electronics

Review

Privacy-Preserving Deep Neural Network Methods: Computational and Perceptual Methods—An Overview

Raghida El Saj [1,2,*], Ehsan Sedgh Gooya [2], Ayman Alfalou [2] and Mohamad Khalil [1]

[1] CRSI Research Center, Faculty of Engineering, Lebanese University, 1300 Tripoli, Lebanon; mohamad.khalil@ul.edu.lb

[2] L@bISEN, LSL Team, Yncrea Ouest, 29200 Brest, France; ehsan.sedgh-gooya@isen-ouest.yncrea.fr (E.S.G.); ayman.alfalou@isen-ouest.yncrea.fr (A.A.)

* Correspondence: raghida.saj@hotmail.com

Abstract: Privacy-preserving deep neural networks have become essential and have attracted the attention of many researchers due to the need to maintain the privacy and the confidentiality of personal and sensitive data. The importance of privacy-preserving networks has increased with the widespread use of neural networks as a service in unsecured cloud environments. Different methods have been proposed and developed to solve the privacy-preserving problem using deep neural networks on encrypted data. In this article, we reviewed some of the most relevant and well-known computational and perceptual image encryption methods. These methods as well as their results have been presented, compared, and the conditions of their use, the durability and robustness of some of them against attacks, have been discussed. Some of the mentioned methods have demonstrated an ability to hide information and make it difficult for adversaries to retrieve it while maintaining high classification accuracy. Based on the obtained results, it was suggested to develop and use some of the cited privacy-preserving methods in applications other than classification.

Keywords: privacy-preserving; deep neural networks; cryptography

Citation: El Saj, R.; Sedgh Gooya, E.; Alfalou, A.; Khalil, M. Privacy-Preserving Deep Neural Network Methods: Computational and Perceptual Methods—An Overview. *Electronics* **2021**, *10*, 1367. https://doi.org/10.3390/electronics10111367

Academic Editor: Stefanos Kollias

Received: 20 April 2021
Accepted: 18 May 2021
Published: 7 June 2021

1. Introduction

Recently, artificial intelligence (AI) has been significantly developed as it has allowed solving various complex issues in different fields. It has also been possible to carry machine learning (ML) algorithms and deep neural networks (DNN) in cloud environments. Moreover, a huge amount of data are required to achieve high performance and accuracy. Some fields, such as biomedical, military, financial, and surveillance, benefit from AI in their application, but at the same time, they require maintenance of data confidentiality, privacy, and security. Therefore, it becomes necessary to develop privacy-preserving systems. The key idea was to benefit from cryptography in AI applications, which is not a new field of study.

AI and cryptography have met in different applications. AI and ML have been applied in steganography to hide secret information [1–3], in cryptanalysis through the development of new ML-based attacks that attempt to predict keys [4], or evaluate the security of a cryptosystem and quantify the strength of a cipher [5]. They have also been applied to create cryptosystems; the Neural Network (NN) structure containing an input layer, hidden layers, an output layer, and updated weights has been used as a secret key [6]. ML algorithms were used as well to process and classify data in the encrypted domain in order to preserve privacy and security; here lies the privacy-preserving domain.

With the wide diffusion of DNN and their widespread use in many fields, even those which need personal and confidential information, and therefore those which need to maintain confidentiality, privacy-preserving DNN is becoming an urgent challenge. Privacy-preserving DNN development aims to enable the use of unsecured cloud servers in security-critical applications, such as facial recognition, biometric authentication, and medical image analysis. Various methods have been proposed. This article reviews and compares the most relevant and well-known privacy-preserving DNN methods proposed. In this article, these methods will be classified into two types: computational methods and perceptual image encryption methods. Computational methods allow NN computation to be applied to encrypted data, which requires specific modifications and imposes certain limitations on the structure of NN and the activation functions, while perceptual methods protect visual information by creating incomprehensible images that remain directly applied to image processing algorithms and NN. The criteria that will be used in the comparison are the accuracy of the classification and the availability of training and testing the model in the encrypted domain. For perceptual methods, the robustness against attacks, i.e., the ability to reconstruct encrypted data by adversaries, will also be considered as an essential criterion. In addition, the time spent training and testing these methods (if available) will also be taken into account, especially in the comparison between computational and perceptual methods. The objectives of the study and the comparison of these methods are to select one of them, be inspired by it, and further develop a new method that will be used in applications other than classification, while preserving the confidentiality and the security of the data.

The article starts with a definition of the computational methods, then it reviews some privacy-preserving DNN methods based on them, and later it holds a comparison between these methods. Then, in the next section, there will be a review of the most well-known privacy-preserving DNNs based on perceptual image encryption, and there will also be a comparison between these methods. Finally, a general conclusion and some possible tracks for the future will be proposed.

2. Computational Methods

Computational methods enable computation over encrypted data without knowledge of the encrypted information. This can be done using the Homomorphic Encryption (HE) scheme or the Functional Encryption (FE) scheme.

In HE [7], the decryption of the calculation result with encrypted data is the same as the calculation result with unencrypted data. HE supports addition and/or multiplication operations on encrypted messages, so that:

$$Enc(x) * Enc(y) = Enc(x * y), \tag{1}$$

where Enc denotes the encryption function, $*$ denotes the operation supported by HE, addition or multiplication, and x, y denote the sample messages. Therefore, only polynomial functions can be supported in HE scheme.

On the other hand, FE of a function f consists of four essential algorithms [8]:

- Setup: generates the public key mpk and the master secret key msk.
- KeyDrive: outputs the function secret key sk_f for the function f using msk.
- Encrypt: encrypts the sample message x using mpk.
- Decrypt: computes $f(x)$, using mpk, sk_f, and the ciphertext of x generated in *Encrypt* step.

The difference between HE and FE is that the computation output of HE is encrypted since HE evaluates data without decryption, while the computation output of FE is plaintext since FE applies the decryption step to evaluate the encrypted data [9].

