INVITED PAPER *Special Section on Enriched Multimedia–Advanced Safety, Security and Convenience–*

Image and Model Transformation with Secret Key for Vision Transformer

Hitoshi KIYA[†a)], ***Fellow***, **Ryota IIJIMA**[†b)], **Aprilpyone MAUNGMAUNG**[†c)], ***Nonmembers***, ***and*** **Yuma KINOSHITA**[†d)], ***Member***

SUMMARY In this paper, we propose a combined use of transformed images and vision transformer (ViT) models transformed with a secret key. We show for the first time that models trained with plain images can be directly transformed to models trained with encrypted images on the basis of the ViT architecture, and the performance of the transformed models is the same as models trained with plain images when using test images encrypted with the key. In addition, the proposed scheme does not require any specially prepared data for training models or network modification, so it also allows us to easily update the secret key. In an experiment, the effectiveness of the proposed scheme is evaluated in terms of performance degradation and model protection performance in an image classification task on the CIFAR-10 dataset.

key words: *perceptual image encryption, vision transformer, DNN, privacy preserving*

1. Introduction

Machine learning (ML) algorithms have been deployed in many applications including security-critical ones such as biometric authentication, automated driving, and medical image analysis [1], [2]. However, training successful models requires three ingredients: a huge amount of data, GPU accelerated computing resources, and efficient algorithms, and it is not a trivial task. In fact, collecting images and labeling them is also costly and will also consume a massive amount of resources. Therefore, trained ML models have great business value. Considering the expenses necessary for the expertise, money, and time taken to train a model, a model should be regarded as a kind of intellectual property (IP). In addition, generally, data contains sensitive information and it is difficult to train a model while preserving privacy. In particular, data with sensitive information cannot be transferred to untrusted third-party cloud environments (cloud GPUs and TPUs) even though they provide a powerful computing environment [3]–[9]. Accordingly, it has been challenging to train/test a ML model with encrypted images as one way for solving these issues [10].

Learnable image encryption, which has given new solutions to the above issues, is encryption that allows us not only to generate visually protected images to protect personally identifiable information included in an image, such as an individual or the time and location of the taken photograph, but to also apply encrypted images to a ML algorithm in the encrypted domain [10]. In addition, image encryption with a secret key, referred to as image transformation with a secret key, can embed unique features controlled with the key into images. The use of the unique features was demonstrated to be effective in applications such as adversarial defense and model protection [11]–[16]. However, even though many image transformation methods with a secret key have been studied so far for application to such applications, no conventional methods can avoid the influence of image transformation. In other words, the use of transformed images degrades the performance of models compared with models trained with plain images [11]–[16]. In addition, if we want to update the key, models have to be re-trained by using a new key.

In this paper, we show that the use of the vision transformer (ViT) [17] allows us to reduce the influence of block-wise encryption thanks to its architecture. After that, to overcome the problems with conventional image transformation, we propose a novel framework for ML algorithms with encrypted images that uses ViT.

In the framework, a model trained with plain images is also transformed with a secret key using the unique features in addition to test images, and the combined use of the transformed model and test images is proposed for using ML algorithms in the encrypted domain. The proposed scheme allows us not only to obtain the same performance as models trained with plain images but to also update the secret key easily. In an experiment, the effectiveness of the proposed scheme is evaluated in terms of performance degradation and model protection performance in an image classification task on the CIFAR-10 dataset.

2. Related Work

Conventional image transformation for machine learning and ViT are summarized here.

2.1 Image Transformation for Machine Learning

Various image transformation methods with a secret key, often referred to as perceptual image encryption or image cryptography, have been studied so far for many applica-

Manuscript received March 16, 2022.
Manuscript revised July 19, 2022.
Manuscript publicized November 2, 2022.

[†]The authors are with Tokyo Metropolitan University, Hino-shi, 191–0065 Japan.
a) E-mail: kiya@tmu.ac.jp
b) E-mail: iijima-ryota@ed.tmu.ac.jp
c) E-mail: fugokidi@gmail.com
d) E-mail: ykinoshita@tmu.ac.jp

DOI: 10.1587/transinf.2022MUI0001

Copyright © 2023 The Institute of Electronics, Information and Communication Engineers

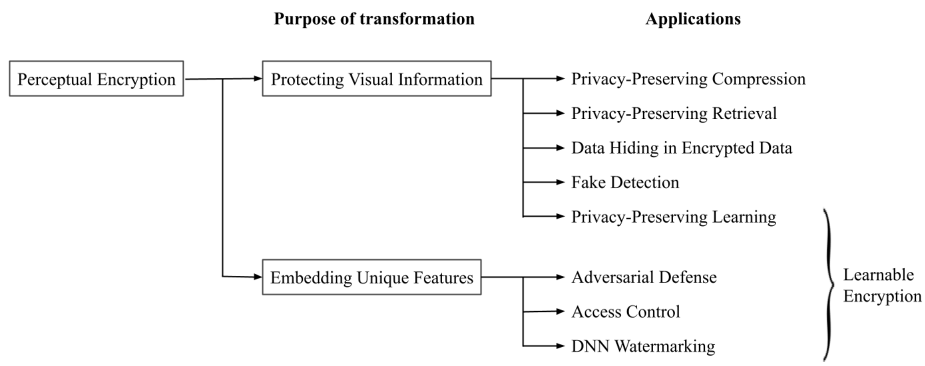


Fig. 1 Applications of perceptual image encryption

tions. Perceptional encryption can offer encrypted images that are described as bitmap images, so the encrypted images can be directly applied to image processing algorithms. In addition, encrypted images can be decrypted even when noise is added to them, although the use of standard encryption algorithms such as DES and AES cannot.

Figure 1 shows typical applications of image transformation with a key. Image transformation with a key allows us not only to protect visual information on plain images but to also embed unique features controlled with the key into images. The use of visually protected images has enabled various kinds of applications. One of the origins of image transformation with a key is in block-wise image encryption schemes for encryption-then-compression (EtC) systems [18]–[27]. Image encryption prior to image compression is required in certain practical scenarios such as secure image transmission through an untrusted channel provider. An EtC system is used in such scenarios, although the traditional way of securely transmitting images is to use a compression-then-encryption (CtE) system. Compressible encryption methods have been applied to privacy-preserving compression, data hiding, and image retrieval [28]–[30] in cloud environments. In addition, visually protected images have been demonstrated to be effective in fake image detection [31] and various learning algorithms [10], [32]–[37].

