77%
SU	1	0	1	0	0	1	**78**	96.30%
Average Accuracy								**95.24%**

6.3 Computational and Communication complexity

We now compare the complexity of the proposed algorithm against [29]. The recommended key size for Paillier encryption based scheme in [29] is 1024-bits, hence for the sake of comparison, we consider 1024-bits long random integers in the proposed scheme and show that our algorithm still outperforms [29].

6.3.1 *Computational complexity*

In order to compare these works, let us set, without loss of generality, the security parameter to 1024

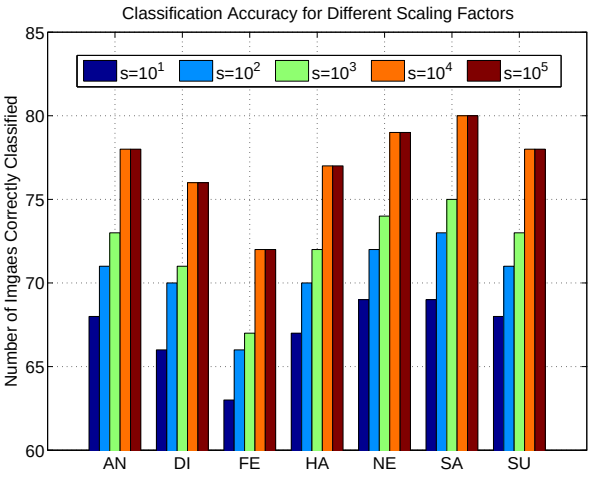

Fig. 5. Number of images correctly classified in randomized domain for each class for different scaling factors.

bits and denote the computational time (in ms) for multiplication, modulo exponentiation in 1024 bits field as C_m and C_e, respectively. Table 8 compares the complexity of both the schemes for both the server (S) and client (C) (the size of the images, total number of training images, and the total number of feature vectors are denoted as n, N, and m) in terms of the number of modulo exponentiations and multiplications. From [37], we can roughly estimate $C_e \approx 240C_m$. Let us define computational efficiencies of the proposed algorithm against [29] at the client side and the server side as e_C and e_S, respectively where

$$e_C = \frac{Complexity\ for\ client\ in\ [29]}{Complexity\ for\ client\ in\ proposed\ method},$$
$$e_S = \frac{Complexity\ for\ server\ in\ [29]}{Complexity\ for\ server\ in\ proposed\ algorithm}.$$

Table 9 shows the computational efficiency of the proposed algorithm for different set of parameters. In Table 9, we calculate the efficiency at both the client and server side by varying the parameters m, N, and n. The computational complexity to the client in the proposed algorithm is almost 120 times less than the complexity required for [29]. At the server side, our algorithm outperforms when the total number of training images (N) is less than 300. When N increases, the efficiency of the proposed algorithm at the server side drops slightly compared to [29] if we don't consider the parallel computation.

It is obvious from Table 8 and Table 9 that the computation time at the server is dominated by $mN(n+1)C_m$ i.e., Section 5.3 in Table 8. In fact this computation, as shown in Fig. 6, can be performed in parallel by splitting this into mN X $(n+1)C_m$ i.e., mN parallel computations. These computations are represented in (24) and (25). In fact, for these computations the client needs to send only $\boldsymbol{\gamma}$ and $(2\tilde{\mathbf{t}} + \boldsymbol{\gamma})$ to the server. Then the server can compute those the five scalar products (shown in (24) and (25) or in Fig. 6) for all feature vectors and training images in parallel since these computations are independent from each other (note that these scalar products are computed using PP algorithm in Table 1). This approach reduces the computational time at the server side approximately by mN times i.e., if $m = 10$ and $N = 300$

TABLE 8
Comparison of computational cost.

Section	for	Method in [29]	Proposed Method
5.1	C	$n(C_m + C_e)$	–
	S	$n(2C_m + C_e)$	–
5.2	C	–	–
	S	$m(n-1)C_m + mnC_e$	–
5.3	C	NC_m	$2nC_m$
	S	$[(2+5m)C_m + (1+4m)C_e]N$	$mN(n+1)C_m$
5.4	C	$(2C_m + C_e)log_2 N$	NC_m
	S	$(9C_m + 3C_e)log_2 N$	$(n+1)C_m$
Total	C	$(n+N+2log_2 N)C_m + (n+log_2 N)C_e$	$(2n+N)C_m$
	S	$[2n+(n-1)m+(2+5m)N+9log_2 N]C_m + (n+mn+N+4mN+3log_2 N)C_e$	$(mN+1)(n+1)C_m$