Several computational-based methods have been proposed to preserve privacy in ML applications [9,10]. In the following, some privacy-preserving DNNs based on HE and FE will be mentioned.

2.1. CryptoNets

In 2016, CryptoNets [11] were proposed as NNs capable of classifying encrypted data. Some necessary adjustments were applied to the NN to create the CryptoNets. Since HE supports only polynomial functions, all functions in the NN should be polynomial. Thus, all pooling layers have been replaced by a scaled mean pooling layer whose function is $\sum x$, and all activation functions such as *sigmoid* and *ReLu* have been replaced by the square function x^2, except for the last sigmoid activation function which is necessary for the training phase. The modified NN, fully compatible with HE, is trained using unencrypted data. After training, the sigmoid activation function is removed and consecutive layers that use linear transformations only are collapsed to increase efficiency.

CryptoNets apply prediction of data and provide as well the prediction result in the encrypted domain. The user should first encrypt his plain data m to obtain the ciphertext c:

$$c = [[q/t]m + e + hs]_q \,, \tag{2}$$

where the plain data $m \in R_t^n = \mathbb{Z}_t[x]/(x^n + 1)$, the ciphertext $c \in R_q^t = \mathbb{Z}_q[x]/(x^t + 1)$, $\mathbb{Z}_t[x]$ and $\mathbb{Z}_q[x]$ are the rings of polynomials modulo t and q, respectively, n, t, and q are integers, $h = tgf^{-1}$ is the public key, $f = tf' + 1$ is the secret key, and e, s, g, and $f' \in R_q^n$ are random polynomials. Then, the user can send the encrypted data to CryptoNets to be classified and to receive an encrypted result r. The user should finally decrypt r using the secret key f to obtain the unencrypted result d:

$$d = [[\frac{t}{q} fr]]_t \,. \tag{3}$$

The CryptoNets were performed using the Modified National Institute of Standards and Technology (MNIST)) dataset, a dataset containing images of handwritten digits between 0 and 9. The network was trained using 50,000 unencrypted images, and it was tested using the remaining 10,000 images after encoding and encrypting them. The encoding scheme is used to convert the atomic structure of the neural network—real numbers—to the atomic structure of the HE scheme—polynomials. The network mislabeled 105 out of the 10,000 images. Thus, the obtained accuracy was equal to 99%. A single prediction takes 250 s; however, 4096 predictions can be made simultaneously at no additional cost. The details and the results of CryptoNets are shown in Table 1.

Table 1. Details and properties of the computational methods through the availability of the model training (T) and testing (t) using encrypted data, the dataset used, the depth of the NN (number of convolutional layers in the network), the accuracy of the classification, the accuracy of the original model(without modification and using simple images), the training time, and the number of predictions per hour that can be processed (# p/h).

	T	t	Dataset	NN Depth	Accuracy	Original Accuracy	Training Time	# p/h
CryptoNets [11]	○	⋆	MNIST	2	99%	-	-	58,982
CryptoDL [12]	○	⋆	MNIST CIFAR-10	5 8	99.52% 91.5%	99.56% 94.5%	- -	163,840 2524
CryptoNN [13]	⋆	⋆	MNIST	3	95.49%	95.48%	75 h	-

⋆: available, ○: not available, -: not defined in the article.

2.2. *CryptoDL*

In 2017, a solution for running DNN on encrypted data, CryptoDL, was proposed [12]. This technique consists of two basic components: convolutional neural networks (CNNs), and HE, precisely the leveled HE. To make the CNN compatible with HE, pooling layers have been replaced with scaled mean pooling layers, and a new method has been designed to approximate the most common activation functions, *ReLu*, *sigmoid*, and *tangh*, with low degree polynomials. The higher degree polynomial leads to better performance but at a high computational cost; therefore, only polynomials of degree two and three have been used.

The proposed approximation technique is based on the derivative of the activation function instead of the activation function, using polynomials of degree three. This method was compared with the numerical analysis, Taylor series, standard Chebychev polynomials, and modified Chebychev polynomials for approximating the $ReLu$ function, and it achieved the best approximation. The accuracy obtained by training CNN using the derivative-based approximation was 98.52%, while the best accuracy obtained through the application of other methods using the modified Chebichev polynomials was 88.53%. Furthermore, it was proved that the behavior of the polynomial approximation of the $ReLu$ function is robust against changes in the CNN structure; a different CNN structure was trained using this approximation technique and the accuracy obtained was close to the first structure, 98.38%. Furthermore, the different activation functions, $ReLu$, $sigmoid$ and $tangh$, with their polynomial replacement have been compared and the best accuracy has been obtained using the $ReLu$ activation function.

The CryptoDL was performed using the MNIST and the Canadian Institute for Advanced Research, 10 classes (CIFAR-10) datasets. The CIFAR-10 dataset contains color images categorized into 10 classes. The CNN model was trained using 50,000 unencrypted images and tested using the remaining 10,000 images after encryption. For the MNIST dataset, the obtained accuracy was 99.52%, and the network can make 163,840 predictions per hour. On the other hand, the accuracy obtained using a deeper network trained using the CIFAR-10 dataset was 91.5%, and it can make 2524 predictions per hour. This slowness is due to the complexity of the CIFAR-10 dataset compared to the MNIST dataset, and to the depth of the network used. The details and the results of CryptoDL are shown in Table 1.

2.3. *CryptoNN*

In 2019, CryptoNN [13] was proposed as a framework that supports both training and inference phases over encrypted data. This was possible due to the secure matrix computation based on functional encryption. A functional encryption scheme for an inner product functionality $f(x,y)$ FEIP was adopted, where n is the length of data vectors x and y:

$$f(x,y) = \sum_{i=1}^{n}(x_i y_i). \tag{4}$$

Another functional encryption scheme for basic operations $f_\Delta(x,y)$ FEBO was proposed, where Δ can be addition, substraction, multiplication, or division:

$$f_\Delta = x\Delta y. \tag{5}$$

FEBO has different approaches to key generation according to the arithmetic computation. Thus, according to Δ, the function derived key sk_{f_Δ} is computed at the KeyDrive step, and the result is computed at the Decrypt step.