In this paper, we focus on image transformation methods for machine learning including deep neural networks (DNNs), called learnable encryption. Learnable encryption enables us to directly apply encrypted data to a model as training and testing data. Encrypted images have no visual information on plain images in general, so privacy-preserving learning can be carried out by using visually protected images. In addition, the use of a secret key allows us to embed unique features controlled with the key into images. Adversarial defense [11], [13], [14], access control [12], [38], [39], and DNN watermarking [14], [40]–[47] are carried out with encrypted data using the unique features.

2.2 Vision Transformer

The transformer architecture has been widely used in natural language processing (NLP) tasks [48]. The vision trans-

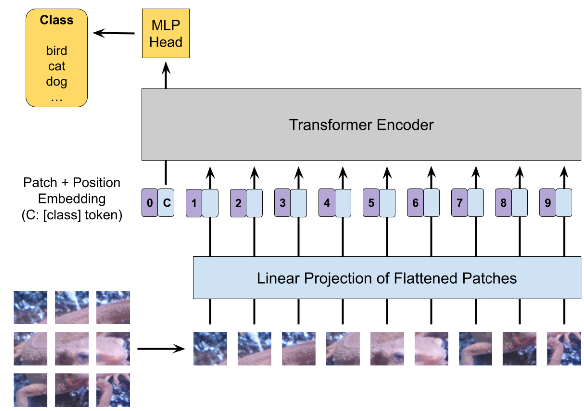


Fig. 2 Architecture of ViT [17]

former (ViT) [17] has also provided excellent results compared with state-of-the-art convolutional networks. Following the success of ViT, a number of isotropic networks (with the same depth and resolution across different layers in the network) have been proposed such as MLP-Mixer [49], ResMLP [50], CycleMLP [51], gMLP [52], vision permutator [53], and ConvMixer [54]. In this paper, we focus on ViT because it utilizes patch embedding and position embedding (see Fig. 2). Figure 2 illustrates the architecture of ViT. The main procedure of ViT is given as follows:

1. Split an image into fixed-size patches, and linearly embed each of them.
2. Add position embedding to patch embedding.
3. Feed the resulting sequence of vectors to a standard transformer encoder.
4. Feed the output of the transformer to a multi-layer perceptron (MLP), and get a result.

ViT utilizes patch embedding and position embedding. In this paper, we point out that the embedding structure enables us to reduce the influence of block-wise encryption. In patch embedding, patches are mapped to vectors, and in position embedding, the position information is embedded. If every patch (block) is transformed with the same key, pixel shuffling and bit flipping can be expressed as the operation of patch embedding. In addition, block permutation can be given as the operation of position embedding. The relation between block-wise encryption and the embedding structure is demonstrated to avoid the influence of block-wise image encryption.

3. Image and Model Transformation with Secret Key

An image transformation method with a secret key is proposed here. The method makes it possible to simultaneously use both transformed images and models.

3.1 Notation

The following notations are utilized throughout this paper.

- W, H, and C denote the width, height, and the number of channels of an image, respectively.
- The tensor $x \in [0, 1]^{C\times W\times H}$ represents an input color image.
- The tensor $x' \in [0, 1]^{C\times W\times H}$ represents an encrypted image.
- M is the block size of an image.
- Tensors x_b and ${x'}_\mathrm{b} \in [0, 1]^{W_\mathrm{b}\times H_\mathrm{b}\times p_\mathrm{b}}$ are a block image and an encrypted block image, respectively, where $W_\mathrm{b} = \frac{W}{M}$ is the number of blocks across width W, $H_\mathrm{b} = \frac{H}{M}$ is the number of blocks across height H, and $p_\mathrm{b} = M \times M \times C$ is the number of pixels in a block. We assume that W and H are divisible by M, so W_b and H_b are positive integers.
- A pixel value in a block image (x_b or ${x'}_\mathrm{b}$) is denoted by $x_\mathrm{b}(w, h, c)$ or ${x'}_\mathrm{b}(w, h, c)$, where $w \in \{0, ..., W_\mathrm{b} - 1\}$, $h \in \{0, ..., H_\mathrm{b} - 1\}$, and $c \in \{0, ..., p_\mathrm{b} - 1\}$ are indices corresponding to the dimension of x_b of ${x'}_\mathrm{b}$.
- $x_\mathrm{b}(w, h, :)$ denotes a vector $(x_\mathrm{b}(w, h, 0), ..., x_\mathrm{b}(w, h, p_\mathrm{b} - 1))$ of a tensor x_b.
- $x_\mathrm{b}(:, :, c)$ denotes a matrix of a tensor x_b as given in

$$\begin{pmatrix} x_\mathrm{b}(0,0,c) & \dots & x_\mathrm{b}(W_\mathrm{b}-1,0,c) \\ \vdots & \ddots & \vdots \\ x_\mathrm{b}(0,H_\mathrm{b}-1,c) & \dots & x_\mathrm{b}(W_\mathrm{b}-1,H_\mathrm{b}-1,c) \end{pmatrix}. \tag{1}$$

- B is a block in an image, and its dimension is $M\times M\times C$.
- $\hat{B}$ is a flattened version of block B, and its dimension is $1 \times 1 \times p_\mathrm{b}$.
- P is the patch size of an image.

3.2 Overview

Figure 3 shows the scenario of the proposed scheme, where it is assumed that the classification model builder is trusted, and the classification service provider is untrusted. The classification model builder trains a model by using plain images and transforms the trained model with a secret key where the transformation by using secret keys is performed only on the preprocessing part of the transformer encoder. The transformed model is given to the classification service provider, and the key is sent to a client. The client prepares a transformed test image with the key and sends it to the provider. The provider applies it to the transformed model to obtain a classification result, and the result is sent back to the client.

Note that the provider has neither a key nor plain images. The proposed scheme enables us to achieve this scenario without any performance degradation compared with the use of plain images.

