SecureDL: A privacy preserving deep learning model for image recognition over cloud

This paper was downloaded from TechRxiv (<https://www.techrxiv.org>).

LICENSE

CC BY 4.0

SUBMISSION DATE / POSTED DATE

27-01-2021 / 28-01-2021

CITATION

Tanwar, Vishesh Kumar; Raman, Balasubramanian; Rajput, Amitesh Singh; Bhargava, Rama (2021): SecureDL: A privacy preserving deep learning model for image recognition over cloud. TechRxiv. Preprint. <https://doi.org/10.36227/techrxiv.13650059.v1>

DOI

[10.36227/techrxiv.13650059.v1](https://doi.org/10.36227/techrxiv.13650059.v1)

SecureDL: A privacy preserving deep learning model for image recognition over cloud

Vishesh Kumar Tanwar^{a,c,*}, Balasubramanian Raman^{b,c}, Amitesh Singh Rajput^{b,c}, Rama Bhargava^{a,c}

^a*Department of Mathematics*

^b*Department of Computer Science & Engineering*

^c*Indian Institute of Technology Roorkee, India*

Abstract

The key benefits of cloud services, such as low cost, access flexibility, and mobility, have attracted users worldwide to utilize the deep learning algorithms for developing computer vision tasks. Untrusted third parties maintain these cloud servers, and users are always concerned about sharing their confidential data with them. In this paper, we addressed these concerns for by developing *SecureDL*, a privacy-preserving image recognition model for encrypted data over cloud. Additionally, we proposed a block-based image encryption scheme to protect images' visual information. The scheme constitutes an order-preserving permutation ordered binary number system and pseudo-random matrices. The encryption scheme is proved to be secure in a probabilistic viewpoint and through various cryptographic attacks. Experiments are performed for several image recognition datasets, and the achieved recognition accuracy for encrypted data is close with non-encrypted data. *SecureDL* overcomes the storage, and computational overheads occurred in fully-homomorphic and multi-party computations based secure recognition schemes.

Keywords: Cloud computing, Image classification, Permutation ordered binary number system, Encrypted domain

*Corresponding author

Email address: vtanwar@ma.iitr.ac.in (Vishesh Kumar Tanwar)

1. Introduction

Over the last decade, numerous state-of-the-art algorithms for various artificial intelligence tasks [1, 2, 3], utilizing convolutional neural networks (CNN) and long short term memory (LSTM), have been proposed. Besides their high performances, these systems are time-consuming and require high-configuration computing resources such as GPUs. To facilitate users, the cloud service providers, hereafter abbreviated as CSP, have developed Infrastructure-as-a-Service (IaaS), which constitutes deep learning (DL) frameworks and the respective resources. Azure Machine Learning¹ and Google Prediction API² are few of CSPs. In these services, users have to share their private data with CSP to learn on-demand DL services. In this protocol, both user and CSP will get benefited. However, user data may contain sensitive and confidential information such as patient’s medical reports or organization’s legal documents, whose leakage can drastically affect the functioning and reputation of the underlying individual/organization³. Hence, along with the superior benefits offered by cloud infrastructures, users are also concerned to utilize the cloud services while preserving their data confidentiality.

Various image encryption schemes [4, 5] have been proposed to deal with the problem of secure storage over the cloud servers as an adversary cannot reveal any information from the encrypted form. However, processing an encrypted image to achieve the same results as that of plain (non-encrypted) images poses a challenging problem because encryption schemes distort pixels inter-correlation. This paper focuses on proposing a system for learning an image recognition model using DL algorithms over the cloud without revealing image-data information.

In recent years, researchers have proposed algorithms for privacy-preserving image recognition over the cloud. These algorithms utilize multi-party computation (MPC) protocols or fully homomorphic encryption (FHE) techniques. In MPCs, a CSP provides a recognition model to multiple parties to individually train the model using their private data, followed by aggregation of encrypted weights over the CSP. This protocol may perform several

¹<https://azure.microsoft.com/en-us/services/machine-learning-studio/>

²<https://cloud.google.com/prediction/>

³<https://www.nbcnews.com/news/us-news/thousands-patient-records-leaked-hospital-data-breach-n756981>

times, and then the final model gets available to each party. In this way, each party will get a trained recognition model without compromising with their sensitive data. In [6, 7, 8], CSP provides a recognition model, pre-trained over a local data by CSP, to all parties to iterate one time over their private data. The resulting weights are encrypted by each party using a homomorphic encryption scheme and transmitted to CSP. Then, the CSP process, for instance, aggregates these encrypted weights utilizing the homomorphic property of encryption schemes and transmits the processed weights to each party. Each party then performs the second iteration and transmits parameters to CSP, and this loop continues until the terminating condition is achieved. These schemes preserve the data information efficiently; however, they exponentially increase the communication overheads between parties and CSP. Moreover, users have to maintain the computing resources, which increase the computation overheads at the user-end, henceforth contradicts CSPs' flexible services. The schemes proposed in [9, 10] utilize FHE technique [11] for encrypting user data which reduces the communication overhead between user and CSP, in comparison with MPCs. However, these models increase the data size [12] drastically; hence, the storage and computation overheads make it unsuitable for the recent era of technology such as cloud-based services and Internet-of-things environments.

In existing works, we observe two possible solutions for accomplishing image recognition tasks while preserving data privacy. In the first solution, the data can be encrypted using existing homomorphic encryption schemes, and new image recognition models can be proposed for these data, as presented in [9, 10]. In the second scenario, a new image encryption scheme can be proposed, which secures data information. The existing state-of-the-art recognition models, for instance VGG16 [13] and ResNet50 [14], can be used for learning model parameters.

In this work, we follow-up the later solution and propose an image encryption scheme that partially preserves the *trichotomous property* over the set of image pixels intensity values in encrypted form. The main contribution of *SecureDL* are as follows -

1. A novel symmetric encryption scheme based on the paradigm of substitution-permutation (SP) is proposed, constituting a permutation ordered binary (POB) number system followed by block-permutation matrices. The POB transforms each 8-bits gray-level intensity value of the input image to n -bits value. Then, the resulting transformed image is parti-

tioned into non-overlapping sub-blocks, and each sub-block is permuted through a different permutation matrix. Pseudo-random number generators generate the permutation matrices. Finally, the permuted sub-blocks of the image are re-positioned to their respective positions.

2. The proposed encryption scheme is proved to be secure from a probabilistic point-of-view. The security of the scheme is evaluated through histogram attack, pixel-correlation attack, and image information sensitivity.
3. The computational time required to encrypt an 299×299 -dimensional color image using blocks-size of 13×13 is 0.4069 second only. The storage space of the encrypted image is the same as that of the original image. These computational and storage overheads are very less as compared with the existing image encryption schemes. Moreover, *SecureDL* requires single round of interaction between the user and CSP (user $\rightarrow$ CSP $\rightarrow$ user), which make it suitable candidate for real-time cloud-based services.
4. The performance of *SecureDL* is evaluated over state-of-the-art image recognition models such as *VGG16*, *ResNet50*, *NasNet* and *InceptionV3* for public datasets namely, Kaggle’s Dogs Vs. Cats, ISIC dermoscopy (malignant and benign recognition), Linnaeus 5 dataset, and CIFAR-10. The recognition results achieved over encrypted data are close to that of plain (unencrypted) data.

Organization: Section 2 provides an overview of the related work of privacy preserving schemes. Section 3 is devoted for preliminaries, followed by the proposed methodology and proof-of-security in Sections 4 and 5 respectively. In Section 6, analytical security analysis of proposed scheme over various cryptographic attacks is discussed. The experimental recognition results and performance analysis of *SecureDL* are explained in Section 7 and Section 8 respectively. Finally, Section 9 concludes the paper and discusses the future work.

2. Related Work

Over the recent decades, researchers have proposed various privacy-preserving schemes for processing multimedia [15, 16, 17, 18]. We discuss a glimpse of

privacy-preserving models for image processing and other Machine Learning (ML) tasks followed by image recognition models, where our scheme lies. A detailed survey for preserving visual data was presented by Padilla-López et al. [19].

The existing privacy-preserving systems address the tasks of enhancement, and feature extraction [15, 20, 21] over the encrypted domain (ED). The scheme proposed by Lathey et al. [15, 20] is efficient for noise removal, edge sharpening, unsharp masking, and anti-aliasing from an image in ED. The image is secured through the (N, K) -threshold Shamir Secret Sharing (SSS) scheme. Tanwar et al. [21] proposed a privacy-preserving algorithm for image scaling over the cloud in which the image is encrypted by ramp SSS. They utilized 2D-Bicubic interpolation to interpolate the new pixel intensity values over cloud shares. Yan and Kankanhali [22] proposes a novel scheme for extracting and locating face regions in the ED. The information of the image is preserved using coefficients of discrete cosine transform. Rahulmathavan et al. [23] presented a facial expression recognition system in ED using local fisher discriminant analysis. The input image is encrypted using the Paillier cryptosystem. Further, privacy-preserving models for image denoising based on wavelet transformation over SSS scheme [24], Ring Learning with Errors cryptosystem [25], and Non-Local Means with Paillier cryptosystem are proposed. Researchers have proposed privacy-preserving models for performing clustering in ED for vertically [16, 26] and arbitrary [27, 28] partitioned data. Zhang et al. [29] proposed a possibilistic c-means algorithm for fuzzy clustering of FHE encrypted big data. Xing et al. [30] proposed a two-stage protocol for privacy-preserving k -means clustering based on Paillier cryptosystem over social networking big-data.