The CryptoNN framework consists of three entities (see Figure 1):

- The authority: generates the secret key msk, the public key mpk, and the function secret key sk_f (KeyDrive step).
- The client: preprocesses and encrypts the data—input (x) and labels (y)—using mpk and sends them to the server (Encrypt step). The labels must first be coded using the one-hot method, and then mapped to a random vector number r whose components are r_i.
- The server: trains and tests the NN model using the data received from the client(s). Having the data and the first hidden layer in the feed-forward process, and the labels and the output layer in the back-propagation process, the server gets from the authority the sk_f corresponding to the specific function and then decrypts the result of the function (Decrypt step). The server can continue the feed-forward and back-propagation processes normally. The output of the network is p_i, which is the probability that the data x belongs to the class i.

Accordingly, the proposed scheme inserts two rounds of secure computation; at the beginning of the feed-forward process known as secure feed-forward, and at the beginning of the back-propagation process known as secure back-propagation/evaluation.

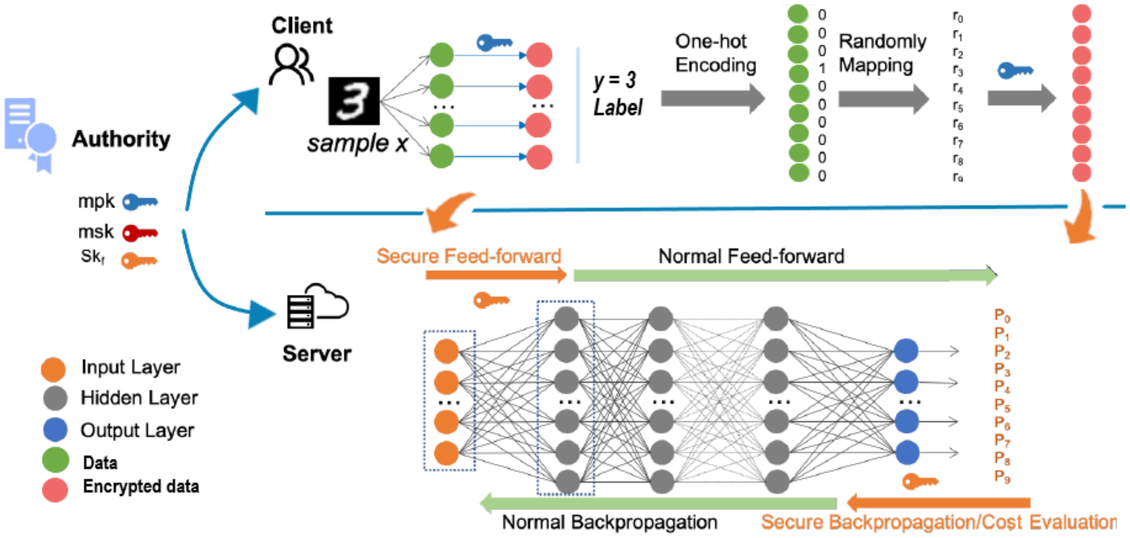


Figure 1. CryptoNN framework, reproduced from [13].

A concrete case of CryptoNN, i.e., CryptoCNN using the LetNet-5 architecture which includes five hidden layers, was created and used. The secure feed-forward takes place in the first hidden layer, i.e., the convolutional layer, while the secure back-propagation takes place at the output layer. The CryptoCNN was performed using the MNIST dataset. It was trained using a training set of 60,000 examples and was tested using a test set of 10,000 examples. The accuracy obtained was 95.49%, while the accuracy of the original LetNet-5 was 95.48%. The training time for two epochs of the CryptoCNN was 75 h, as compared to only 4 h for the original LetNet-5. The details and the results of CryptoNN are shown in Table 1.

2.4. Comparison

The computational methods aim to use NN on encrypted data in order to preserve privacy. They encrypt data using known encryption methods.

CryptoNets and CryptoDL modify the NN to be able to process encrypted data and to be compatible with the encryption method used. These modifications affect the network's performance in terms of computational complexity; they lead to a significant latency in the prediction. The computational complexity and the prediction latency are increased due to the computation of all functions using nested additions and multiplications, and due to the large size of the encrypted data as compared to the unencrypted data (the encrypted data are one to three times larger in magnitude than the unencrypted data [11]), and thus, due to the large amount of data transferred (hundreds of MB).

CryptoDL attempts to improve the performance and the latency of CryptoNets, and to apply deeper NN on encrypted data; but also, it showed it showed certain limits. The main differences between CryptoNets and CryptoDL are the activation functions and their approximation techniques. The approximation of *sigmoid* was used as the activation function in CryptoNets, while CryptoDL compared different activation functions and finally adopted the approximation of the *ReLu* function. The importance and the influence of choosing the appropriate activation function with the appropriate approximation technique were revealed through the significant increase in the number of predictions per hour, as shown in Table 1.

Although CryptoDL gave a better performance with a computational cost that is lower than CryptoNets on the MNIST dataset, CryptoDL has limitations in the number of hidden layers and the complexity of the dataset used, just like CryptoNets. These limitations were revealed through a sharp decrease in the number of predictions per hour, as well as through the noticeable difference between the obtained accuracy and the original one, when using the CIFAR-10 dataset and a deeper NN (see Table 1).

CryptoNN processes NN computations while preserving their confidentiality and security using an encryption scheme that is different than HE used by cryptoNets and CryptoDL. This privacy preservation is achieved through secure matrix computation based on FE, not through special computation requiring the modification of NN's functions and structure. While CryptoNets and CryptoDL allow the classification of encrypted data using NNs that are previously trained with unencrypted data, CryptoNN allows both training and testing on encrypted data. The problem with the CryptoNN is the need for frequent communications with the authority to generate and obtain the corresponding keys. As a result of this problem, along with the time-consuming cryptographic computations, CryptoNN requires a much longer learning time than the original network (73 more hours).