3.3 Image Transformation

A block-wise image transformation with a secret key is proposed for application to test images. As shown in Fig. 4, the procedure of the transformation consists of three steps: block segmentation, block transformation, and block integration. To transform an image x, we first divide x into $W_\mathrm{b} \times H_\mathrm{b}$ blocks, as in $\{B_{11}, B_{12}, ..., B_{W_\mathrm{b}H_\mathrm{b}}\}$. In this paper, we assume that the block size of the segmentation is the same as the patch size of ViT. Next, each block is flattened, and it is concatenated again to obtain a block image x_b. Then, x_b is transformed to ${x'}_\mathrm{b}$ in accordance with block transformation with key K. Finally, ${x'}_\mathrm{b}$ is transformed so that it has the same $C \times H \times W$ dimensions as those of the original image, and encrypted image x' is obtained.

In addition, the block transformation is carried out by using the four operations shown in Fig. 4. Details on each operation are given below.

A Block Permutation

1. Generate a random permutation vector $\boldsymbol{v}_\mathbf{A} = (v_0, v_1, ..., v_k, ..., v_{k'}, ..., v_{W_\mathrm{b}\times H_\mathrm{b}-1})$ that consists of randomly permuted integers from 0 to $W_\mathrm{b} \times H_n - 1$ by using a key K_1, where $k, k' \in \{0, ..., H_\mathrm{b} \times W_\mathrm{b} - 1\}$, and $v_k \neq v_{k'}$ if $k \neq k'$.
2. Blocks are permutated to replace x_b with $x_\mathrm{b}^{\prime(1)}$ by using vector $\boldsymbol{v}_\mathbf{A}$ (see Algorithm 1).

Algorithm 1 Block permutation

Input: x_b, K_1
Output: $x_\mathrm{b}^{\prime(1)}$
Generate a vector $\boldsymbol{v}_\mathbf{A}$ using key K_1 .
$y_\mathrm{b} \leftarrow (x_\mathrm{b}[0, 0, :], x_\mathrm{b}[1, 0, :], ..., x_\mathrm{b}[W_\mathrm{b} - 1, H_\mathrm{b} - 1, :])$
$i \leftarrow 0$
while $i < H_\mathrm{b} \times W_\mathrm{b}$ **do**
  $y'_\mathrm{b}[i] \leftarrow y_\mathrm{b}[\boldsymbol{v}_\mathbf{A}[i]]$
  $i \leftarrow i + 1$
end while
$w \leftarrow 0$
$h \leftarrow 0$
while $h < H_\mathrm{b}$ **do**
  while $w < W_\mathrm{b}$ **do**
    $x_\mathrm{b}^{\prime(1)}[w, h, :] \leftarrow y'_\mathrm{b}[w + h \times W_\mathrm{b}]$
    $w \leftarrow w + 1$
  end while
  $h \leftarrow h + 1$
end while
return $x_\mathrm{b}^{\prime(1)}$