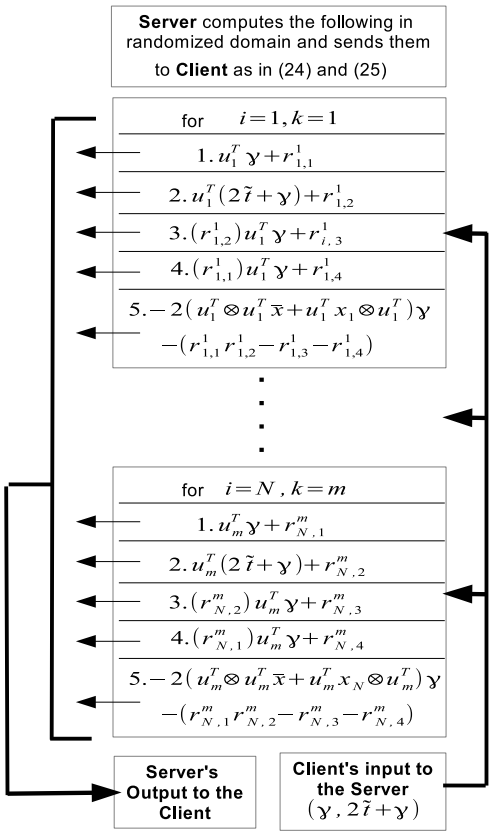

Fig. 6. Parallel Computation at server side.

then parallel computation reduces the computational time by approximately 3000 times. Even though the proposed algorithm increases the computational task to the server (when $N > 300$), if we assume that the server could be cloud and capable to perform parallel computations, then the computational time can be reduced substantially. At the same time it is not possible to perform parallel computation in [29] since the Euclidian distance comparison is dependent on previous comparison.

It is also worth noting that even though the MUG database is almost five times larger than the JAFFE database, their is no substantial difference in top recognition rates between the two databases (i.e., top recognition rates of JAFFE and MUG are 94.37% and 95.24%, respectively), so if it is desired that the server side overall overhead be further reduced then one could consider sticking with smaller databases without needing to substantially compromise the classification performance.

TABLE 9
Comparison of computational efficiency of the proposed algorithm against [29] .

		$n = 10000$		$n = 20000$		$n = 30000$	
m	N	e_C	e_S	e_C	e_S	e_C	e_S
10	200	119	1.42	119	1.37	120	1.35
20	200	119	1.36	119	1.31	120	1.29
10	400	118	0.76	119	0.71	119	0.69
20	400	118	0.73	119	0.69	119	0.66
10	800	116	0.43	118	0.38	119	0.36
20	800	116	0.41	118	0.37	118	0.35

6.3.2 Communication complexity

We measured the total communication complexity in terms of data being communicated between the server and client. In our algorithm, the client and server use only 8-bit size of data during the difference test vector calculation and projection steps. Later they use 1024-bits size of data and interact via the secure two-party algorithm proposed in Table 1. Let us compare the communication complexity for JAFFE database. In the proposed scheme, the client initially sends $2 \times 2501 \times 1024$-bits (i.e., $n = 2501$) to the server. Then the server sends back $2 \times 40 \times 212 \times 1024 + 1$ (i.e., $N = 212$, and $m = 40$) size of data to the client. Finally, in order to find the identity of the matched image, the client sends 1024×212-bits to the server which sends back another 1024-bits to the client. In total, the communication bandwidth required for our algorithm is $2.84MB$. However, the communications complexity of the Paillier cryptography based method in [29] requires $(n+m+1) \times 2048 = (2501+40+1) \times 2048 = 0.65MB$ bandwidth for the difference test vector computation and projection steps. For the match finding steps, [29] requires $6N \times 2048 + N(2l+1) \times 1024 = 6 \times 212 \times 2048 + 212 \times (106+1) \times 2048 = 3.23MB$. In total, [29] requires 3.72MB of bandwidth, when security parameter is 1024, which is nearly 31 percent higher than the proposed approach.

7 Conclusions and Future works

In this paper, we have proposed a lightweight FEC to outsource the data classification task to the untrusted third-parties. We exploited randomization technique to preserve the privacy of the client and server side parameters. Since the proposed method is developed using randomization, the computational and communication complexities were substantially reduced compared to the existing schemes. In order to validate the proposed method, we have experimented our method on popular FaE databases. The experiment results show that the classification accuracy of the proposed method is the same as the traditional approach (which has no privacy preservation) while preserving the privacy of the subjects involved in the classification. This proves the reliability of the proposed method.

Let us discuss some limitations and possible future works in this area. The proposed algorithm used LFDA and NN techniques to achieve accuracy above 95% for the JAFFE and MUG databases. However, as seen in [38], the same techniques are not sufficient to achieve a better classification accuracy when the expressions are spontaneous [39]. Hence, developing PP algorithms for more robust techniques such as local feature analysis, Gabor features, non-negative matrix factorization and local nonnegative matrix factorization, and local binary pattern are crucial.