All of the computational methods cited have been tested only on simple datasets, MNIST and CIFAR-10, so their scalability is not certain yet. Hence, more complex and realistic datasets should be used and tested. Despite the use of simple datasets and not very deep networks in testing these methods, their complexity, and latency appeared clearly. This highlights the difficulty of applying these methods to state-of-the-art DNN and to solving more realistic problems.

3. Perceptual Methods

Perceptual image encryption methods protect images by generating visually-protected images that have pixel values, not ciphertexts. The protected images can therefore be directly applied to image processing algorithms. Many encryption methods have been proposed, some of which can be applied to traditional ML algorithms such as support vector machines and random forest [14,15]. The best-known perceptual methods applied to DNN as privacy-preserving DNN will be mentioned in the following.

3.1. Tanaka's Scheme

In 2018, Tanaka proposed a block-based encryption scheme known as Tanaka's scheme [16].

As shown in Figure 2, the 8-bit pixel RGB image is divided first into $M \times M$ blocks, and then each block is split to the upper and the lower 4-bit pixel values to form 6-channel image blocks. The intensities of the pixel values are randomly inverted and shuffled using the secret key $K = \{K_{inv}, K_{shuff}\}$, where K_{inv} and K_{shuff} are respectively the secret keys for inversion and shuffling. Finally, the 6-channel blocks are reformed into 3-channel blocks, and the encrypted image is obtained. The key space N_{Tanaka} of Tanaka's scheme is given by [17], where . is a dot multiplication:

$$N_{Tanaka} = 96! \,.\, 2^{96}. \tag{6}$$

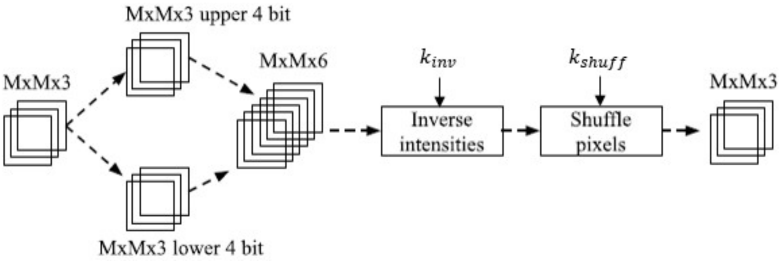


Figure 2. Tanaka's block-based encryption scheme. © 2019 IEEE. Reprinted, with permission, from [18].

To apply the encrypted images to DNN, an adaptation network should be added prior to the utilized DNN to reduce the influence of image encryption. The adaptation includes a convolution layer with $M \times M$ sized filter and $M \times M$ stride, several network-in-network layers, and a sub-pixel convolution. After the adaptation, any kind of network can be followed.

Tanaka's scheme was performed using CIFAR-10 and CIFAR-100 datasets. The CIFAR-100 dataset is identical to CIFAR-10, except that it has 100 classes. A pyramidal residual network was used after the adaptation network, and the block size M was set at four. The accuracy obtained for CIFAR-10 was 86.3% and that for CIFAR-100 was 56.8%, compared to 88.4% for CIFAR-10 and 59.1% for CIFAR-100 using plain images. Moreover, it provides robustness against adversarial attacks, where the images are designed to make the NN misclassify with high confidence [18]. However, the visual information of the encrypted images can be reconstructed using Generative Adversarial Network attack (GAN-attack) and Inverse Transformation Network attack (ITN-attack) [17]. The details and the results of Tanaka's scheme are shown in Table 2.

Table 2. Details and properties of the perceptual methods through the availability of the model training (T) and testing (t) using encrypted data, the dataset used, the classifier network used to classify the encrypted data, the accuracy of the classification, the accuracy of the original model (using simple images), and the robustness against various COA.

		T	t	Dataset	Classifier Network	Accuracy	Original Accuracy	Attacks		
								FR-Attack	**ITN-Attack**	**GAN-Attack**
Tanaka's Scheme [16]		⋆	⋆	CIFAR-10 CIFAR-100	Pyramidal Residual Network	86.3% 56.8%	88.4% 59.1%	×	×	×
Pixel-based	**Same key [19]**	⋆	⋆	CIFAR-10	ResNet-18	91.76%	95.53%	×	×	×
	Different key [20]					91.39%		×	✓	×
TN-GAN [21]		⋆	⋆	CIFAR-10 CIFAR-100	ResNet-18	90.73% 67.36%	95.65% 77.24%	✓	×	✓
TN-model [22,23]		○	⋆	CIFAR-10 CIFAR-100	ResNet-20	91.72% 70.78%	91.23% 67.9%	✓	✓	✓

⋆: available, ○: not available, ✓: robust, ×: non-robust.

In 2020, a block-wise image scrambling method was proposed to increase the security level of the visually-protected images [24]. The method was referred to as Extended Learnable Encryption (ELE). In this method, after dividing the image into $M \times M$ blocks, the positions of the blocks are shuffled, and then the pixels in each block are shuffled. Finally, the blocks are concatenated and the scrambled image is obtained. Increasing the security level of this method causes a very low classification accuracy. For the CIFAR-10 dataset, it achieved 48.39% when the LE adaptation network was used and 83.06% when the ELE adaptation network replaces the LE one. This method has shown an important trade-off between security and classification accuracy. However, not much testing has been done on this method and its robustness against attacks, so it will not be compared with other methods later.