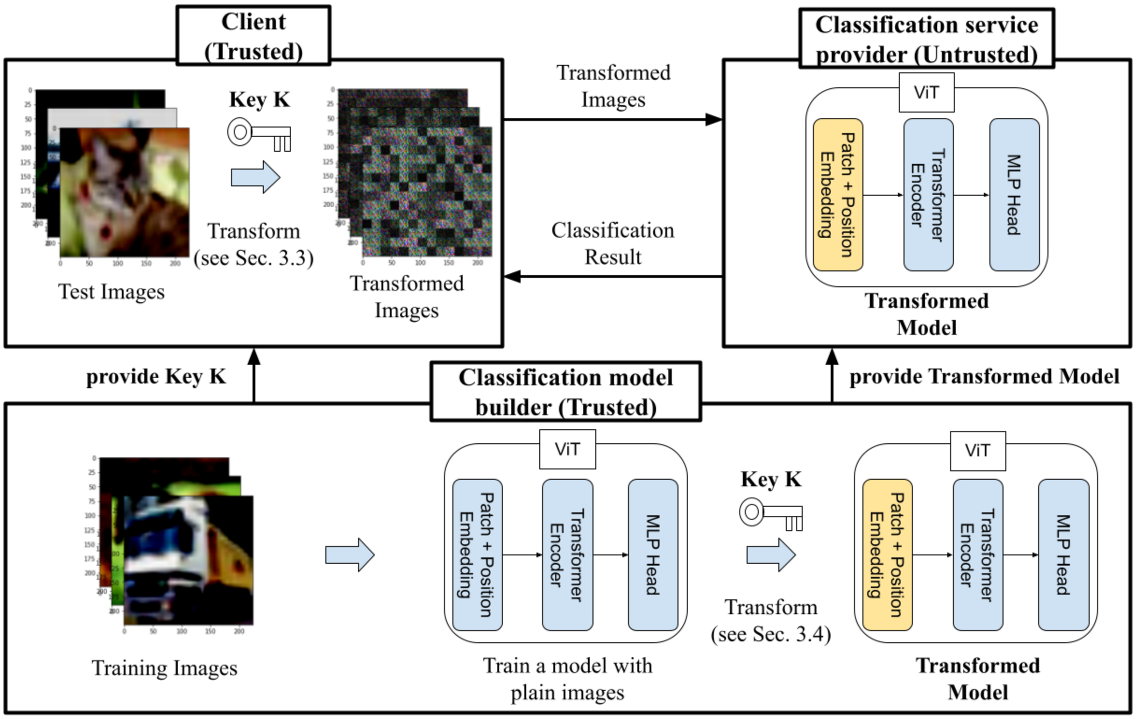


Fig. 3 Scenario of proposed scheme

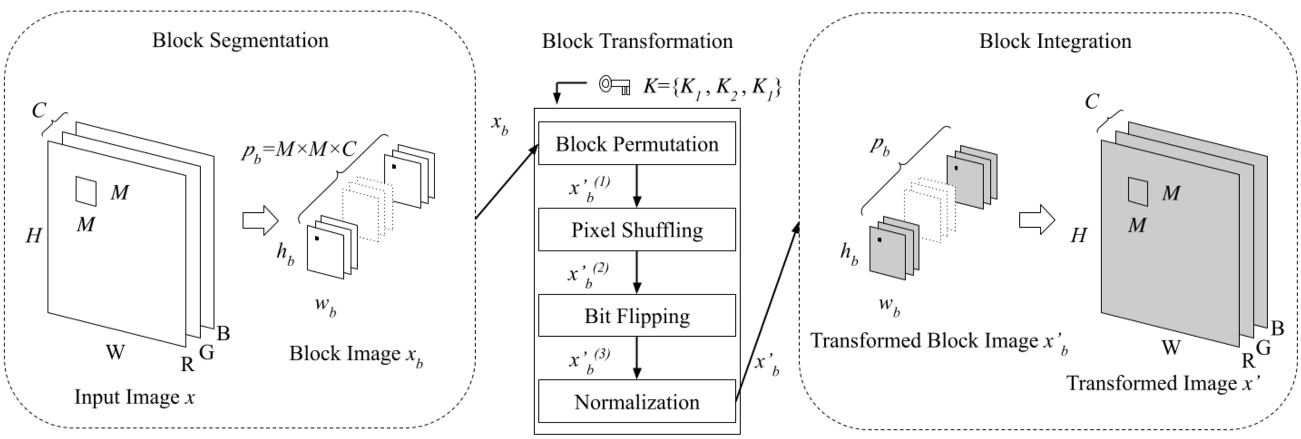


Fig. 4 Procedure of block-wise transformation

B Pixel Shuffling

1. Generate a random permutation vector $\boldsymbol{v}_\mathbf{B} = (v_0, v_1, ..., v_k, ..., v_{k'}, ..., v_{p_b-1})$ by using a key K_2, where $k, k' \in \{0, ..., p_b - 1\}$, and $v_k \neq v_{k'}$ if $k \neq k'$.
2. Pixels in each block are shuffled by vector $\boldsymbol{v}_\mathbf{B}$ as (see Algorithm 2).

$$x'^{(2)}_b(w, h, k) = x'^{(1)}_b(w, h, v_k). \qquad (2)$$

Algorithm 2 Pixel shuffling

Input: $x'^{(1)}_b, K_2$
Output: $x'^{(2)}_b$
Generate a vector $\boldsymbol{v}_\mathbf{B}$ using key K_2 .
$i \leftarrow 0$
while $i < p_b$ **do**
$x'^{(2)}_b[:, :, i] \leftarrow x'^{(1)}_b[:, :, \boldsymbol{v}_\mathbf{B}[i]]$
$i \leftarrow i + 1$
end while
return $x'^{(2)}_b$

C Bit Flipping

1. Generate a random binary vector $\boldsymbol{r} = (r_0, ..., r_k, ..., r_{p_b-1})$, $r_k \in \{0, 1\}$ by using a key K_3. To keep the transformation consistent, r is distributed with 50% of "0"s and 50% of "1"s.
2. Apply negative-positive transformation on the basis of $\boldsymbol{r}$ as

$$x'^{(3)}_{\mathrm{b}}(w,h,k) = \begin{cases} x'^{(2)}_{\mathrm{b}}(w,h,k) & (r_k = 0) \\ 1 - x'^{(2)}_{\mathrm{b}}(w,h,k) & (r_k = 1). \end{cases} \tag{3}$$

Algorithm 3 Bit flipping

Input: $x'^{(2)}_{\mathrm{b}}, K_3$
Output: $x'^{(3)}_{\mathrm{b}}$

```
Generate a vector r using key K3 .
i ← 0
while i < p_b do
    if r[i] = 0 then
        x'^(3)_b[:, :, i] ← x'^(2)_b[:, :, i]
    else
        x'^(3)_b[:, :, i] ← 1 − x'^(2)_b[:, :, i]
    end if
    i ← i + 1
end while
return x'^(3)_b
```

D Normalization

Various normalization methods are widely used to improve the training stability, optimization efficiency, and generalization ability of DNNs. In this paper, we use a normalization method to achieve the combined use of transformed images and models.

From Eq. (3), if $r_k = 0$, a pixel $x'^{(3)}_{\mathrm{b}}(w,h,k)$ is replaced with $x'_{\mathrm{b}}(w,h,k)$ as

$$\begin{aligned} x'_{\mathrm{b}}(w,h,k) &= \frac{x'^{(3)}_{\mathrm{b}}(w,h,k) - 1/2}{1/2} \\ &= \frac{x'^{(2)}_{\mathrm{b}}(w,h,k) - 1/2}{1/2}. \end{aligned} \tag{4}$$

In contrast, if $r_k = 1$, a pixel $x'^{(3)}_{\mathrm{b}}(w,h,k)$ is replaced as follows.

$$\begin{aligned} x'_{\mathrm{b}}(w,h,c) &= \frac{x'^{(3)}_{\mathrm{b}}(w,h,c) - 1/2}{1/2} \\ &= 2x'^{(3)}_{\mathrm{b}}(w,h,c) - 1 \\ &= 2(1 - x'^{(2)}_{\mathrm{b}}(w,h,c)) - 1 \\ &= 1 - 2x'^{(2)}_{\mathrm{b}}(w,h,c) \\ &= -\frac{x'^{(2)}_{\mathrm{b}}(w,h,c) - 1/2}{1/2} \end{aligned} \tag{5}$$

Therefore, bit flipping with normalization can be regarded as an operation that reverses the positive or negative sign of a pixel value. This property allows us to use the model encryption that will be described later.

The above encryption steps are the same as those of a number of conventional methods [10], [11], but the performance of conventional models is degraded due to the influence of encryption when the encryption steps are used in the conventional schemes. In contrast, the proposed method is demonstrated to avoid the influence of encryption in this paper, which is one of the reasons to apply the algorithm written in each step. In addition, the encryption steps can be expressed as a linear transform as described in the paper. Other encryption steps can also be used under the framework of the proposed scheme, if they are expressed as a linear transform such as a random matrix.

3.4 Model Transformation

In model transformation, some parameters in models trained with plain images are transformed by using a secret key. In this paper, a model transformation method is proposed that can achieve the combined use of models and images transformed with the same key.