Towards privacy-preserving image recognition models, various schemes have been presented by Ma et al [6], Yonetani et al. [7], Xie et al. [9], Gilad et al. [10], Orlandi et al. [17], Bost et al. [18] and, Shokri and Shmatikov [31]. Xie et al. [9] presented a protocol for learning the neural networks over encrypted data. Authors approximated the non-linear activation functions to polynomials and discussed the theoretical aspects to learn and inference for encrypted data. However, the protocol was not tested for any standard dataset to analyze the proposed scheme’s robustness. Gilad et al. [10] proposed an FHE-scheme for encrypting gray-scale images and training a basic neural network architecture. They evaluated their scheme over the MNIST dataset only. The utilization of FHE techniques exponentially increases the computation overhead making these schemes impractical for real-time appli-

cations. In [17], an interactive protocol between the model and data owners for evaluating the activation functions in the neural network is presented using Homomorphic Encryption (HE). Bost et al. [18] proposed a scheme for Naive Bayes, Decision Trees, and Hyperplane Decision classifiers. The authors combined FHE without bootstrapping, Quadratic Residuosity, and Paillier cryptosystems for encrypting data.

This paper developed a privacy-preserving image recognition system to learn a DL framework for encrypted data over the cloud. A block-based image encryption scheme is proposed to protect each image’s visual information in the user’s data.

3. Preliminaries

In this section, we discuss the theoretical foundations used to understand the proposed method. The first layer in DL frameworks is the convolutional layer containing a class of different filters/kernels. In contrast with NN, whose input is a vector, input for the first layer of CNN-based architectures is a normalized raw image. These filters operate on the input points for extracting the spatial features.

3.1. Permutation ordered binary (POB) number system

Sreekumar and Sundar [32] introduced the generalized form of number system which depends upon two positive integers n and k where $n > k$. Specifically, for fixed n and k ($n > k$), each value lies in $\left\{0, 1, \dots, \binom{n}{k} - 1\right\}$ can be transformed into n -bit binary string with 1 at exactly k positions and 0 at the remaining $n - k$ positions. The mathematical formulation of POB-number for a positive integer i with parameters n and k ($n > k$) is given as $\mathcal{P}_{num}^{n,k}(i) = r_{n-1}r_{n-2}\dots r_0$, where r_i is either 0 or 1. Further, the POB-number $\mathcal{P}_{num}^{n,k}(i)$ is converted into a positive integer known as POB-value, denoted by $\mathcal{P}_{val}^{n,k}(i)$ and defined as

$$\mathcal{P}_{val}^{n,k}(i) = \sum_{i=0}^{n-1} r_i \binom{i}{t_i}, \text{ where } t_i = \sum_{j=0}^i r_j, \quad (1)$$

It is noted that the POB-number and POB-value for positive integer i (Eq.1) will be different for different parameters as shown in Table 1. For instance, the POB-value for $i = 5$ with parameters ($n = 9, k = 4$) and

($n = 35, k = 15$) are 39 and 64511 respectively. Also, POB number system follows the partial trichotomous property of positive integers (Defs. 3.1 and 3.1). For more details of POB system, the reader is advised to refer [32].

Table 1: Comparison of POB-values with different (n, k)

i	$\mathcal{P}_{val}^{9,4}(i)$	$\mathcal{P}_{val}^{9,7}(i)$	$\mathcal{P}_{val}^{35,15}(i)$
5	39	251	64511
88	308	508	130043

Definition 3.1. A set X of positive integers is said to be trichotomous if it satisfies the following property -

$$x < y \text{ or } x > y \text{ or } x == y \quad \forall x, y \in X \quad (2)$$

Definition 3.2. A function $\mathcal{F}$ defined over set X of positive integers is said to be partial trichotomous if -

$$x < y \implies \mathcal{F}(x) \leq \mathcal{F}(y) \quad \forall x, y \in X \quad (3)$$

Deductions from POB number system

1. For fix parameters (n, k) , $n > k$, there exists a unique POB-number and POB-value for each element of the set $\left\{0, 1, \dots, \binom{n}{k} - 1\right\}$.
2. POB is a partial trichotomous over the set of positive integers of finite cardinality.
3. For each set of parameters (n, k) , there are exactly $\binom{n}{k}$ POB-values.

4. Proposed methodology

This section discusses the details of the proposed encryption scheme for encrypting user's image datasets and our protocol defined for developing a secure image recognition model.

Figure 1: A pictorial representation of *SecureDL*. Two entities user ($\mathcal{U}$) and cloud ($\mathcal{C}$) are involved. $\mathcal{U}$ encrypts private data $\mathcal{D}$ to $\mathcal{D}^{enc}$ using Algorithm 2 and transmits to $\mathcal{C}$. $\mathcal{C}$ owns high-configuration resources and DL model $\mathcal{M}$ which he trains over $\mathcal{D}^{enc}$. Finally, transmits back the models parameters values $\mathcal{M}(\mathcal{W}, \mathcal{H})$ to $\mathcal{U}$.

4.1. Threat model

In our system, two entities are involved: a user or data owner $\mathcal{U}$, who wishes to develop an image recognition model for personal data without revealing visual information contained in data to the third party entity, and a CSP $\mathcal{C}$, who owns an image recognition DL framework $\mathcal{M}$ and provides it as a cloud service on a pay-as-you-go business model. We assume that CSP is "honest-but-curious or semi-honest," which means that CSP performs all the desired computations accurately but will try to learn the possible data information. Users' task is to encrypt the data at the local server and transmit the encrypted data to $\mathcal{C}$. The CSP will perform all the computations (i.e., train the model $\mathcal{M}$) over the users' transmitted encrypted data. Then, $\mathcal{C}$ will transmit the trained encrypted model to $\mathcal{U}$ for further inference purposes. Since $\mathcal{M}$ is trained over encrypted data; therefore, the learned parameters of $\mathcal{M}$ will also be in encrypted form. We assume that an adversary is an entity, present at cloud server, who can access users' encrypted data, weights, and hyper-parameters of the trained model $\mathcal{M}$.

4.2. System overview

In the framework, $\mathcal{U}$ generates the secret encryption key-set $\mathcal{K}$ which are used to encrypt the data $\mathcal{D}$ before transmitting to $\mathcal{C}$. The CSP trains and simulates the model $\mathcal{M}$ over user's encrypted data $\mathcal{D}^{enc}$. Finally, the encrypted weights of trained model $\mathcal{M}$ is sent back to the user for inference purposes. The pictorial representation of our proposed framework is depicted in Fig. 1. We assume that $\mathcal{D}$ contains gray-level images of dimension $M \times N$. The step-by-step description for developing image recognition model in the ED is defined below:

Step 1 - Generation of Security Key-set ($\mathcal{K}$): The encryption scheme defined in this work for encrypting image information requires a set of hyper-parameters values $\{n, k, K_1, K_2\}$, where (n, k) , $(n \geq k)$ are parameters of POB number system (defined in Section 3.1) and $K_1 \times K_2$ is the dimension of each block-permutation matrix. The values of n and k are generated using a pseudo-random number generator $\mathcal{G}$ whereas K_1 and K_2 are divisors of M and N respectively. Additionally, T permutation matrices say $\pi = \{\pi_1, \pi_2, \dots, \pi_T\}$ each of dimension $K_1 \times K_2$ are generated using $\mathcal{G}$, where

$$T = \frac{M \times N}{K_1 \times K_2} \quad (4)$$

Specifically, the value T indicates the number of blocks in which $\mathcal{U}$ wishes to partition plain image I . Therefore, the secret-keys are $\mathcal{K} = \{n, k, \rho_1, \rho_2, \pi_1, \dots, \pi_T\}$ which are generated by $\mathcal{U}$ only and does not share with $\mathcal{C}$.

Step 2 - Formation of Encrypted Data: Here we explain the scheme for transforming plain data $\mathcal{D}$ into encrypted data $\mathcal{D}^{enc}$. The successful generation of secret-keys $\mathcal{K} = \{n, k, \rho_1, \rho_2, \pi_1, \dots, \pi_T\}$ where $(n \geq k)$ and $\binom{n}{k} \gg l$, l is maximum intensity value of all $M \times N$ -dimensional gray-scale images in $\mathcal{D}$, leads to data encryption process at user-end. The proposed encryption scheme is the two-stage process which provides high security to user's data.

In the primary stage, the POB-value corresponding to each intensity value $I(u, v)$ at pixel (u, v) of image I in $\mathcal{D}$ is computed and substituted

at pixel (u, v) of POB-space image, denoted by I_{POB} . Mathematically,

$$I_{POB}(u, v) = \mathcal{P}_{val}^{n,k}(I(u, v)) \quad \text{for each pixel } (u, v) \quad (5)$$

In the second stage, POB-spaced image I_{POB} , is partitioned into T sub-images say $I_{POB_1}, I_{POB_2}, \dots, I_{POB_T}$ each of dimension $\rho_1 \times \rho_2$ in row/column roster method. Then, pixel values of t^{th} sub-image I_{POB_t} are shuffled using t^{th} permutation matrix π_t . The obtained sub-image is denoted by $I_{POB_t}^{\pi_t}$ in which the intensity value at pixel (u, v) is the intensity value of I_{POB_t} at pixel-location given by $\pi_t(u, v)$. The mathematical equation is

$$I_{POB_t}^{\pi_t}(u, v) \mapsto I_{POB_t}(\pi_t(u, v)) = (I_{POB_t} \circ \pi_t)(u, v) \quad (6)$$

where $1 \leq u \leq \rho_1$, $1 \leq v \leq \rho_2$ and $\circ$ indicates the composition of functions. Then, all shuffled sub-images $I_{POB_1}^{\pi_1}, I_{POB_2}^{\pi_2}, \dots, I_{POB_T}^{\pi_T}$ are placed at the position of sub-images $I_{POB_1}, I_{POB_2}, \dots, I_{POB_T}$ respectively to obtain the block-wise shuffled image I_{POB}^{π} . The pseudo-code of the encryption scheme is presented in Algorithm 1.