3.2. Pixel-Based Image Encryption

In 2019, a pixel-based image encryption method that considers data augmentation in the encrypted domain was introduced [19,20]. In this method, the data augmentation can be performed by the user before encryption or by the server after encryption. As shown in Figure 3, to generate an encrypted image I_e from a color image I with n pixels, the image I must first be divided into pixels. Then, a Negative-Positive (NP) transformation must be applied individually to each pixel of the three color channels; red I_R, green I_G, and blue I_B, to obtain a transformed pixel p' using a random binary integer $r(i)$ generated by a set of secret keys $K_{NP} = \{K_R, K_G, K_B\}$, where K_R, K_G, and K_B are respectively the keys used for

I_R, I_G and I_B, i is the ith pixel of I, and p is the pixel value of the original image with L bits per pixel:

$$p' = \begin{cases} p & \text{if } r(i) = 0 \\ p \oplus (2^L - 1) & \text{if } r(i) = 1 \end{cases} \tag{7}$$

Finally, the three color components of each pixel are shuffled optionally by using an integer that is randomly selected from six integers generated by a secret key K_s. Thus, the secret encryption key becomes $K = \{K_{NP}, K_s\}$. Two encryption key conditions for generating encrypted training and test images exist:

- Same encryption key: all training and test images are encrypted using the same encryption key K.
- Different encryption keys: different keys are independently assigned to training and test images.

Thus, the key space N_{pix} of the pixel-based encryption method is:

$$N_{pix} = 2^{3n} \, . \, 6^n. \tag{8}$$

Moreover, an adaptation network of 1 × 1 convolutional layers was proposed to make the encrypted images compatible with the DNN. The method was performed using the CIFAR-10 dataset and the ResNet-18 classifier network. Data augmentation techniques such as horizontal or vertical flip and shifting were carried out. The accuracy levels obtained were 92.03% using the same key and 91.23% using different keys when the data augmentation was carried out after encryption, while the accuracy levels obtained were 91.76% using the same key and 91.39% using different keys when the data augmentation was in the plain domain, as compared to 95.53% using plain images [20].

On the other hand, it was proved that Feature-Reconstruction attack (FR-attack) and GAN-attack can reconstruct the visual information from encrypted images using the pixel-based encryption method. The FR-attack reconstructs the edge information of plain images from encrypted ones. However, the pixel-based image encryption method is robust against ITN-attack when images are encrypted under different encryption keys [17,25]. The details and the results of the pixel-based method are shown in Table 2; only the results of applying data augmentation before encryption are displayed in the table.

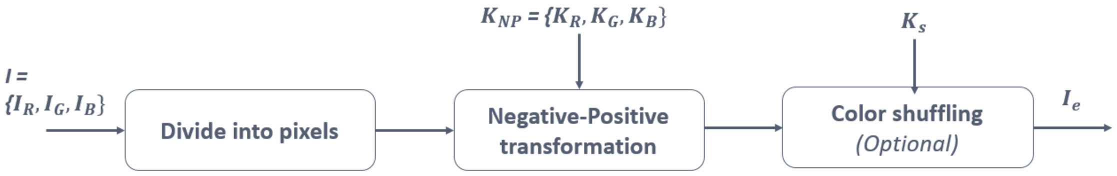


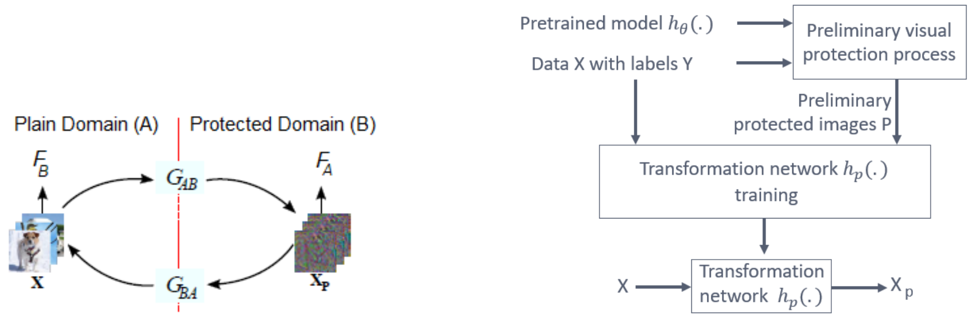
Figure 3. Pixel-based image encryption. © 2019 IEEE. Reprinted, with permission, from [19].

3.3. GAN-Based Image Transformation Scheme

In 2020, an image transformation network for privacy-preserving DNN using Generative Adversarial Networks (TN-GAN) was proposed [21].

A transformation network $h_p(.)$ was used to protect training and test images. To obtain this network, an unpaired image-to-image translation using a cycle-consistent adversarial network (cycle-GAN) was trained and used. The cycle-GAN consists of two GANs: two generative networks G_{AB} and G_{BA} and two discriminating networks F_A and F_B (see Figure 4a). In this application, the cycle-GAN converts images between two domains: the plain domain A and the visually-protected domain B. The generative network G_{AB} of the cycle-GAN was used as the transformation network $h_p(.)$. This network was trained using training images X with their corresponding labels Y and a set of preliminary protected images P, as shown in Figure 4b. The output of this network is visually-protected images

$X_p = h_p(X)$. P is generated from a pre-trained model $h_\theta(.)$ using X and Y in a preliminary visual protection process, and $h_\theta(.)$ is intended to accurately classify images without visual information.


Plain Domain (A)
Protected Domain (B)
F_B
F_A
G_{AB}
G_{BA}
X
X_p
Pretrained model $h_\theta(.)$
Data X with labels Y
Preliminary visual protection process
Preliminary protected images P
Transformation network $h_p(.)$ training
X
Transformation network $h_p(.)$
X_p

(a) **(b)**

Figure 4. (**a**) Cycle-GAN architecture. (**b**) Training process of $h_p(.)$ in the TN-GAN-based method. © 2021 IEEE. Reprinted, with permission, from [21].

In the experiments, the VGG-13 was used as the pre-trained network $h_\theta(.)$, the U-Net was used as the transformation network $h_p(.)$, and the ResNet-18 was used as the classifier DNN. $h_\theta(.)$ was trained using the CIFAR-10 dataset, and $h_p(.)$ was trained using the CIFAR-10 dataset with the preliminary protected images generated by $h_\theta(.)$.