ViT utilizes patch embedding and position embedding (see Fig. 2), so it has the following two properties.

1. Patch-order invariance of transformer encoder: the output of the transformer encoder corresponding to an input patch is independent of the order of input patches.
2. Ability to adapt to pixel order by patch embedding: patch embedding can be adapted to pixel shuffling and bit flipping because they can be expressed as an invertible linear transformation as described below.

In the proposed scheme, it is assumed that the patch size P used for patching is the same as the block size used for image encryption, and the number of patches is equal to that of blocks in an image. The transformation of parameters in trained models is described below.

A Position Embedding and Patch Embedding

In ViT [17], all segmented patches are flattened. A pixel value in the flattened patches is given by a pixel value in the original image as

$$\begin{aligned} &\boldsymbol{x}^i_{\mathrm{p}}[k] = x[h,w,c], \\ &h = \left\lfloor \frac{i-1}{W/P} \right\rfloor P + \left\lfloor \frac{k-1 \bmod P^2}{P} \right\rfloor, \\ &w = ((i-1) \bmod (W/P))\,P + ((k-1) \bmod P), \\ &c = \left\lfloor \frac{k-1}{P^2} \right\rfloor, \\ &\boldsymbol{x}^i_{\mathrm{p}} \in \mathbb{R}^{P^2C}, i \in \{1,2,...,N\}, k \in \{1,2,...,P^2C\} \end{aligned} \tag{6}$$

where $\boldsymbol{x}^i_{\mathrm{p}}[k]$ is a pixel value in the i-th patch, and N =

HW/P^2 is the number of patches. To simplify the discussion, we assume that H and W are divisible by P.

Then, in patch embedding, the flattened patches are mapped to vectors with dimensions of D by using a matrix $\mathbf{E} \in \mathbb{R}^{(P^2C)\times D}$, and in position embedding, the position information $\mathbf{E}_{\text{pos}} \in \mathbb{R}^{(N+1)\times D}$ is embedded into each patch as

$$
\begin{aligned}
z_0 &= [\boldsymbol{x}_{\text{class}}; \boldsymbol{x}_{\text{p}}^1\mathbf{E}; \boldsymbol{x}_{\text{p}}^2\mathbf{E}; \cdots; \boldsymbol{x}_{\text{p}}^N\mathbf{E}] + \mathbf{E}_{\text{pos}}, \\
\mathbf{E}_{\text{pos}} &= \left(\left(\boldsymbol{e}_{\text{pos}}^0\right)^\top \quad \left(\boldsymbol{e}_{\text{pos}}^1\right)^\top \quad \cdots \quad \left(\boldsymbol{e}_{\text{pos}}^N\right)^\top\right)^\top \\
\boldsymbol{e}_{\text{pos}}^i &\in \mathbb{R}^D, i = 0, 1, ..., N
\end{aligned}
\tag{7}
$$

where $\boldsymbol{x}_{\text{class}}$ is the classification token which is the input to MLP (see Fig. 2), $\boldsymbol{e}_{\text{pos}}^0$ is the information of the classification token.

The proposed model transformation is carried out in accordance with the above relation.

B Position Embedding Transformed with Key

Position embedding is an operation that embeds position information into classification token and each patch as in Eq. (7). Let us define a matrix $\hat{\mathbf{E}}_{\text{pos}}$ consisting of position information of each patch as

$$
\begin{aligned}
\hat{\mathbf{E}}_{\text{pos}} &= \left(\left(\boldsymbol{e}_{\text{pos}}^1\right)^\top \quad \left(\boldsymbol{e}_{\text{pos}}^2\right)^\top \quad \cdots \quad \left(\boldsymbol{e}_{\text{pos}}^N\right)^\top\right)^\top, \\
\boldsymbol{e}_{\text{pos}}^i &\in \mathbb{R}^D, i = 1, 2, ..., N.
\end{aligned}
\tag{8}
$$

Note that $\hat{\mathbf{E}}_{\text{pos}}$ does not contain information about the classification token.

The permutation of rows in $\hat{\mathbf{E}}_{\text{pos}}$ corresponds to the block permutation in image transformation. Therefore, $\hat{\mathbf{E}}_{\text{pos}}$ can be permuted using key K_1 used for block permutation, and the model can be encrypted. The transformed model offers a high accuracy for only test images transformed by block permutation with K_1. By defining a permutation matrix $\mathbf{E}_1$ with key K_1, the transformation from $\hat{\mathbf{E}}_{\text{pos}}$ to $\hat{\mathbf{E}}'_{\text{pos}}$ can be given as follows.

$$
\begin{aligned}
\hat{\mathbf{E}}'_{\text{pos}} &= \mathbf{E}_1\hat{\mathbf{E}}_{\text{pos}} \\
&= \left(\left(\boldsymbol{e}'^1_{\text{pos}}\right)^\top \quad \cdots \quad \left(\boldsymbol{e}'^N_{\text{pos}}\right)^\top\right)^\top, \\
\boldsymbol{e}'^i_{\text{pos}} &\in \{\boldsymbol{e}_{\text{pos}}^1, \cdots, \boldsymbol{e}_{\text{pos}}^N\}
\end{aligned}
\tag{9}
$$

Therefore, from Eq. (7), the transformed matrix $\mathbf{E}'_{\text{pos}}$ is given by

$$
\mathbf{E}'_{\text{pos}} = \left(\left(\boldsymbol{e}_{\text{pos}}^0\right)^\top \quad \left(\boldsymbol{e}'^1_{\text{pos}}\right)^\top \quad \cdots \quad \left(\boldsymbol{e}'^N_{\text{pos}}\right)^\top\right)^\top. \tag{10}
$$

C Patch Embedding Transformed with Key

In patch embedding, flattened patches are mapped to vectors with a dimension of D as in Eq. (7). When the patch size of ViT is equal to the block size of image transformation, $P^2C = p_{\text{b}}$ is satisfied. Therefore, the permutation of rows in $\mathbf{E}$ corresponds to pixel shuffling, so the model can be encrypted with key K_2 used for pixel shuffling. The accuracy of the transformed model is high only when test images are encrypted by using pixel shuffling with key K_2. As well as the transformation of $\mathbf{E}_{\text{pos}}$, a permutation matrix $\mathbf{E}_2$ is defined with key K_2, and the transformation from matrix $\mathbf{E}$ to $\mathbf{E}'$ is shown as follows.