Step 3 - Normalization of block-wise shuffled image I_{POB}^{π} : As explained in Section 3.1, POB number system transforms the q -bit intensity value to n -bit POB value. In contrast with decimal number system, the difference between two consecutive POB values is non-uniform. For instance, let P_{x-1}, P_x, P_{x+1} are three consecutive POB values for fixed parameters (n, r) with $D_{x+1} = P_{x+1} - P_x$ and $D_x = P_x - P_{x-1}$, then D_{x+1} may or may not be equal D_x . Further, the POB values are large for bigger values of parameters (n, r) . It exponentially increases the storage size of each image making the user's dataset $\mathcal{D}$ expensive in terms of transmission and storage. Also, it requires high computational resources for learning model parameters. Therefore, we normalize each image I_{POB}^{π} over the range $[0, l]$ to obtain the encrypted image I_{Enc} as follows

$$I_{Enc}(u, v) = \left(\frac{I_{POB}^{\pi}(u, v) - \min_{I_{POB}^{\pi}}}{\max_{I_{POB}^{\pi}} - \min_{I_{POB}^{\pi}}} \right) \times l \quad (7)$$

where $\min_{I_{POB}^{\pi}}$ and $\max_{I_{POB}^{\pi}}$ indicate the minimum and maximum value of image I_{POB}^{π} for all $1 \leq u \leq M$ and $1 \leq v \leq N$. In this manner,

$\mathcal{U}$ can secure all the sensitive information contained in image dataset $\mathcal{D}$. For an RGB-color image, Step 2 and 3 are performed over each color-channel simultaneously with same key-set. Now, $\mathcal{U}$ can transmit the encrypted data $\mathcal{D}^{enc}$ to $\mathcal{C}$ for learning the parameters of recognition model over $\mathcal{D}^{enc}$. The toy-example of the encryption scheme over an RGB-image of dimension 4×4 is presented in Fig. 2.

We observed that the input image’s security increases with a large value of T and vice-versa. The cloud server can know the total number of classes in $\mathcal{D}^{enc}$ and the image associated with class number irrespective of image information. The deep learning-based recognition models belong to the class of supervised learning algorithms so, it is necessary to provide the class number associated with the input image. $\mathcal{U}$ does not share $\mathcal{K}$ with $\mathcal{C}$ instead can be destroyed after completion of encryption process.

Step 4 - Learning of Recognition Model ($\mathcal{M}$): In this step, $\mathcal{C}$ performs an end-to-end training of model $\mathcal{M}$ over the encrypted data $\mathcal{D}^{enc}$. The values of $\mathcal{M}$ ’s learnable weights $\mathcal{W}$ and hyper-parameters $\mathcal{H}$ will be updated, automatically or manually, till the termination condition is achieved. Note that $\mathcal{D}^{enc}$ is in the encrypted form, so the values $\mathcal{H}$, $\mathcal{W}$ and predictions are also in the encrypted form and thus, $\mathcal{M}$ can be assumed to be secure. Therefore, $\mathcal{C}$ is unable to extract correct information related to model’s parameter and predictions. We noticed that the (encrypted) trained model $\mathcal{M}^{Enc}$ perform worst when validated over plain (un-encrypted) test data. It is worth to say that if an adversary utilizes the trained model $\mathcal{M}^{Enc}$ over the plain-data to extract the class-related information, then the obtained inferences will be incorrect.

Step 5 - Transmission of Encrypted Results: After successfully learning the parameters, denoted as $\mathcal{M}(\mathcal{W}, \mathcal{H})$, $\mathcal{C}$ transmits these parameters to $\mathcal{U}$ for inference purposes. As the model $\mathcal{M}^{Enc}$ is trained over encrypted data $\mathcal{D}^{enc}$, therefore $\mathcal{U}$ have to encrypt the inference data with the same key-set $\mathcal{K}$ before computation.

In contrast with the existing encryption scheme, the proposed scheme partially preserves the trichotomous property of pixel intensity value along with encrypting image features using block-based shuffling matrices. Moreover, the existing encryption schemes exponentially increase the image size,

Figure 2: Working example of our proposed encryption scheme. (a) RGB-color image and its corresponding color channels, (b) Transforming each pixel value to POB-values with parameters $(n = 17, k = 13)$, (c) permutations matrices $\{\pi_1, \pi_2, \pi_3, \pi_4\}$, (d) it represents the relation between permutation matrices with RGB-channels, (e) RGB-channels obtained after permuting each image-channel through permutation matrices, given in (d), and (f) encrypted image obtained after normalizing over the range $[0, l]$. Note that here $l = 255$, maximum value in an RGB-color image.

requiring high storage and computational resources, making them incompatible for real-time scenarios. Besides the significant benefits, the scheme leaks some of the image information. However, we believe that a user may compromise with few shortcomings at the cost of a robust and efficient image recognition model with data privacy.

Algorithm 1 Formation of Encrypted Data ($\mathcal{D}^{enc}$)

```
1: Input: Image  $I$  and  $n, k$  ( $n \geq k$ )  $\triangleleft$  parameters of POB
2: Input:  $\pi = \{\pi_1, \pi_2, \dots, \pi_T\}$   $\triangleleft$  permutation matrices
3: Assume:  $I(u, v)$   $\triangleleft$  intensity value of  $I$  at  $(u, v)$ 
4: Output:  $I_{Enc}$   $\triangleleft$  encrypted image
5: procedure FORMATION OF POB-IMAGE ( $I_{POB} \leftarrow$  Eq.5)
6:   for  $u \leftarrow 1$  to  $M$  do
7:     for  $v \leftarrow 1$  to  $N$  do
8:        $I_{POB}(u, v) = \mathcal{P}_{val}^{n,k}(I(u, v)) + \text{random}([0, L])$ 
9:     end for
10:  end for
11: end procedure
12: Assume:  $I_{POB}$  is divided into  $T$  sub-images each of dimension  $\rho_1 \times \rho_2$ 
13: Randomly choose  $\pi_t$  from  $\pi$  for  $t^{th}$  sub-image  $I_{POB}^{\pi_t} \forall t = 1, 2, \dots, T$ 
14: procedure ENCRYPTING  $t^{th}$  SUB-IMAGE ( $I_{POB_t}^{\pi_t} \leftarrow$  Eq.6)
15:   for  $u \leftarrow 1$  to  $\rho_1$  do
16:     for  $v \leftarrow 1$  to  $\rho_2$  do
17:        $I_{POB_t}^{\pi_t}(u, v) = I_{POB_t}(\pi_t(u, v))$ 
18:       or  $= (I_{POB_t} \circ \pi_t)(u, v),$  (Eq. 6)
19:     end for
20:   end for
21: end procedure
22: Assume:  $I_{POB}^{\pi}$  be image obtained by re-positioning
23:   the permuted POB sub-images  $\{I_{POB}^{\pi_t}\}_{t=1}^T$ 
24: Assume:  $\max_{I_{POB}^{\pi}}$  be maximum of  $I_{POB}^{\pi}$ 
25: Assume:  $\min_{I_{POB}^{\pi}}$  be minimum of  $I_{POB}^{\pi}$ 
26: procedure NORMALIZING IN THE RANGE  $[0, l]$  (Eq. 7)
27:    $I_{Enc}(u, v) = \left( \frac{I_{POB}^{\pi}(u, v) - \min_{I_{POB}^{\pi}}}{\max_{I_{POB}^{\pi}} - \min_{I_{POB}^{\pi}}} \right) \times l \quad \forall (u, v)$ 
28:   return  $I_{Enc}$ 
29: end procedure
```

5. Proof-of-security of *SecureDL*

The use of multiple CSPs by the user to process the private data increases data breaching threat as curious CSPs can perform colluding attacks by collaborating their data. Singh et al. [33] proposed encryption using the

POB number system over Shamir's SSS, proving that the POB is robust for securing image data. In contrast, our encryption scheme does not share secret data over different CSPs. Hence, it secures user data efficiently and preserves it from colluding attacks. Furthermore, accessing a single CSP will make monetary benefits to the user.

Here, we will mathematically analyze the security strength of the combination of (n, k) -POB number system and permutation matrices $\pi = \{\pi_1, \pi_2, \dots, \pi_T\}$. As we choose T permutation matrices randomly, each of the dimensions say $\rho_1 \times \rho_2$, so we will prove the probability of selecting these T matrices is minuscule. If an adversary tries to find these T matrices through brute-force, then the probability of getting these matrices is negligible, and so the image information. Further, we will prove that an image encrypted through our proposed scheme using key $\mathcal{K} = \{n, k, \pi_1, \pi_2, \dots, \pi_T\}$ is probabilistic-secure. It means that the probability of extracting image content from its encrypted form is close to zero.

Theorem 5.1. *The total number of permutation matrices each of dimension $\rho_1 \times \rho_2$ over the range $[1, \rho_1 \times \rho_2]$ is $(\rho_1 \times \rho_2)!$.*

Proof. Let π_t be an arbitrary permutation matrix of dimension $\rho_1 \times \rho_2$ over the range $[1, \rho_1 \times \rho_2]$. Matrix π_t can be transformed into a row-vector through roaster-scanning manner say V_{π_t} , of length $\rho_1 \times \rho_2$. Then, V_{π_t} can be seen as a permutation obtained by rearranging the elements of set $S = \{1, 2, \dots, \rho_1 \times \rho_2\}$.