The proposed scheme required to send the whole image to the server for classification where the portion corresponds to the expression is substantially lower than the whole image. Hence the computational complexity could be further reduced by not sending the redundant image potions to the server. Instead client could use techniques such as SIFT to extract some of non-sensitive features in plain domain. Hence developing a PP algorithm to share classification task between client-and server might further reduced the complexity.

References

[1] L. Zhang and D. Tjondronegoro, "Facial expression recognition using facial movement features," *IEEE Trans. Affective Computing*, vol. 2, pp. 219–229, Oct.-Dec. 2011.

[2] R. Calvo and S. D'Mello, "Affect detection: An interdisciplinary review of models, methods, and their applications," *IEEE Trans. Affective Computing*, vol. 1, pp. 18–37, Jan. 2010.

[3] R. Picard, *Affective Computing*. MIT Press, 1997.

[4] M. Pantic and L. Rothkrantz, "Automatic analysis of facial expressions: the state of the art," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, pp. 1424–1445, Dec. 2000.

[5] A. Saman and P. Iyengar, "Automatic recognition and analysis of human faces and facial expressions: A survey," *Pattern Recognition*, vol. 25, pp. 65–77, 1992.

[6] B. Fasel and J. Luettin, "Automatic facial expression analysis: A survey," *Pattern Recognition*, vol. 36, no. 1, pp. 259–275, 2003.