$$
\mathbf{E}' = \mathbf{E}_2\mathbf{E} \tag{11}
$$

In addition, as shown in Eqs. (4) and (5), bit flipping with normalization can be regarded as an operation that randomly inverses the positive/negative sign of a pixel value. Therefore, we can encrypt a model by inverting the sign of the rows in matrix $\mathbf{E}$ with key K_3 used for bit flipping. The transformed model offers a high accuracy only for test images transformed by bit flipping with key K_3. Using key K_3 to generate the same vector $\boldsymbol{r}$ used in bit flipping, the transformation from $\mathbf{E}$ to $\mathbf{E}'$ can be expressed as follows.

$$
\mathbf{E}'(k,:) = \begin{cases} \mathbf{E}(k,:) & (r_k = 0) \\ -\mathbf{E}(k,:) & (r_k = 1) \end{cases} \tag{12}
$$

where $\mathbf{E}(k,:)$ and $\mathbf{E}'(k,:)$ are k-th rows of matrices $\mathbf{E}$ and $\mathbf{E}'$.

Accordingly, the procedure of the proposed method can be summarized as follows.

Step 1: Prepare a key set $K = \{K_1, K_2, K_3\}$.
Step 2: Generate $\mathbf{E}_1$ with K_1, $\mathbf{E}_2$ with K_2, and $\boldsymbol{r}$ with K_3
Step 3: Transform a model V_θ trained with plain images by using $\mathbf{E}_1$, $\mathbf{E}_2$, and $\boldsymbol{r}$ on the basis of Eqs. (9), (10), and (11) as

$$
V'_\theta = t(V_\theta, \{\mathbf{E}_1, \mathbf{E}_2, \boldsymbol{r}\}), \tag{13}
$$

where $t(V_\theta, \{\mathbf{E}_1, \mathbf{E}_2, \boldsymbol{r}\})$ is the proposed model transformation algorithm, and V'_θ is a transformed model.
Step 4: Transform test images with $K = \{K_1, K_2, K_3\}$.

3.5 Properties of Proposed Scheme

The proposed method has the following properties.

- The model performs well only if test images are transformed with the same key as that used for transforming the model.
- The proposed scheme does not cause performance degradation, due to the relation

$$
V'_\theta(x') = V_\theta(x). \tag{14}
$$

- Model training and encryption are independent. Therefore, it is possible to easily update a key.

4. Experiment and Discussion

In an experiment, the effectiveness of the proposed scheme was shown in terms of image classification accuracy and model protection performance.

4.1 Experiment Setup

To confirm the effectiveness of the proposed method, we evaluated the accuracy of an image classification task on the CIFAR-10 dataset (with 10 classes). The CIFAR-10 consists of 60,000 color images (dimension of $3 \times 32 \times 32$), where 50,000 images are for training, 10,000 for testing, and each class contains 6,000 images. Images in the dataset were resized to $3 \times 224 \times 224$ to input them to ViT, before applying the proposed encryption algorithm, where the block size was 16×16.

We used the PyTorch [55] implementation of ViT and fine-tuned a ViT model with a patch size $P = 16$, which was pre-trained on ImageNet-21k. The ViT model was fine-tuned for 5000 epochs. The parameters of the stochastic gradient descent (SGD) optimizer were a momentum of 0.9 and a learning rate value of 0.03.

In addition, we used three conventional visual information protection methods (Tanaka's method [56], the pixel-based encryption method [57], and the GAN-based transformation method [58]) to compare them with the proposed method. ResNet-20 was used to validate the effectiveness of the conventional method with reference to [59]. The CIFAR-10 was also used for training networks, and the networks were trained for 200 epochs by using stochastic gradient descent (SGD) with a weight decay of 0.0005 and a momentum of 0.9. The learning rate was initially set to 0.1, and it was multiplied by 0.2 at 60, 120, and 160 epochs. The batch size was 128.

4.2 Image Classification

First, we evaluated the proposed and conventional methods in terms of the accuracy of image classification under the use of ViT and ResNet-20. As shown in Table 1, the performance of all conventional methods was degraded compared with the baselines. In contrast, the proposed method did not degrade the performance at all.

4.3 Model Protection

Next, we validated whether the proposed method has the ability to protect models. Table 2 shows the accuracy of image classification when encrypted or plain images were input to the encrypted model.

The encrypted model performed well for test images with the correct key, but its accuracy was not high when using plain test images. The CIFAR-10 dataset consists of ten classes, so 9.06 is almost the same accuracy as that when test images are randomly classified.

Next, we confirmed the performance of images encrypted with a different key from that used in the model encryption. We prepared 100 random keys, and test images encrypted with the keys were input to the encrypted model. From the box plot in Fig. 5, the accuracy of the models was not high under the use of the wrong keys. Accordingly, the proposed scheme was confirmed to be robust against a random key attack.

The use of a large key space enhances robustness against various attacks in general. In this experiment, the key space of block permutation, pixel shuffling, and bit flipping (O_{P}, O_{S}, and O_{F}) is given by

$$\begin{aligned} O_{\mathrm{P}} &= (W_{\mathrm{b}} H_{\mathrm{b}})! \\ &= (WH/M^2)! = 196!, \end{aligned} \tag{15}$$

$$\begin{aligned} O_{\mathrm{S}} &= p_{\mathrm{b}}! \\ &= (M^2 C)! = 768!, \end{aligned} \tag{16}$$

and

$$\begin{aligned} O_{\mathrm{F}} &= \frac{p_{\mathrm{b}}!}{(p_{\mathrm{b}}/2)! \cdot (p_{\mathrm{b}}/2)!} \\ &= \frac{(M^2 C)!}{(M^2 C/2)! \cdot (M^2 C/2)!} = \frac{768!}{384! \cdot 384!}. \end{aligned} \tag{17}$$

Therefore, the key space of the proposed method is represented as follows.