To evaluate the performance, the classifier DNN was trained and tested using visually-protected images generated from the CIFAR-10 and the CIFAR-100 datasets. The obtained accuracy for CIFAR-10 was 90.73% and it was 67.36% for CIFAR-100, as compared to 95.05% for CIFAR-10 and 77.24% for CIFAR-100 using plain images. The application of the classifier DNN on the CIFAR-10 and CIFAR-100 datasets, with the transformation network being trained on CIFAR-10, proves that this scheme is applicable to all datasets, regardless of the dataset used to train the model. This method proved robustness against FR-attack and GAN-attack, but not against ITN-attack, where the visual information was reconstructed [17]. The details and the results of the TN-GAN method are shown in Table 2.

3.4. Model-Based Image Transformation Scheme

Still in 2020, an image transformation network trained with a model (TN-model) was proposed [22,23]. The classification model ψ, usually in the cloud provider, is trained using plain images X with their corresponding labels Y. $X' = \{x_1, x_2, \ldots, x_m\}$, a subset of training images ($X' \subseteq X$), is sent to the user to train the transformation network $h_\theta(.)$, which can be open to the public (see Figure 5). The output $\hat{X}' = \{\hat{x}_1, \hat{x}_{,2} \ldots, \hat{x}_m\}$ of $h_\theta(.)$ is a visually-protected image set, where $\hat{x}_i = h_\theta(x_i)$. $Y' = \{y_1, y_2, \ldots, y_m\} \subseteq Y$ is the corresponding target label set of X', and $\hat{Y}' = \{\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_m\}$ is the output of the classification model ψ, where $\hat{y}_i = \psi(\hat{x}_i)$. The transformation network $h_\theta(.)$ is trained in coordination with the classification model ψ in order to reduce its classification loss in a way as to ensure that the visually-protected images generated by $h_\theta(.)$ are correctly classified. The loss function of the transformation network L_{trans} must be minimized.

$$L_{trans}(x_i, \hat{x}_i, y_i) = L_{class}(\hat{x}_i, y_i) - \alpha \; . \; L_{feat}(x_i, \hat{x}_i), \tag{9}$$

where i is an integer number between 1 and m, m is the number of images in X', the classification loss L_{class} is a cross entropy loss function calculated using $\hat{y}_i = \psi(\hat{x}_i)$ and y_i, $\alpha \in \mathbb{R}$ is a weight of L_{feat}, and L_{feat} is a feature reconstruction loss calculated using the feature map of the original plain image $\phi_k(x_i)$ and the feature map of the reconstructed visually-protected image $\phi_k(\hat{x}_i)$:

$$L_{feat}(x_i, \hat{x}_i) = \frac{1}{C_k \times H_k \times W_k} ||\phi_k(\hat{x}_i) - \phi_k(x_i)||_2^2. \tag{10}$$

$C_k \times H_k \times W_k$ is the size of the feature map $\phi_k(.)$.

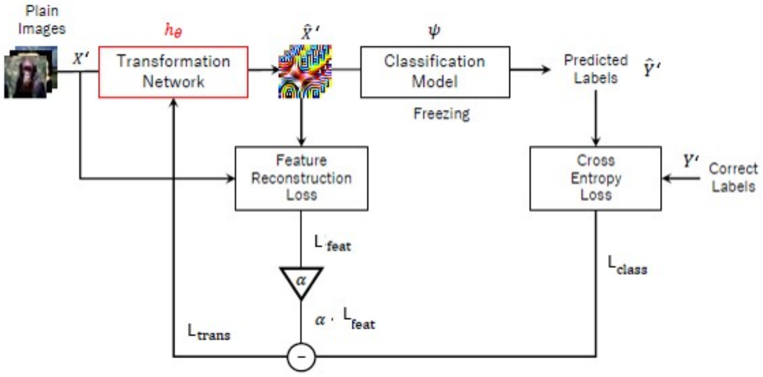


Figure 5. Training process of the transformation network in the TN-model-based method. © 2020 IEEE. Reprinted, with permission, from [23].

The value of α affects the classification performance since it affects the performance of the visual protection. In the case of $\alpha = 0$, the generated images are not visually-protected, whereas if $\alpha = 0.005$ the generated images have almost no visual information and have almost the same patterns. After training the transformation network, the visually-protected images generated are classified using ψ.

In the experiments, CIFAR-10 and CIFAR-100 datasets were used, U-Net was used as h_θ, and ResNet-20 was used as ψ. The accuracy obtained was 91.72% for CIFAR-10 and 70.78% for CIFAR-100, compared to 91.23% for CIFAR-10 and 67.9% for CIFAR-100 using plain images [23]. The accuracy obtained was 91.94% for CIFAR-10 when VGG-16 was used as ψ, compared to 92.23% using plain images [22]. All these results were obtained when α was equal to 0.005. The TN-model proved robust against an FR-attack, GAN-attack, and ITN-attack [17]. The details and the results of the TN-model method are shown in Table 2; only the results of applying ResNet-20 as a classifier network are displayed in the table.

3.5. Comparison

Perceptual image encryption methods are learnable image encryption methods; they transform or encrypt an image in a way that makes it incomprehensible to humans, but remains learned by machines. The resulting images are visually-protected images that do not contain any clear information. These methods allow the use of any type of network without any limitation of architecture or activation functions, unlike the computational methods.

Tanaka's scheme allows both training and testing of the DNN on encrypted images. It uses an adaptation network to reduce the influence of image encryption, whereas the analysis of this network can lead to understanding the scrambling or the encryption process [24]. Thus, it is possible to reconstruct protected images and obtain hidden information. The use of an adaptation network is therefore not a perfect solution, despite the advantages it provides. Furthermore, Tanaka's scheme is weak against Ciphertext-Only Attacks (COA) as proven in [17] and as shown in Table 2, so the visually protected images can be easily reconstructed by adversaries. Moreover, the same encryption key is used to encrypt all images, and very low accuracy was reached when testing DNN models that are trained using the images encrypted with different encryption keys [20].