[7] R. Duda and P. Hart, *Pattern classification and scene analysis*. New York: Wiley, 1973.
[8] Z. Erkin, M. Franz, J. Guajardo, S. Katzenbeisser, I. Lagendijk, and T. Toft, "Privacy-preserving face recognition," in *Proc. 9th International Symposium on Privacy Enhancing Technologies*, PETS '09, pp. 235–253, 2009.
[9] H. Lipmaa, S. Laur, and T. Mielikainen, "Cryptographically private support vector machines," *In: Proc. 12th ACM SIGKDD Int'l Conf. Knowledge Discovery and Data Mining*, pp. 618–624, Aug. 2006.
[10] H. Yu, X. Jiang, and J. Vaidya, "Privacy-preserving SVM using nonlinear kernels on horizontally partitioned data," in *Proc. of the 2006 ACM symposium on Applied computing*, pp. 603–610, ACM, 2006.
[11] M. Barni, et. al. "Privacy-preserving fingercode authentication," in *12th ACM workshop on Multimedia and security*, (New York, NY, USA), pp. 231–240, ACM, 2010.
[12] W. Du and Z. Zhan, "Building decision tree classifier on private data," in *Proc. IEEE Int'l Conf. Privacy, Security and Data Mining-Volume 14*, pp. 1–8, Australian Computer Society, Inc., 2002.
[13] M. Barni, P. Failla, R. Lazzeretti, A.-R. Sadeghi, and T. Schneider, "Privacy-preserving ECG classification with branching programs and neural networks," *IEEE Trans. Information Forensics and Security*, vol. 6, no. 2, pp. 452–468, 2011.
[14] Y. Rahulamathavan, R. Phan, S. Veluru, K. Cumanan, and M. Rajarajan, "Privacy-preserving multi-class support vector machine for outsourcing the data classification in cloud," *IEEE Trans. Dependable Secure Computing*, vol. 11, no. 5, pp. 467–479, Sept. 2014.
[15] Y. Rahulamathavan, S. Veluru, R. Phan, J. Chambers, and M. Rajarajan, "Privacy-preserving clinical decision support system using gaussian kernel based classification," *IEEE Journal of Biomedical and Health Informatics*, vol. 18, no. 1, pp. 56–66, Jan. 2014.
[16] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *Advances in Cryptology EUROCRYPT 99*, vol. 1592 of *LNCS*, pp. 223–238, 1999.
[17] C. Meyer, *Matrix Analysis and Applied Linear Algebra*. In Society for Industrial and Applied Mathematics, ISBN – 0-89871-454-0, 2000.
[18] C. E. Shannon, "Communication theory of secrecy systems*," *Bell system technical journal*, vol. 28, no. 4, pp. 656–715, 1949.
[19] R. Lu, X. Lin, and X. Shen, "SPOC: A secure and privacy-preserving opportunistic computing framework for mobile-healthcare emergency," *IEEE Trans. Parallel and Distributed Systems*, vol. 24, no. 3, pp. 614–624, 2013.
[20] L. Sirovich and M. Kirby, "Low-dimensional procedure for the characterization of human faces," *J. Opt. Soc. Am. A*, vol. 2, pp. 519–524, 1987.
[21] M. Turk and A. Pentland, "Face recognition using eigenfaces," in *Proc. IEEE Computer Vision and Pattern Recognition*, pp. 586–591, Jun. 1991.
[22] M. Turk and A. Pentland, "Eigenfaces for recognition," *J. Cognitive Neuroscience*, vol. 3, no. 1, 1991.
[23] P. Belhumeur, J. Hespanha, and D. Kriegman, "Eigenfaces vs. fisherfaces: Recognition using class specific linear projection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, pp. 711–720, Jul. 1997.
[24] R. A. Fisher, "The use of multiple measures in taxonomic problems," *Ann. Eugenics*, vol. 7, pp. 179–188, 1936.
[25] M. Sugiyama, "Dimensionality reduction of multimodal labeled data by local fisher discriminant analysis," *The Journal of Machine Learning Research*, vol. 8, pp. 1027–1061, 2007.
[26] X. Yu and X. Wang, "Uncorrelated discriminant locality preserving projections," *IEEE Signal Process. Lett.*, vol. 15, pp. 361–364, 2008.
[27] S. Zhang, X. Zhao, and B. Lei, "Facial expression recognition using local fisher discriminant analysis," in *Advances in Computer Science, Environment, Ecoinformatics, and Education*, vol. 214 of *Communications in Computer and Information Science*, pp. 443–448, Springer Berlin Heidelberg, 2011.
[28] W. Yu, X. Teng, and C. Liu, "Face recognition using discriminant locality preserving projections," *Image and Vision Computing*, vol. 24, no. 3, pp. 239–248, 2006.
[29] Y. Rahulamathavan, R. Phan, J. Chambers, and D. Parish, "Facial expression recognition in the encrypted domain based on local fisher discriminant analysis," *IEEE Trans. Affective Computing*, vol. 4, no. 1, pp. 83–92, Jan.-Mar. 2012.
[30] L. Zelnik-manor and P. Perona, "Self-tuning spectral clustering," in *Advances in Neural Information Processing Systems 17*, pp. 1601–1608, MIT Press, 2004.
[31] K. Fukunaga, *Introduction to Statistical Pattern Recognition*. Academic Press, second edition ed., 1990.
[32] M. Lyons, J. Budynek, and S. Akamatsu, "Automatic classification of single facial images," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 21, pp. 1357–1362, Dec. 1999.
[33] N. Aifanti, C. Papachristou, and A. Delopoulos, "The MUG facial expression database," in *Proc. 11th Int. Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS)*, pp. 1–4, Apr. 2010.
[34] P. Ekman and W. Friesen, *Facial Action Coding System*. Palo Alto, CA:Consulting Psychologists Press, 1978.
[35] G. Cawley and N. Talbot, "Efficient leave-one-out cross-validation of kernel fisher discriminant classifiers," *Pattern Recognition*, vol. 36, no. 11, pp. 2585–2592, 2003.
[36] X. Zhao and S. Zhang, "Facial expression recognition based on local binary patterns and kernel discriminant isomap," *Sensors*, vol. 11, no. 10, pp. 9573–9588, 2011.
[37] K.-H. Huang, Y.-F. Chung, C.-H. Liu, F. Lai, and T.-S. Chen, "Efficient migration for mobile computing in distributed networks," *Computer Standards & Interfaces*, vol. 31, no. 1, pp. 40–47, 2009.
[38] S. Aina, Y. Rahulamathavan, R. C.-W. Phan, and J. A. Chambers, "Spontaneous expression classification in the encrypted domain", *9th IMA International Conference on Mathematics in Signal Processing*, Dec. 2012, Birmingham, UK.
[39] S. Wang, et.al., "Analyses of a multimodal spontaneous facial expression database", *IEEE Transactions on Affective Computing*, vol. 4, no. 1, pp. 34–46, Apr. 2013.
[40] O. Goldreich, "Secure multiparty computation", (working draft), available: http://www.wisdom.wei zmann.ac.il/oded/pp.html. (Sep. 1998)

Yogachandran Rahulamathavan received the PhD degree in Signal Processing from Loughborough University, UK in 2011. He is currently working as a Research Fellow with the Information Security Group at City University London, UK. His research interests include signal processing, machine learning and information security and privacy.

Muttukrishnan Rajarajan (SM'05) received the BEng and PhD degrees from City University London in 1994 and 1999 respectively. He joined City University London back in 2002 where he is currently a full Professor and heads the Information Security Research Group. He has research expertise in the areas of privacy preserving techniques, mobile security and Cloud security. More details can be found at www.staff.city.ac.uk/∼raj.