It is noted that interchanging the positions of a single-pair of values in π_t , results new permutation matrix π_j , thus changes the vector V_{π_t} to V_{π_j} , $t \neq j$. Therefore, permutation vector (V_{π_t}) is unique for each permutation matrix π_t . The collection of all permutation matrices is equal to the set of all possible permutations over the set S .

In the context of abstract group theory [34], the collection of all permutations over a finite set S forms a finite symmetric group G_S [34]. Therefore, the total number of permutation matrices is equal to the order of the group G_S [34], defined as

$$\text{Order}(G_S) = (\rho_1 \times \rho_2)! \quad (8)$$

where $x!$ indicates the factorial of x .

Hence, the total number of permutation matrices of dimension $\rho_1 \times \rho_2$ are $(\rho_1 \times \rho_2)!$. For instance, ref. Corollary 5.1.1. $\square$

Corollary 5.1.1. *In particular for $\rho_1 = 256$ and $\rho_2 = 256$, the number of permutation matrices are $65536!$.*

Lemma 5.2. *Consider an image of dimension $M \times N$ which can be partitioned into T sub-images (sub-blocks) each of dimension $m \times n$, $m < M$ and $n < N$. Then the probability of getting T permutation matrices of dimensions $m \times n$ to permute the complete image is $\frac{T}{(m \times n)!}$.*

Proof. The lemma can be readily proved using Theorem 5.1. $\square$

Theorem 5.3. *A secret image I of dimension $M \times N$, when encrypted using key $\mathcal{K} = \{n, k, \pi_1, \pi_2, \dots, \pi_T\}$ is secure, where n and k are sufficiently large.*

Proof. Let $A = \{p_1, p_2, \dots, p_L\}$ be set of all distinct pixel intensity values in I and $B = \{q_1, q_2, \dots, q_Q\}$ be collection of all POB values with parameters n and k , where $Q = \binom{n}{k} - 1$. The total number of elements in A are L . As p_i is pixel intensity value, so $0 \leq p_i \leq 255 \quad \forall \quad i = 1, 2, \dots, L$.

Suppose a be an arbitrary pixel intensity value of I . Then, it is equiprobable that a can be any value from set A of cardinality L . The probability is given as -

$$\text{prob}(a = g) = \frac{1}{L}, \quad \text{where } g \in A \quad (9)$$

Suppose q_a be the POB value in B corresponding to pixel value a . Then, it is equiprobable that q_a can be any value from the set B of cardinality Q . This probability is given as

$$\text{prob}(q_a = g') = \frac{1}{Q}, \quad \text{where } g' \in B \quad (10)$$

Since, the parameters n and k are sufficiently large, so they can be chosen such that L is very less than $Q = \binom{n}{k} - 1$. It increases the range of POB values B for choosing q_a corresponding to pixel intensity value a in set A . (For instance, Ref. Corollary 2).

Further, I is partitioned into T sub-images (matrices) say $\{I_1, I_2, \dots, I_T\}$ each of dimension $\rho_1 \times \rho_2$. Also, I_t^{th} sub-image is permuted through π_t . The probability of choosing a permutation matrix π_t for I_t is given as -

$$\text{prob}(\pi_t | I_t) = \frac{1}{T} \quad (11)$$

It has been proved in Theorem 1 that total number of permutation matrices are $(\rho_1 \times \rho_2)!$ and therefore, the probability of generating T permutation matrices of dimensions $\rho_1 \times \rho_2$, is given as

$$\text{prob}(\pi = \{\pi_1, \pi_2, \dots, \pi_T\}) = \frac{T}{(\rho_1 \times \rho_2)!} \quad (12)$$

Hence, the probability of extracting a pixel intensity value q_a from an encrypted image I_{Enc} corresponding to pixel value a in I is given as

$$\text{prob}(q_a = g') = \frac{T}{Q \times (\rho_1 \times \rho_2)!}, \quad \text{where } g' \in B \quad (13)$$

It can be observed through Eq. 13 that the probability of extracting pixel values is very small. $\square$

Corollary 5.3.1. *Consider an image of size 299×299 which is partitioned into $T = 529$ sub-images each of dimension $\rho_1 \times \rho_2 = 13 \times 13$. Consider POB-parameters as $n = 17$ and $k = 13$, then the total number of POB values are 2380 i.e., $Q = 2380$. Also, the maximum intensity value in RGB-color image is 255, thus $L = 255$. Therefore, using Eq. 13, we have $\text{prob}(q_a = g') = \frac{23 \times 23}{2380 \times (13 \times 13)!} = 5.2065 \times 10^{-306}$.*

6. Security analysis

In this section, we present some commonly performed cryptographic attacks by an adversary. These attacks depend upon the quantitative values of the encrypted form of the input image.

6.1. Histogram attack

Histogram of a digital image refers to the graphical representation of its pixel intensity values, which significantly explains the contrast, brightness, and saturation effects of the image content. Histogram analysis is performed to illustrate the diffusion and confusion properties of the encrypted image in comparison with the plain image. In this attack, an adversary is known with the histograms corresponding to encrypted image I_{Enc} , but unknown with the histogram of plain image I . The adversary manages to extract the histogram information of I . Therefore, the histograms corresponding to I and I_{Enc} must be unrelated as much as possible.

Figure 3: Histogram Analysis. (a) secret image I , (b)-(d) histograms corresponding to red-green-blue channels of (a), (e) encrypted image I_{Enc} ($n = 17, k = 13, \rho_1 \times \rho_2 = 47 \times 47$), and (f)-(h) histograms corresponding to red-green-blue channels of (e).

As an illustration, the histograms of secret image I of dimension 329×329 and its encrypted form are depicted in Figs. 3(a) and 3(e) respectively. The histograms of red-green-blue channels of I (Fig. 3(a)) are depicted in Fig. 3 (b)-(d). The encrypted image I_{Enc} is obtained with POB-parameters $n = 17, k = 13$ and permutation matrices of dimensions $\rho_1 = \rho_2 = 47$. The histograms of respective channels are presented in Fig. 3 (f)-(h). Note that the frequency of pixel intensity values for I lies in the range $[0, 1200]$, whereas in I_{Enc} they lie in $[0, 800]$. It is clearly observed that histograms of corresponding channels of I and I_{Enc} are drastically different. Therefore, if an adversary exercised to extract any information from the histogram of encrypted image, it will receive incorrect information.

6.2. Pixels-correlation attack

The correlations between the neighboring pixels in an image depict a statistical relevance [35]. The high correlation indicates the high-visual quality of an image, which helps to extract meaningful information. The resistance of the image information from an adversary is considered high when the correlation value is low for its encrypted form I_{Enc} , which we will analyze here for *SecureDL*.

The correlation value ($corr$) for a gray-scale image I is evaluated by

considering a set of K neighboring pixels say $\{(x_1, y_1), (x_2, y_2), \dots, (x_K, y_K)\}$, as follows

$$cov(x, y) = \frac{1}{K} \sum_{i=1}^K (x_i - \mu(x))(y_i - \mu(y)) \quad (14)$$

$$corr(x, y) = \frac{cov(x, y)}{\sqrt{\sigma^2(x)} \times \sqrt{\sigma^2(y)}} \quad (15)$$

where (x, y) designates the two neighboring pixels of a secret image and μ, σ^2 and $cov(x, y)$ represent the mean, variance, and covariance, respectively. For a true-color image, $corr$ is an average sum of correlation for each channel. $corr$ is a real number over the range -1 and 1 . The values “1” and “ -1 ” indicate perfect positive and negative correlation respectively, whereas “0” represents no relationship between adjacent pixels depicting highly uncorrelated.

Experimentally, $corr$ is evaluated in each of diagonal, horizontal and vertical axes through randomly selected $K = 2000$ adjacent pixel pairs, over a standard Lena image I , of dimension 256×256 . The keys considered to encrypt I is same as that in Section 3. The corresponding value of $corr$ in each direction is presented in Table 2. It can be observed that the $corr$ obtained for I_{Enc} are close to 0 than the existing schemes [36, 37, 38]. It indicates that *SecureDL* significantly breaks the pixels inter-correlations. Further, we computed the $corr$ for different number of adjacent values such as $K = 1000, 5000, 10000, 15000, 25000$ and 35000 in Table no. 3.

Table 2: $corr$ comparison in diagonal, horizontal and vertical directions.

Dir.	Plain Lena Image	Chong et al. [36]	Teng et al. [37]	Rajput et al. [38]	<i>SecureDL</i> K = 2000
Diag.	0.9257	0.0060	0.0243	0.0177	0.0115
Hori.	0.9404	0.0088	0.0242	0.0128	0.0183
Vert.	0.9299	-0.0087	-0.0194	0.0097	0.0069

6.3. Comparison of information randomness sensitivity

In this section, we discuss the differences between the number of pixel intensity values of the secret image I and its encrypted image. The comparison is performed with different variations in parameters of the encryption scheme. It is noticed that the graphical representation between a gray-level (8-bit)

Table 3: *corr* comparison for different K of encrypted *Lena* image using *SecureDL*.