On the other hand, the pixel-based image encryption method has a larger key space than Tanaka's scheme if the image is larger than 11 × 11 pixels, and it authorizes the use of different encryption keys for each image while maintaining the accuracy of the image classification. The use of different encryption keys makes this method robust against ITN-attack, while the visual information can be reconstructed by other COAs [17]. The pixel-based image encryption method allows as well data augmentation in the encrypted domain. The accuracies obtained by applying data augmentation before or after encryption were very close.

The pixel-based encryption method, like Tanaka's scheme, allows both training and testing of the DNN on encrypted images. Both methods, i.e., Tanaka's scheme and the pixel-based encryption method have low computational cost and apply simple transformation functions to the initial images. Therefore, adversaries can derive the encryption process using COA methods, especially when the same encryption key is used to encrypt all images (see Table 2).

Unlike Tanaka's scheme and the pixel-based encryption method, the other two perceptual methods (TN-GAN and TN-model) do not use simple transformation functions to generate visually-protected images; instead, they use transformation networks. These networks must be trained, which increases the computational cost. The computational cost of the TN-GAN is higher than that of the TN-model because TN-GAN needs to train a cycle-GAN which consists of two generative and two discriminating networks to get its transformation network, while the TN-model trains only one network.

As shown in the Table 2, the classification accuracy of the TN-GAN is affected by the dataset used; the accuracy decreased by about 10% when using CIFAR-10 as compared to the original accuracy (classification accuracy of the unencrypted images). However, the classification accuracy of the TN-model was able to outperform the original accuracy in certain cases. The superiority of the TN-model accuracy over the original accuracy is due to the training process of the transformation network; it is trained in coordination with the classifier to classify correctly protected images and to decrease its classification loss. Thus, the total number of parameters is increased, and better results are obtained. However, the TN-model does not allow the DNN to be trained on encrypted data, whereas the TN-GAN does. TN-model and TN-GAN have proven their robustness against the FR-attack and GAN-attack. Although the TN-GAN method was not robust against the ITN-attack, unlike the TN-model, it is impossible for adversaries to apply this attack on TN-GAN; ITN-attack requires the exact pairs of plain images and the corresponding encrypted ones, while the transformation network in the TN-GAN must remain private to the user [17].

4. Conclusions

In this article, we have reviewed and compared the most well-known privacy-preserving DNN methods that have allowed DNN to process and classify data in the encrypted domain. These methods have been classified into two types: computational methods and perceptual methods.

While computational methods preserve the security and privacy of data, they may limit the structure of NN and its activation functions, as is the case with HE-based methods. In addition, their computational cost is very high, and this cost increases with the depth of the NN. Therefore, their training time and the predictions take a long time, and the number of predictions per hour is relatively small. Thus, computational methods do not support the state-of-the-art DNN, but remain the most secure options for privacy-preserving computation.

On the other hand, perceptual methods allow the use of different types of DNN, without any limitation on the number of hidden layers, the structure of the DNN, or the activation functions. Their computational cost is very low when compared with computational methods, and they have sought a compromise in security to support other demands such as data augmentation in the encrypted domain. The perceptual methods aim to create visually-protected images and to make the DNN process these encrypted images exactly as it does with plain images. The visually-protected images are directly applied to the DNN. As a matter of fact, the process of generating these images differs from one method to another; some methods use simple transformation functions, while others use transformation NN. The complexity of the transformation process affects the computational cost as well as the ability to protect data and maintain confidentiality against attacks. This was revealed through the robustness of the GAN-model and TN-model against COAs that use transformation NN, and their relatively high complexity. Perceptual methods allow encrypted data to be used and processed more flexibly and smoothly, expanding the boundaries of AI applications in fields that use sensitive, secret, or personal data.

In both types, the computational and the perceptual ones, some methods allow the training of DNN using encrypted data, while others do not. In order to obtain durable privacy-preserving, the methods that support both training and testing over encrypted data are the most preferable. All these methods have focused on the classification of encrypted data and have managed to achieve acceptable performance.

The results obtained allow a wider reflection and development of the privacy-preserving DNN methods. These methods should be tested on more challenging and realistic datasets and critical problem-solving. Privacy-preserving DNN methods, especially the perceptual ones, can be developed and used to solve problems that are more complicated than data classification, such as object detection in the encrypted domain. Since the visually-protected images have carried and preserved enough information and features to be properly classified in the encrypted domain using previously trained networks, these specific networks can be trained and developed to detect objects in these protected images, as well as to apply segmentation or even transformation of these encrypted objects. In this way, it is possible to try to develop state-of-the-art AI algorithms on the visually-protected images that are generated by perceptual image encryption methods, especially TN-GAN and TN-model which have proven their ability to generate images that are robust against different COAs, without visual information, and with good classification accuracy. However, the TN-GAN has an advantage over the TN-model due to its ability to be trained using visually-protected images.