Table 1 Comparison with conventional methods in terms of classification accuracy

Model	Method	Accuracy
ViT	Baseline	99.03
	Proposed	**99.03**
ResNet-20 [59]	Baseline	91.55
	Tanaka [56]	87.02
	Pixel-based [57]	86.66
	GAN-based [58]	82.55

Table 2 Robustness against use of plain images

Model	Test Image: Plain	Test Image: Proposed
Baseline	**99.03**	-
Proposed	9.06	**99.03**

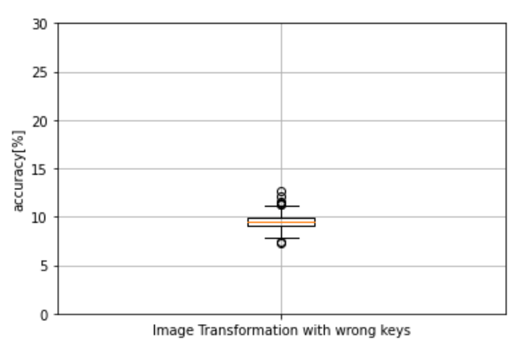


Fig. 5 Evaluating robustness against random key attack. Boxes span from first to third quartile, referred to as $Q1$ and $Q3$, and whiskers show maximum and minimum values in range of $[Q1 - 1.5(Q3 - Q1), Q3 + 1.5(Q3 - Q1)]$. Band inside box indicates median. Outliers are indicated as dots.

$$O = O_{\mathrm{P}} \times O_{\mathrm{S}} \times O_{\mathrm{F}} \simeq 2^{8237} \tag{18}$$

Typical cipher systems are recommended to have 2^{128} as a key space as in [60]. Accordingly, the key space O is sufficiently large, so it is difficult to find the correct key by random key estimation.

5. Conclusion

In this paper, we proposed the combined use of an image transformation method with a secret key and ViT models transformed with the key. The proposed scheme enables us not only to use visually protected images but to also maintain the same classification accuracy as that of models trained with plain images. In addition, in an experiment, the proposed scheme was demonstrated to outperform state-of-the-art methods with perceptually encrypted images in terms of classification accuracy, and it was also verified to be effective in model protection.

Acknowledgments

This study was partially supported by JSPS KAKENHI (Grant Number JP21H01327) and JST CREST (Grant Number JPMJCR20D3).

References

[1] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," nature, vol.521, no.7553, p.436, 2015.
[2] X. Liu, Z. Deng, and Y. Yang, "Recent progress in semantic image segmentation," Artif. Intell. Rev., vol.52, no.2, pp.1089–1106, 2019.
[3] C.-T. Huang, L. Huang, Z. Qin, H. Yuan, L. Zhou, V. Varadharajan, and C.-C.J. Kuo, "Survey on securing data storage in the cloud," APSIPA Trans. Signal and Information Processing, vol.3, p.e7, 2014.
[4] M.-R. Ra, R. Govindan, and A. Ortega, "P3: Toward privacy-preserving photo sharing," 10th USENIX Symposium on Networked Systems Design and Implementation (NSDI 13), pp.515–528, USENIX Association, Lombard, IL, April 2013.
[5] R. Lagendijk, Z. Erkin, and M. Barni, "Encrypted signal processing for privacy protection: Conveying the utility of homomorphic encryption and multiparty computation," IEEE Signal Process. Mag., vol.30, no.1, pp.82–105, Jan. 2013.
[6] M. Fredrikson, S. Jha, and T. Ristenpart, "Model inversion attacks that exploit confidence information and basic countermeasures," Proc. 22nd ACM SIGSAC Conference on Computer and Communications Security, ser. CCS'15, pp.1322–1333, Association for Computing Machinery, New York, NY, USA, 2015.
[7] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," 2017 IEEE symposium on security and privacy (SP), pp.3–18, IEEE, 2017.
[8] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I.J. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," 2nd Int. Conf. Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings, Y. Bengio and Y. LeCun, eds., 2014.
[9] R.S.S. Kumar, M. Nyström, J. Lambert, A. Marshall, M. Goertzel, A. Comissoneru, M. Swann, and S. Xia, "Adversarial machine learning-industry perspectives," 2020 IEEE Security and Privacy Workshops (SPW), pp.69–75, 2020.
[10] H. Kiya, A.P.M. Maung, Y. Kinoshita, S. Imaizumi, and S. Shiota, "An overview of compressible and learnable image transformation with secret key and its applications," APSIPA Trans. Signal and Information Processing, vol.11, no.1, e11, 2022. [Online]. Available: http://dx.doi.org/10.1561/116.00000048
[11] M. Aprilpyone and H. Kiya, "Block-wise image transformation with secret key for adversarially robust defense," IEEE Trans. Inf. Forensics Security, vol.16, pp.2709–2723, 2021.
[12] A. Maungmaung and H. Kiya, "A protection method of trained CNN model with a secret key from unauthorized access," APSIPA Trans. Signal and Information Processing, vol.10, p.e10, 2021.
[13] M. Maung, A. Pyone, and H. Kiya, "Encryption inspired adversarial defense for visual classification," 2020 IEEE Int. Conf. Image Process. (ICIP), 2020, pp.1681–1685.
[14] A. MaungMaung and H. Kiya, "Ensemble of key-based models: Defense against black-box adversarial attacks," 2021 IEEE 10th Global Conference on Consumer Electronics (GCCE), pp.95–98, 2021.
[15] M. AprilPyone and H. Kiya, "Privacy-preserving image classification using an isotropic network," IEEE MultiMedia, vol.29, no.2, pp.23–33, April -June 2022.
[16] Z. Qi, A. MaungMaung, Y. Kinoshita, and H. Kiya, "Privacy-preserving image classification using vision transformer," arXiv preprint arXiv:1804.00750, 2022. [Online]. Available: https://arxiv.org/abs/2205.12041
[17] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," Int. Conf. Learning Representations, 2021. [Online]. Available: https://openreview.net/forum?id=YicbFdNTTy
[18] T. Chuman, W. Sirichotedumrong, and H. Kiya, "Encryption-then-compression systems using grayscale-based image encryption for jpeg images," IEEE Trans. Inf. Forensics Security, vol.14, no.6, pp.1515–1525, June 2019.
[19] T. Chuman, K. Kurihara, and H. Kiya, "On the security of block scrambling-based etc systems against jigsaw puzzle solver attacks," 2017 IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP), pp.2157–2161, 2017.
[20] J. Zhou, X. Liu, O.C. Au, and Y.Y. Tang, "Designing an efficient image encryption-then-compression system via prediction error clustering and random permutation," IEEE Trans. Inf. Forensics Security, vol.9, no.1, pp.39–50, Jan. 2014.
[21] M. Ghonge and K. Nimbokar, "A survey based on designing an efficient image encryption-then-compression system," Int. J. Computer Applications, p.8887, 2014.
[22] T.Y. Liu, K.J. Lin, and H.C. Wu, "ECG data encryption then compression using singular value decomposition," IEEE J. Biomedical and Health Informatics, vol.22, no.3, pp.707–713, May 2018.
[23] W. Liu, W. Zeng, L. Dong, and Q. Yao, "Efficient compression of encrypted grayscale images," IEEE Trans. Image Process., vol.19, no.4, pp.1097–1102, April 2010.
[24] R. Hu, X. Li, and B. Yang, "A new lossy compression scheme for encrypted gray-scale images," 2014 IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP), pp.7387–7390, 2014.
[25] M. Johnson, P. Ishwar, V. Prabhakaran, D. Schonberg, and K. Ramchandran, "On compressing encrypted data," IEEE Trans. Signal Process., vol.52, no.10, pp.2992–3006, Oct. 2004.
[26] M. Gaata and F.F. Hantoosh, "An efficient image encryption technique using chaotic logistic map and rc4 stream cipher," Int. J. Modern Trends in Engineering and Research, vol.3, pp.213–218, 2016.
[27] O. Watanabe, A. Uchida, T. Fukuhara, and H. Kiya, "An encryption-then-compression system for jpeg 2000 standard," 2015 IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP), pp.1226–1230, 2015.
[28] S. Imaizumi, Y. Izawa, R. Hirasawa, and H. Kiya, "A reversible data hiding method in compressible encrypted images," IEICE Trans. Fundamentals, vol.E103-A, no.12, pp.1579–1588, Dec. 2020.
[29] K. Iida and H. Kiya, "Privacy-preserving content-based image retrieval using compressible encrypted images," IEEE Access, vol.8,