	K					
	1000	5000	10000	15000	25000	35000
Diag.	-0.0094	0.0218	-0.0071	-0.0018	0.0079	0.0030
Hori.	0.0103	-0.0286	-0.0105	-0.0016	0.0093	-0.0175
Vert.	0.0143	0.0013	0.0092	0.0137	0.0016	-0.0251

image with itself gives a linear line indicating the zero difference. However, this representation must be very random in the encrypted form, as depicted in Fig. 4. A secret image is presented in Fig. 4 (a) followed by its encrypted images (b) - (d) with POB parameters $n = 17$, $k = 13$ and permutation matrices of dimensions (left-to-right) $\rho_1 \times \rho_2 = 13 \times 13$, $\rho_1 \times \rho_2 = 37 \times 37$ and $\rho_1 \times \rho_2 = 47 \times 47$ respectively. The differences in the red, green, and blue channels of secret image (a) and encrypted image (b) are shown in Fig. 4 (e). Further, the randomness in each channel of secret image and the encrypted image (c) and (d) are analyzed and presented in Fig. 4 (f) and Fig. 4 (g) respectively. It is easy to infer that our scheme adds high-intensity confusion in the spatial domain of each color-channel and hence in the whole image. It deduce that the predictions of pixel intensity values of secret image are unable to extract from its encrypted image.

6.4. Information sensitivity in encrypted images

A comparative analysis between the variation occurred in the spatial domains of different encrypted images is performed. The pictorial representation of spatial variation is depicted in Fig. 5. Consider a secret image Fig. 4 (a) and its respective encrypted images Fig. 4 (b) - (d). In Fig. 5 (b), the differences in the corresponding red, green and blue channels of encrypted images Fig. 4 (b) and (c) can be visualized. Moreover, the differences in encrypted images Fig. 4 (b) and Fig. 4 (d), and Fig. 4 (c) and Fig. 4 (d) are pictures in Fig. 5 (b) and 5 (d) respectively. It is inference that if an image is encrypted with different encryption parameters, then the obtained encrypted images are drastically different. Hence, it can be observed the encrypted images are key-sensitive.

Figure 4: Comparison of randomness occurred in the encrypted images. (a) Secret image. (b)-(d) encrypted image obtained with $n = 17, k = 13$ and block-permutation matrices of dimensions 13, 37 and 47 respectively. (e) indicates the differences in red, green and blue channels of (a) and (b). (f) indicate the differences in red, green and blue channels of (a) and (c). (g) indicate the differences in red, green and blue channels of (a) and (d).

Figure 5: Comparison of the encrypted images obtained with different security keys. (left-to-right) (a) Difference in red, green and blue channels of Fig. 4 (b) Fig. 4 (c, (b))Fig. 4 (b) Fig. 4 (d), and (c) Fig. 4 (c) and Fig. 4 (d).

7. Experiments

The robustness of *SecureDL* in various scenarios have been validated through benchmark image classification datasets. The qualitative and quantitative analyses of the proposed scheme are also reported.

7.1. Datasets

1. **Kaggle Cat-Dog dataset**⁴: It contains true-color images of dogs and cats with varying breeds. The dataset is provided in kaggle’s recognition public competition presented by microsoft research. The archive

⁴<https://www.kaggle.com/c/dogs-vs-cats>

has 12500 images for each class with varying dimensions.

2. **International Skin Imaging Collaboration⁵ (ISIC)**: It is a Malignant and Benign classification dataset of dermoscopy images which is provided in ISIC archive. We extracted 800 and 320 images from each class in training and testing dataset respectively.
3. **Linnaeus 5 dataset⁶**: It is provided by Chaladze, consisting of 256×256 -dimensional true-color images belong to 5 different classes namely, "berry, bird, dog, flower and other". The dataset contains a total of 6000 training images (1200 for each class) and 2000 testing images (400 for each class).
4. **CIFAR10⁷**: This dataset is originally presented by Alex et al. consisting of 32×32 dimensional images for 10 different classes. The resolutions of the images are very low and do not exist in real-time scenarios. Therefore, we extracted images of the same classes from the Flickr image database but of high resolutions. The training and testing sets consist ≈ 2100 and ≈ 500 images respectively. The classes are "airplane, automobile, bird, cat, deer, dog, frog, horse, ship and truck."

Implementation details: Codes are implemented using Keras API running on 64-bit Ubuntu 16.04 OS, over a Dell PC with Intel(R) core *i7* – 3770 3.40 GHz processor, 8 GB of RAM. Each image in all datasets are resized to $299 \times 299 \times 3$. The model is trained on NVidia Kepler GPU over batch-size of 32 for 100 epochs with Adam optimizer [39]. The *imagenet* weights and base learning rate of 10^{-4} are initialized. For experimental analysis, we utilize the architecture, *VGG16* [13], *ResNet50* [14], *NasNet* [40] and *InceptionV3* [41].

7.2. Qualitative results

The visual representation of encrypted images obtained with secret keys $\mathcal{K}$ are presented in Fig. 6. The first row Fig. 6 (a) represents the secret images extracted randomly from above-defined datasets. The image depicted in Fig. 6 (b)-(e) indicate the encrypted images obtained with POB parameters as

⁵<https://www.isic-archive.com/#!/topWithHeader/onlyHeaderTop/gallery>

⁶<http://chaladze.com/l5/>

⁷<https://www.cs.toronto.edu/~kriz/cifar.html>

Figure 6: Examples of the encrypted images with varying encryption keys $\mathcal{K}$ (top-bottom). (a) Original images, (b)-(e) the encrypted images corresponding to images in (a) with POB parameters $n = 17$ and $k = 13$ and permutation matrices of dimension 13×13 in (b), 23×23 (c), 37×37 in (d), and 47×47 in (e).

$n = 17$ and $k = 13$ and permutation matrices of varying dimensions. It can be observed that the probability of extracting the object shape and features in the encrypted images is nearly complicated. The distortion and decay in information increases with the increment in dimensions of block-size ($\rho_1 \times \rho_2$). Moreover, the encrypted images corresponding to cat and dog are visually similar; however, representing distinct classes.

In Fig. 6, it can be observed that the encrypted images have similar intensity values as that of secret image I at some pixel locations. In other words, the encrypted forms of I leaks a small amount of image's visual information. It is happening due to the order-preserving property of the permutation ordered binary (POB) number system. It can also be noted that the value of parameters (n, k) in POB number system (ref. Eq. 5) is same for all images

of a dataset specifically $n = 17$ and $k = 13$. The information leakage can be reduced by considering different combinations of varying values of parameters n and k for different blocks, provided in a bounded range. For instance, the combinations of $n \in [11, 21]$ and $k \in [9, 17]$.

Also, we normalized all the POB-spaced images I_{POB} into a fixed range namely $[0, 255]$. All the encrypted images have the same minimum and maximum intensity values, namely 0 and 255, irrespective of POB parameters and block-size. This information leakage can be reduced by normalizing the POB-spaced encrypted images into the range of the non-encrypted image I (ref. Eq. 7). It can easily be observed in Figs. 21-26 and Figs. 39-39 depicted in Appendices. However, the other statistical moments such as mean and standard deviation of encrypted images with different block-sizes will be close to each other and distinct from the original image. For instance, consider cat image in Fig. 6 (column - 4) then the original mean and standard deviation are 78.8688 and 13.8414, respectively. In contrast, the mean and standard deviation of encrypted images lie in the range $[102.70, 104.14]$ and $[3.27, 3.89]$ respectively.

7.3. Quantitative analysis

Here, we analyze the recognition accuracy obtained for the above-defined datasets in two scenarios. In the first scenario, the model is trained and tested over encrypted data. In the second scenario, the model is trained over encrypted data and tested over plain data. Note that testing data considered in the latter scenario is the same as that in the first scenario but in plain form. The datasets are encrypted with different secret keys and then, state-of-the-art deep recognition models *VGG16*, *ResNet50*, *NasNet* and *InceptionV3* are used for learning over these encrypted datasets. The secret encryption keys utilized for encrypting data are the same as those in Section 7.2. The data is divided into a ratio of 90% for training and 10% for testing purposes.

7.3.1. Training and testing on encrypted data

Here, we analyze the scenario in which model $\mathcal{M}$ is trained and tested over the encrypted data. The results of experiments in this scenario with over different datasets are reported in Table 4 - 7. It is observed that the recognition results over the encrypted data obtained with small values of ρ_1 and ρ_2 are high and vice-versa. In resemblance of *no free lunch* theorem, there cannot exist a unique DL model that outperforms for all datasets and, therefore, resultant accuracies vary with different architectures. For instance

in Table 6, the recognition accuracies over encrypted data obtained by permutation matrices of dimension 37×37 and, $n = 17$ and $k = 13$, are high as compared to others.

To analyze the recognition robustness of *SecureDL*, we compared the achieved recognition accuracies for the encrypted data with the existing image encryption schemes. In other words, we encrypt each dataset by shuffling image pixels, and the image encryption schemes proposed by Tong et al. [42] and Hua et al. [43]. Then, each of the DL model namely *VGG16*, *ResNet50*, *NasNet* and *InceptionV3* are trained and tested for obtained encrypted datasets. The achieved testing accuracies are presented in Table 8. It can easily be observed that *SecureDL* outperforms the existing schemes. In *SecureDL*, the encryption is performed by obfuscating image information in a block-based manner. Each block is shuffled using a different permutation matrix, and in each block, the accumulated noise is bounded within the same block’s maximum and minimum intensity values. In this way, our proposed image encryption scheme de-correlates each block’s pixels independently and preserves some local features. These preserved features play a significant role in learning robust parameters of model $\mathcal{M}$ to give high recognition accuracy. In contrast, the existing image encryption schemes [42, 43] de-correlate the complete image’s pixels simultaneously and do not preserve any feature. It is worth mentioning that these existing encryption schemes are more robust than *SecureDL* for storage purposes only however, they cannot be utilized for processing the encrypted images. The deep learning recognition models for instance *Inception V3* utilizes the preserved local features in *SecureDL* while training and inferencing, but it is not happening with the existing image encryption schemes.