Author Contributions: Conceptualization, R.E.S., E.S.G. and A.A.; writing—original draft preparation, R.E.S. and E.S.G.; writing—review and editing, R.E.S., E.S.G., A.A. and M.K.; supervision, A.A. and M.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Hussain, H. A Review of Artificial Intelligence Techniques in Image Steganography Domain. *J. Eng. Sci. Technol.* **2017**, *12*, 1835–1845.
2. Liu, J.; Ke, Y.; Zhang, Z.; Lei, Y.; Li, J.; Zhang, M.; Yang, X. Recent Advances of Image Steganography with Generative Adversarial Networks. *IEEE Access* **2020**, *8*, 60575–60597. [CrossRef]
3. Fu, Z.; Wang, F.; Cheng, X. The secure steganography for hiding images via GAN. *EURASIP J. Image Video Process.* **2020**, *2020*. [CrossRef]
4. So, J. Deep Learning-Based Cryptanalysis of Lightweight Block Ciphers. *Secur. Commun. Netw.* **2020**, *2020*, 1–11. [CrossRef]
5. Xiao, Y.; Hao, Q.; Yao, D.D. Neural Cryptanalysis: Metrics, Methodology, and Applications in CPS Ciphers. In Proceedings of the 2019 IEEE Conference on Dependable and Secure Computing (DSC), Hangzhou, China, 18–20 November 2019; pp. 1–8. [CrossRef]
6. Volna, E.; Kotyrba, M.; Kocian, V.; Janosek, M. Cryptography Based on Neural Network. *ECMS* **2012**, 386–391. [CrossRef]
7. Acar, A.; Aksu, H.; Uluagac, A.S.; Conti, M. A Survey on Homomorphic Encryption Schemes: Theory and Implementation. *arXiv* **2017**, arXiv:1704.03578.
8. Boneh, D.; Sahai, A.; Waters, B. Functional Encryption: Definitions and Challenges. In *Theory of Cryptography*; Ishai, Y., Ed.; Springer: Berlin/Heidelberg, Germany, 2011; pp. 253–273.
9. Tanuwidjaja, H.C.; Choi, R.; Baek, S.; Kim, K. Privacy-Preserving Deep Learning on Machine Learning as a Service—A Comprehensive Survey. *IEEE Access* **2020**, *8*, 167425–167447. [CrossRef]
10. Bost, R.; Popa, R.; Tu, S.; Goldwasser, S. Machine Learning Classification over Encrypted Data. *NDSS* **2015**, *4325*. [CrossRef]
11. Dowlin, N.; Gilad-Bachrach, R.; Laine, K.; Lauter, K.; Naehrig, M.; Wernsing, J. CryptoNets: Applying Neural Networks to Encrypted Data with High Throughput and Accuracy. In Proceedings of the 33rd International Conference on Machine Learning, New York, NY, USA, 19–24 June 2016; pp. 201–210.
12. Hesamifard, E.; Takabi, H.; Ghasemi, M. CryptoDL: Deep Neural Networks over Encrypted Data. *arXiv* **2017**, arXiv:1711.05189.
13. Xu, R.; Joshi, J.B.D.; Li, C. CryptoNN: Training Neural Networks over Encrypted Data. *arXiv* **2019**, arXiv:1904.07303.
14. Maekawa, T.; Kawamura, A.; Kinoshita, Y.; Kiya, H. Privacy-Preserving SVM Computing in the Encrypted Domain. In Proceedings of the 2018 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC), Honolulu, HI, USA, 12–15 November 2018; pp. 897–902.

15. Kawamura, A.; Kinoshita, Y.; Nakachi, T.; Shiota, S.; Kiya, H. A Privacy-Preserving Machine Learning Scheme Using EtC Images. *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.* **2020**, *E103.A*, 1571–1578. [CrossRef]
16. Tanaka, M. Learnable Image Encryption. In Proceedings of the 2018 IEEE International Conference on Consumer Electronics-Taiwan (ICCE-TW), Taichung, Taiwan, 19–21 May 2018; pp. 1–2. [CrossRef]
17. Sirichotedumrong, W.; Kiya, H. Visual Security Evaluation of Learnable Image Encryption Methods against Ciphertext-only Attacks. In Proceedings of the 2020 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC), Auckland, New Zealand, 7–10 December 2020; pp. 1304–1309.
18. AprilPyone, M.; Sirichotedumrong, W.; Kiya, H. Adversarial Test on Learnable Image Encryption. In Proceedings of the 2019 IEEE 8th Global Conference on Consumer Electronics (GCCE), Osaka, Japan, 15–18 October 2019; pp. 667–669. [CrossRef]
19. Sirichotedumrong, W.; Maekawa, T.; Kinoshita, Y.; Kiya, H. Privacy-Preserving Deep Neural Networks with Pixel-Based Image Encryption Considering Data Augmentation in the Encrypted Domain. In Proceedings of the 2019 IEEE International Conference on Image Processing (ICIP), Taipei, Taiwan, 22–25 September 2019; pp. 674–678. [CrossRef]
20. Sirichotedumrong, W.; Kinoshita, Y.; Kiya, H. Pixel-Based Image Encryption Without Key Management for Privacy-Preserving Deep Neural Networks. *IEEE Access* **2019**, *7*, 177844–177855. [CrossRef]
21. Sirichotedumrong, W.; Kiya, H. A GAN-Based Image Transformation Scheme for Privacy-Preserving Deep Neural Networks. In Proceedings of the 2020 28th European Signal Processing Conference (EUSIPCO), Amsterdam, The Netherlands, 18–21 January 2021; pp. 745–749. [CrossRef]
22. Ito, H.; Kinoshita, Y.; Kiya, H. A Framework for Transformation Network Training in Coordination with Semi-trusted Cloud Provider for Privacy-Preserving Deep Neural Networks. In Proceedings of the 2020 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC), Auckland, New Zealand, 7–10 December 2020; pp. 1420–1424.
23. Ito, H.; Kinoshita, Y.; Kiya, H. Image Transformation Network for Privacy-Preserving Deep Neural Networks and Its Security Evaluation. *arXiv* **2020**, arXiv:2008.03143.
24. Madono, K.; Tanaka, M.; Onishi, M.; Ogawa, T. Block-wise Scrambled Image Recognition Using Adaptation Network. *arXiv* **2020**, arXiv:2001.07761.
25. Sirichotedumrong, W.; Kinoshita, Y.; Kiya, H. On the Security of Pixel-Based Image Encryption for Privacy-Preserving Deep Neural Networks. In Proceedings of the 2019 IEEE 8th Global Conference on Consumer Electronics (GCCE), Osaka, Japan, 15–18 October 2019; pp. 121–124. [CrossRef]