In our scheme, a trade-off between the dimensions of block permutation matrices and accuracy is noticed. In today’s era of data-driven approaches, data is no less than the user’s monetary assets. Thus, a few decrements inaccuracy at the cost of data confidential can be accepted by the user. The improvement in the recognition rate is considered in our future works.

7.3.2. Training on encrypted data and testing on plain data

In this scenario, we consider the threat in which an adversary is likely to extract the class labels related information by utilizing the model $\mathcal{M}$ trained over encrypted data. It can be possible by transfer learning over plain (non-encrypted) data. This threat-likelihood is resistible if model $\mathcal{M}$ gives low confidence rates over plain data. Given this, the recognition rates

Table 4: Accuracy comparison over Kaggle dataset - trained and tested in ED.

Model	Block-size				PD
	13×13	23×23	37×37	47×47	
VGG16 [13]	58.40 %	55.10 %	53.50 %	51.15 %	92.78 %
ResNet50 [14]	62.15 %	58.70 %	56.33 %	54.66 %	94.15 %
NasNet [40]	68.37 %	66.25 %	65.75 %	62.43 %	97.22 %
<i>InceptionV3</i> [41]	81.60 %	78.52 %	74.12 %	72.34%	98.82 %

Table 5: Accuracy comparison over ISIC dataset - trained and tested in ED.

Model	Block-size				PD
	13×13	23×23	37×37	47×47	
VGG16 [13]	57.18 %	56.83 %	55.27 %	53.42 %	89.45 %
ResNet50 [14]	87.31 %	84.91 %	84.77 %	84.25 %	95.90 %
NasNet [40]	82.03 %	81.74 %	81.84 %	71.29 %	97.43 %
<i>InceptionV3</i> [41]	88.47 %	86.61 %	83.59 %	81.06 %	98.95 %

Table 6: Accuracy comparison over Linnaeus 5 dataset - trained and tested in ED.

Model	Block-size				PD
	13×13	23×23	37×37	47×47	
VGG16 [13]	40.03 %	34.47 %	68.16 %	25.78 %	94.53 %
ResNet50 [14]	74.22 %	72.19 %	77.73 %	71.78 %	97.85 %
NasNet [40]	67.58 %	65.72 %	75.18 %	72.27 %	98.05 %
<i>InceptionV3</i> [41]	76.56 %	79.00 %	79.18 %	75.00 %	98.94 %

are evaluated over the trained model in Section 7.3.1. The obtained results are presented in Figs. 7 - 10. The recognition accuracy over Kaggle and ISIC data varies in the range of 40% – 50%, which is an obvious outcome in binary classification tasks. However, the accuracies for Linnaeus 5 and CIFAR10 datasets are in the range of 18% – 28% and 8% – 15% respectively.

Table 7: Accuracy comparison over CIFAR10 dataset - trained and tested in ED.

Model	Block-size				PD
	13×13	23×23	37×37	47×47	
VGG16 [13]	42.23 %	41.76 %	34.38 %	40.43 %	89.26 %
ResNet50 [14]	63.09 %	62.57 %	56.25 %	52.93 %	95.99 %
NasNet [40]	49.22 %	46.15 %	44.92 %	37.70 %	96.66 %
<i>InceptionV3</i> [41]	65.43 %	63.74 %	54.69 %	57.97 %	97.05 %

Table 8: Comparison with existing encryption schemes - trained and tested in ED.

	Recognition accuracy			
	Kaggle Cat - Dog	Skin Lesion	Linnaeus 5	CIFAR10
Original	98.82 %	98.95 %	98.94 %	97.05 %
Image shuffling	13.79 %	15.22 %	11.73 %	9.06 %
Tong et al. [42]	17.23 %	18.10 %	11.87 %	6.13 %
Hua et al. [43]	15.85 %	14.76 %	12.94 %	7.76 %
<i>SecureDL</i> Block - 47×47	72.34 %	81.06 %	75.00 %	57.97 %

8. Performance analysis

This section discusses the computational and storage overheads of our scheme and compares it with the existing algorithms. A cloud service needs to be computationally less expensive and require minimum storage space.

8.1. Computation overhead

In *SecureDL*, the processing required for the generation of secret keys $\mathcal{K} = \{n, k, \pi_1, \pi_2, \dots, \pi_T\}$ using function *KeyGen* and encrypting data $\mathcal{D}$ at user-end are one-time operations. The computational time required for the functioning of these operations depends upon image size as well as key $\mathcal{K}$. Further, the mathematical operations to be evaluated during the learning of model $\mathcal{M}$ are performed by CSP. Each service provider has different computational resources such as quantity and configurations of GPUs and hence,

Figure 7: Accuracy comparison over Kaggle dataset - trained in ED and tested in PD.

Figure 8: Accuracy comparison over ISIC dataset - trained in ED and tested in PD.

computational time varies from one CSP to another.

For an instance, the time taken by our scheme to encrypt an image over PC of configuration explained in Section 7 is analyzed. Consider a standard Lena true-color image (I) of dimension 299×299 . The time required for

Figure 9: Accuracy comparison over Linnaeus5 data - trained in ED and tested in PD.

Figure 10: Accuracy comparison over CIFAR10 dataset - trained in ED and tested in PD.

transforming all pixel intensity values of I to POB-values and generating 529 permutation matrices, $\pi = \{\pi_1, \pi_2, \dots, \pi_{529}\}$, each of size 13×13 is 0.0038 and 0.0501 respectively. Further, the time required for permuting the POB-spaced image (I_{POB}) using π is 0.3530 second. Thus, user needs to give a total

of only $0.0038 + 0.0501 + 0.3530 = 0.4069$ second to encrypt a secret image of dimension 299×299 . Hence, the required time for encrypting dataset of 10000 images = $0.4069 * 10000 = 4069$ seconds only. However, In view of FHE-based model [10], user can encrypt and encode 24193 images of MNIST type in an hour on a PC with window 10 OS, a single Intel Xeon E5-1620 CPU running at 3.5GHz, with 16GB of RAM. It is worth to note that MNIST dataset contains binary images of size 28×28 only whereas our images are RGB-color images of size 299×299 .

8.2. Storage overhead

Similar to computational overhead, there is a one-to-one correspondence between storage overhead and image size. We experimentally proved that the space required for an encrypted image obtained by our scheme is the same as that of the original image and, henceforth, does not increase storage space. Consider an true-color image of size $300 \times 300 \times 3$ pixels. The storage requirement is $300 \times 300 \times 3 \times 8$ bits or 0.2575 MB space.

In the first level of our encryption scheme, POB number system transforms each 8-bit pixel value into an n -bit POB-value ($n \gg 8$). Thus, the space required for the encrypted image is $300 \times 300 \times 3 \times n$ bits. The second level of security shuffles image pixels intensity values based on permutation matrices $\pi = \{\pi_1, \pi_2, \dots, \pi_T\}$, in which size of the POB-spaced image remains the same i.e., $300 \times 300 \times 3 \times n$ bits. As discussed in Section 4, this modified image is normalized, to increase computational efficiency, in the range $[0, 255]$, which again drops down the image size. Thus, size of the encrypted image becomes same as of the original image i.e., $\frac{300 \times 300 \times 3 \times n \times 8}{n} \text{ bits} = 300 \times 300 \times 3 \times 8 \text{ bits} = 0.2575 \text{ MB}$. Hence, *SecureDL* does not expand the data size whereas, existing schemes such as *CryptoNets* [10] and *CryptoDL* [44] drastically increase the data size by 60 and 34 times respectively over simple MNIST data. The storage comparison is presented in Table 9.

Table 9: Comparison of storage overhead.

Methods/Dataset	MNIST	CIFAR10 (original)	Dogs vs. Cats
<i>SecureDL</i>	10.8 MB	176.3 MB	589.6 MB
CryptoNets [10]	595.5 MB	N/ A	N/ A
CryptoDL [44]	339.7 MB	1803 MB	N/ A

9. Conclusion and future work

We propose a novel image encryption scheme using permutation ordered binary number system and block-permutation matrices. The scheme partially preserves the trichotomous law of positive integers. Extending this scheme, we modeled *SecureDL*, a privacy-preserving image recognition system for the encrypted data over the cloud infrastructure. It overcomes the computational and storage overheads that occurred by FHE-based schemes. *SecureDL* is proved to be secure in probabilistic view-point and various cryptographic attacks. The model is evaluated over benchmark datasets: Kaggle Dogs vs. Cats, ISIC dermoscopy images, Linnaeus 5, and CIFAR10. The achieved accuracy over encrypted data is close to that of the plain domain. The preservation of trichotomy law in the proposed scheme allows the extension of image recognition to scene text detection, image segmentation, and video classification.

Acknowledgment

This research was supported by University Grant Commission (UGC), INDIA reference grant number: 21/12/2014(ii)EU-V, 2121440593.

References

- [1] I. Sutskever, O. Vinyals, Q. V. Le, Sequence to sequence learning with neural networks, in: Advances in Neural Information Processing Systems, 2014, pp. 3104–3112.
- [2] M. M. Hassan, M. Z. Uddin, A. Mohamed, A. Almogren, A robust human activity recognition system using smartphone sensors and deep learning, Future Generation Computer Systems 81 (2018) 307–313.
- [3] A. Esteva, B. Kuprel, R. A. Novoa, J. Ko, S. M. Swetter, H. M. Blau, S. Thrun, Dermatologist-level classification of skin cancer with deep neural networks, Nature 542 (2017) 115.
- [4] A. Broumandnia, The 3d modular chaotic map to digital color image encryption, Future Generation Computer Systems (2019).

- [5] K. Muhammad, R. Hamza, J. Ahmad, J. Lloret, H. Wang, S. W. Baik, Secure surveillance framework for iot systems using probabilistic image encryption, *IEEE Transactions on Industrial Informatics* 14 (2018) 3679–3689.
- [6] X. Ma, F. Zhang, X. Chen, J. Shen, Privacy preserving multi-party computation delegation for deep learning in cloud computing, *Information Sciences* 459 (2018) 103–116.
- [7] R. Yonetani, V. Naresh Boddeti, K. M. Kitani, Y. Sato, Privacy-preserving visual learning using doubly permuted homomorphic encryption, in: *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2040–2050.
- [8] Y. Aono, T. Hayashi, L. Wang, S. Moriai, et al., Privacy-preserving deep learning via additively homomorphic encryption, *IEEE Transactions on Information Forensics and Security* 13 (2018) 1333–1345.
- [9] P. Xie, M. Bilenko, T. Finley, R. Gilad-Bachrach, K. Lauter, M. Naehrig, Crypto-nets: Neural networks over encrypted data, *arXiv preprint arXiv:1412.6181* (2014).
- [10] R. Gilad-Bachrach, N. Dowlin, K. Laine, K. Lauter, M. Naehrig, J. Wernsing, Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy, in: *International Conference on Machine Learning*, 2016, pp. 201–210.
- [11] R. L. Rivest, L. Adleman, M. L. Dertouzos, On data banks and privacy homomorphisms, *Foundations of secure computation* 4 (1978) 169–180.
- [12] S. Fau, R. Sirdey, C. Fontaine, C. Aguilar-Melchor, G. Gogniat, Towards practical program execution over fully homomorphic encryption schemes, in: *Eighth International Conference on P2P, Parallel, Grid, Cloud and Internet Computing*, IEEE, 2013, pp. 284–290.
- [13] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, *arXiv preprint arXiv:1409.1556* (2014).
- [14] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: *The IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

- [15] A. Lathey, P. K. Atrey, N. Joshi, Homomorphic low pass filtering on encrypted multimedia over cloud, in: IEEE Seventh International Conference on Semantic Computing, IEEE, 2013, pp. 310–313.
- [16] S. Jha, L. Kruger, P. McDaniel, Privacy preserving clustering, in: European Symposium on Research in Computer Security, Springer, 2005, pp. 397–417. doi:10.1007/11555827_23.
- [17] C. Orlandi, A. Piva, M. Barni, Oblivious neural network computing via homomorphic encryption, EURASIP Journal on Information Security 2007 (2007) 037343.
- [18] R. Bost, R. A. Popa, S. Tu, S. Goldwasser, Machine learning classification over encrypted data., in: NDSS, 2015.
- [19] J. R. Padilla-López, A. A. Chaaraoui, F. Flórez-Revuelta, Visual privacy protection methods: A survey, Expert Systems with Applications 42 (2015) 4177–4195.
- [20] A. Lathey, P. K. Atrey, Image enhancement in encrypted domain over cloud, ACM Transactions on Multimedia Computing, Communications, and Applications 11 (2015) 38.
- [21] V. K. Tanwar, A. S. Rajput, B. Raman, R. Bhargava, Privacy preserving image scaling using 2d bicubic interpolation over the cloud, in: IEEE International Conference on Systems, Man, and Cybernetics, IEEE, 2018, pp. 2073–2078.
- [22] W. Q. Yan, M. S. Kankanhalli, Face search in encrypted domain, in: Pacific-rim Symposium on Image and Video Technology, Springer, 2015, pp. 775–790. doi:10.1007/978-3-319-29451-3_61.
- [23] Y. Rahulamathavan, R. C.-W. Phan, J. A. Chambers, D. J. Parish, Facial expression recognition in the encrypted domain based on local fisher discriminant analysis, IEEE Transactions on Affective Computing 4 (2013) 83–92.
- [24] S. M. SaghaianNejadEsfahani, Y. Luo, S.-c. S. Cheung, Privacy protected image denoising with secret shares, in: IEEE International Conference on Image Processing, IEEE, 2012, pp. 253–256. doi:10.1109/ICIP.2012.6466843.

- [25] A. Pedrouzo-Ulloa, J. R. Troncoso-Pastoriza, F. Pérez-González, Image denoising in the encrypted domain, in: IEEE International Workshop on Information Forensics and Security, IEEE, 2016, pp. 1–6. doi:10.1109/WIFS.2016.7823916.
- [26] J. Vaidya, C. Clifton, Privacy preserving association rule mining in vertically partitioned data, in: Proceedings of the SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2002, pp. 639–644. doi:10.1145/775047.775142.
- [27] M. Upmanyu, A. Namboodiri, K. Srinathan, C. Jawahar, Efficient privacy preserving k-means clustering, in: Pacific-Asia Workshop on Intelligence and Security Informatics, Springer, 2010, pp. 154–166. doi:10.1007/978-3-642-13601-6_17.
- [28] G. Jagannathan, R. Wright, Privacy-preserving distributed k-means clustering over arbitrarily partitioned data, in: Proceedings of the SIGKDD International Conference on Knowledge Discovery in Data Mining, ACM, 2005, pp. 593–599. doi:10.1145/1081870.1081942.
- [29] Q. Zhang, L. Yang, Z. Chen, P. Li, Pphopcm: Privacy-preserving high-order possibilistic c-means algorithm for big data clustering with cloud computing, IEEE Transactions on Big Data (2017).
- [30] K. Xing, C. Hu, J. Yu, X. Cheng, F. Zhang, Mutual privacy preserving *k*-means clustering in social participatory sensing, IEEE Transactions on Industrial Informatics 13 (2017) 2066–2076.
- [31] R. Shokri, V. Shmatikov, Privacy-preserving deep learning, in: Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, ACM, 2015, pp. 1310–1321.
- [32] A. Sreekumar, S. B. Sundar, An efficient secret sharing scheme for n out of n scheme using pob-number system, Hack. in 2009 (2009) 33.
- [33] P. Singh, N. Agarwal, B. Raman, Don’t see me, just filter me: Towards secure cloud based filtering using shamir’s secret sharing and pob number system, in: Proceedings of the Tenth Indian Conference on Computer Vision, Graphics and Image Processing, ACM, 2016, pp. 12:1–12:8. URL: <http://doi.acm.org/10.1145/3009977.3010036>. doi:10.1145/3009977.3010036.

- [34] D. S. Dummit, R. M. Foote, Abstract algebra, volume 3, Wiley Hoboken, 2004.
- [35] G. Chen, Y. Mao, C. K. Chui, A symmetric image encryption scheme based on 3d chaotic cat maps, *Chaos, Solitons & Fractals* 21 (2004) 749–761.
- [36] C. Fu, J. Chen, H. Zou, W. Meng, Y. Zhan, Y. Yu, A chaos-based digital image encryption scheme with an improved diffusion strategy, *Optics Express* 20 (2012) 2363–2378. doi:10.1364/OE.20.002363.
- [37] L. Teng, X. Wang, A bit-level image encryption algorithm based on spatiotemporal chaotic system and self-adaptive, *Optics Communications* 285 (2012) 4048–4054. doi:10.1016/j.optcom.2012.06.004.
- [38] A. S. Rajput, B. Raman, Cryptoct: towards privacy preserving color transfer and storage over cloud, *Multimedia Tools and Applications* (2018) 1–23.
- [39] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, *arXiv preprint arXiv:1412.6980* (2014).
- [40] B. Zoph, V. Vasudevan, J. Shlens, Q. V. Le, Learning transferable architectures for scalable image recognition, *arXiv preprint arXiv:1707.07012* 2 (2017).
- [41] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna, Rethinking the inception architecture for computer vision, in: *The IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2818–2826.
- [42] X.-J. Tong, M. Zhang, Z. Wang, Y. Liu, H. Xu, J. Ma, A fast encryption algorithm of color image based on four-dimensional chaotic system, *Journal of Visual Communication and Image Representation* 33 (2015) 219–234.
- [43] Z. Hua, Y. Zhou, H. Huang, Cosine-transform-based chaotic system for image encryption, *Information Sciences* 480 (2019) 403–419.
- [44] E. Hesamifard, H. Takabi, M. Ghasemi, Cryptodl: Deep neural networks over encrypted data, *arXiv preprint arXiv:1711.05189* (2017).

Appendices

10. Histogram Attack

Here, we provide some more real-time images and compare their red, green, and blue component histograms with the encrypted image's histograms. It can be seen in Figs. 11 - 15. It can be observed that the histograms of the original images vary drastically with each other, whereas the histograms of encrypted images are approximately similar. Further, we perform the histogram equalization over the histograms of the encrypted images in Figs. 16 - 20. We observed that the encrypted image's equalized histograms are similar to the histograms of another encrypted image; however, they are significantly different from the histograms of the original images. It indicates that an adversary's negligible amount of information may be extracted, but that information is not sufficient for extracting the whole original image.

11. Statistical analysis

This section computes the block-wise maximum, mean, median, and minimum of the original and the corresponding encrypted images.

In Figs. 21 - 26, the block-wise maximum intensity value image is presented for original image in (a) and its corresponding encrypted images with POB parameters ($n = 17, k = 13$ with block size 13×13 in (b), 37×37 in (c) and 47×47 in (d). The block-size considered for computing maximum values is 4×4 in second column, 16×16 in third column and 64×64 in fourth column. Similarly, we computed block-wise mean, median and minimum for each of original and encrypted images (as defined for maximum function) in Figs. 27 - 32, Figs. 33 - 38, and Figs. 39 - 44 respectively. It is observed that the obtained images for each of maximum, mean, median and minimum function for original and encrypted images are drastically different and, do not reveal information of original images.

Figure 11: Histogram Analysis. (a) secret image I , (b)-(d) histograms corresponding to red-green-blue channels of (a), (e) encrypted image I_{Enc} ($n = 17, k = 13, \rho_1 \times \rho_2 = 47 \times 47$), and (f)-(h) histograms corresponding to red-green-blue channels of (e).

Figure 12: Histogram Analysis. (a) secret image I , (b)-(d) histograms corresponding to red-green-blue channels of (a), (e) encrypted image I_{Enc} ($n = 17, k = 13, \rho_1 \times \rho_2 = 47 \times 47$), and (f)-(h) histograms corresponding to red-green-blue channels of (e).

Figure 13: Histogram Analysis. (a) secret image I , (b)-(d) histograms corresponding to red-green-blue channels of (a), (e) encrypted image I_{Enc} ($n = 17, k = 13, \rho_1 \times \rho_2 = 47 \times 47$), and (f)-(h) histograms corresponding to red-green-blue channels of (e).

Figure 14: Histogram Analysis. (a) secret image I , (b)-(d) histograms corresponding to red-green-blue channels of (a), (e) encrypted image I_{Enc} ($n = 17, k = 13, \rho_1 \times \rho_2 = 47 \times 47$), and (f)-(h) histograms corresponding to red-green-blue channels of (e).

Figure 15: Histogram Analysis. (a) secret image I , (b)-(d) histograms corresponding to red-green-blue channels of (a), (e) encrypted image I_{Enc} ($n = 17, k = 13, \rho_1 \times \rho_2 = 47 \times 47$), and (f)-(h) histograms corresponding to red-green-blue channels of (e).

Figure 16: Comparison of histograms of original image in (a) and the encrypted image in (b) with the equalized histograms of encrypted image in (c).

Figure 17: Comparison of histograms of original image in (a) and the encrypted image in (b) with the equalized histograms of encrypted image in (c).

Figure 18: Comparison of histograms of original image in (a) and the encrypted image in (b) with the equalized histograms of encrypted image in (c).

Figure 19: Comparison of histograms of original image in (a) and the encrypted image in (b) with the equalized histograms of encrypted image in (c).

Figure 20: Comparison of histograms of original image in (a) and the encrypted image in (b) with the equalized histograms of encrypted image in (c).

Figure 21: Comparison of block-wise maximum intensity value with block-size 4×4 (second column), 16×16 (third column) and 64×64 (fourth column). (a) Original image, (b) encrypted image with patch size 13×13 , (c) encrypted image with patch size 37×37 , and (d) encrypted image with patch size 47×47 .

Figure 22: Comparison of block-wise maximum intensity value with block-size 4×4 (second column), 16×16 (third column) and 64×64 (fourth column). (a) Original image, (b) encrypted image with patch size 13×13 , (c) encrypted image with patch size 37×37 , and (d) encrypted image with patch size 47×47 .

Figure 23: Comparison of block-wise maximum intensity value with block-size 4×4 (second column), 16×16 (third column) and 64×64 (fourth column). (a) Original image, (b) encrypted image with patch size 13×13 , (c) encrypted image with patch size 37×37 , and (d) encrypted image with patch size 47×47 .

Figure 24: Comparison of block-wise maximum intensity value with block-size 4×4 (second column), 16×16 (third column) and 64×64 (fourth column). (a) Original image, (b) encrypted image with patch size 13×13 , (c) encrypted image with patch size 37×37 , and (d) encrypted image with patch size 47×47 .

Figure 25: Comparison of block-wise maximum intensity value with block-size 4×4 (second column), 16×16 (third column) and 64×64 (fourth column). (a) Original image, (b) encrypted image with patch size 13×13 , (c) encrypted image with patch size 37×37 , and (d) encrypted image with patch size 47×47 .

Figure 26: Comparison of block-wise maximum intensity value with block-size 4×4 (second column), 16×16 (third column) and 64×64 (fourth column). (a) Original image, (b) encrypted image with patch size 13×13 , (c) encrypted image with patch size 37×37 , and (d) encrypted image with patch size 47×47 .

Figure 27: Comparison of block-wise mean intensity value with block-size 4×4 (second column), 16×16 (third column) and 64×64 (fourth column). (a) Original image, (b) encrypted image with patch size 13×13 , (c) encrypted image with patch size 37×37 , and (d) encrypted image with patch size 47×47 .

Figure 28: Comparison of block-wise mean intensity value with block-size 4×4 (second column), 16×16 (third column) and 64×64 (fourth column). (a) Original image, (b) encrypted image with patch size 13×13 , (c) encrypted image with patch size 37×37 , and (d) encrypted image with patch size 47×47 .

Figure 29: Comparison of block-wise mean intensity value with block-size 4×4 (second column), 16×16 (third column) and 64×64 (fourth column). (a) Original image, (b) encrypted image with patch size 13×13 , (c) encrypted image with patch size 37×37 , and (d) encrypted image with patch size 47×47 .

Figure 30: Comparison of block-wise mean intensity value with block-size 4×4 (second column), 16×16 (third column) and 64×64 (fourth column). (a) Original image, (b) encrypted image with patch size 13×13 , (c) encrypted image with patch size 37×37 , and (d) encrypted image with patch size 47×47 .

Figure 31: Comparison of block-wise mean intensity value with block-size 4×4 (second column), 16×16 (third column) and 64×64 (fourth column). (a) Original image, (b) encrypted image with patch size 13×13 , (c) encrypted image with patch size 37×37 , and (d) encrypted image with patch size 47×47 .

Figure 32: Comparison of block-wise mean intensity value with block-size 4×4 (second column), 16×16 (third column) and 64×64 (fourth column). (a) Original image, (b) encrypted image with patch size 13×13 , (c) encrypted image with patch size 37×37 , and (d) encrypted image with patch size 47×47 .

Figure 33: Comparison of block-wise median intensity value with block-size 4×4 (second column), 16×16 (third column) and 64×64 (fourth column). (a) Original image, (b) encrypted image with patch size 13×13 , (c) encrypted image with patch size 37×37 , and (d) encrypted image with patch size 47×47 .

Figure 34: Comparison of block-wise median intensity value with block-size 4×4 (second column), 16×16 (third column) and 64×64 (fourth column). (a) Original image, (b) encrypted image with patch size 13×13 , (c) encrypted image with patch size 37×37 , and (d) encrypted image with patch size 47×47 .

Figure 35: Comparison of block-wise median intensity value with block-size 4×4 (second column), 16×16 (third column) and 64×64 (fourth column). (a) Original image, (b) encrypted image with patch size 13×13 , (c) encrypted image with patch size 37×37 , and (d) encrypted image with patch size 47×47 .

Figure 36: Comparison of block-wise median intensity value with block-size 4×4 (second column), 16×16 (third column) and 64×64 (fourth column). (a) Original image, (b) encrypted image with patch size 13×13 , (c) encrypted image with patch size 37×37 , and (d) encrypted image with patch size 47×47 .

Figure 37: Comparison of block-wise median intensity value with block-size 4×4 (second column), 16×16 (third column) and 64×64 (fourth column). (a) Original image, (b) encrypted image with patch size 13×13 , (c) encrypted image with patch size 37×37 , and (d) encrypted image with patch size 47×47 .

Figure 38: Comparison of block-wise median intensity value with block-size 4×4 (second column), 16×16 (third column) and 64×64 (fourth column). (a) Original image, (b) encrypted image with patch size 13×13 , (c) encrypted image with patch size 37×37 , and (d) encrypted image with patch size 47×47 .

Figure 39: Comparison of block-wise minimum intensity value with block-size 4×4 (second column), 16×16 (third column) and 64×64 (fourth column). (a) Original image, (b) encrypted image with patch size 13×13 , (c) encrypted image with patch size 37×37 , and (d) encrypted image with patch size 47×47 .

Figure 40: Comparison of block-wise minimum intensity value with block-size 4×4 (second column), 16×16 (third column) and 64×64 (fourth column). (a) Original image, (b) encrypted image with patch size 13×13 , (c) encrypted image with patch size 37×37 , and (d) encrypted image with patch size 47×47 .

Figure 41: Comparison of block-wise minimum intensity value with block-size 4×4 (second column), 16×16 (third column) and 64×64 (fourth column). (a) Original image, (b) encrypted image with patch size 13×13 , (c) encrypted image with patch size 37×37 , and (d) encrypted image with patch size 47×47 .

Figure 42: Comparison of block-wise minimum intensity value with block-size 4×4 (second column), 16×16 (third column) and 64×64 (fourth column). (a) Original image, (b) encrypted image with patch size 13×13 , (c) encrypted image with patch size 37×37 , and (d) encrypted image with patch size 47×47 .

Figure 43: Comparison of block-wise minimum intensity value with block-size 4×4 (second column), 16×16 (third column) and 64×64 (fourth column). (a) Original image, (b) encrypted image with patch size 13×13 , (c) encrypted image with patch size 37×37 , and (d) encrypted image with patch size 47×47 .

Figure 44: Comparison of block-wise minimum intensity value with block-size 4×4 (second column), 16×16 (third column) and 64×64 (fourth column). (a) Original image, (b) encrypted image with patch size 13×13 , (c) encrypted image with patch size 37×37 , and (d) encrypted image with patch size 47×47 .