pp.200038–200050, 2020.

[30] K. Iida and H. Kiya, "An image identification scheme of encrypted jpeg images for privacy-preserving photo sharing services," 2019 IEEE Int. Conf. Image Process. (ICIP), pp.4564–4568, 2019.

[31] S. Liu, Z. Lian, S. Gu, and L. Xiao, "Block shuffling learning for deepfake detection," arXiv preprint arXiv:2202.02819, 2022. [Online]. Available: https://arxiv.org/abs/2202.02819

[32] A. Kawamura, Y. Kinoshita, T. Nakachi, S. Shiota, and H. Kiya, "A privacy-preserving machine learning scheme using etc images," IEICE Trans. Fundamentals, vol.E103-A, no.12, pp.1571–1578, Dec. 2020.

[33] Y. Bandoh, T. Nakachi, and H. Kiya, "Distributed secure sparse modeling based on random unitary transform," IEEE Access, vol.8, pp.211762–211772, 2020.

[34] T. Nakachi, Y. Bandoh, and H. Kiya, "Secure overcomplete dictionary learning for sparse representation," IEICE Trans. Inf. & Syst., vol.E103-D, no.1, pp.50–58, Jan. 2020.

[35] T. Nakachi, Y. Wang, and H. Kiya, "Privacy-preserving pattern recognition using encrypted sparse representations in l0 norm minimization," ICASSP 2020 - 2020 IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP), pp.2697–2701, 2020.

[36] I. Nakamura, Y. Tonomura, and H. Kiya, "Unitary transform-based template protection and its application to l^2-norm minimization problems," IEICE Trans. Inf. & Syst., vol.E99-D, no.1, pp.60–68, Jan. 2016.

[37] T. Maekawa, A. Kwamura, T. Nakachi, and H. Kiya, "Privacy-preserving support vector machine computing using random unitary transformation," IEICE Trans. Fundamentals, vol.E102-A, no.12, pp.1849–1855, 2019.

[38] M. Chen and M. Wu, "Protect your deep neural networks from piracy," 2018 IEEE International Workshop on Inf. Forensics Security (WIFS), pp.1–7, 2018.

[39] H. Chen, C. Fu, B.D. Rouhani, J. Zhao, and F. Koushanfar, "Deepattest: An end-to-end attestation framework for deep neural networks," 2019 ACM/IEEE 46th Annual International Symposium on Computer Architecture (ISCA), pp.487–498, Junet 2019.

[40] Y. Uchida, Y. Nagai, S. Sakazawa, and S. Satoh, "Embedding watermarks into deep neural networks," Proc. 2017 ACM on Int. Conf. Multimedia Retrieval, ser. ICMR'17, pp.269–277, Association for Computing Machinery, June 2017.

[41] H. Chen, B.D. Rouhani, C. Fu, J. Zhao, and F. Koushanfar, "Deepmarks: A secure fingerprinting framework for digital rights management of deep learning models," Proc. 2019 Int. Conf. Multimedia Retrieval, ser. ICMR'19, p.105–113, Association for Computing Machinery, June 2019.

[42] B.D. Rouhani, H. Chen, and F. Koushanfar, "Deepsigns: A generic watermarking framework for IP protection of deep learning models," arXiv preprint arXiv:1804.00750, 2018. [Online]. Available: https://arxiv.org/abs/1804.00750

[43] L. Fan, K.W. Ng, C.S. Chan, and Q. Yang, "DeepIPR: Deep neural network intellectual property protection with passports," IEEE Trans. Pattern Anal. Mach. Intell., vol.44, no.10, pp.6122–6139, Oct. 2021.

[44] Y. Adi, C. Baum, M. Cisse, B. Pinkas, and J. Keshet, "Turning your weakness into a strength: Watermarking deep neural networks by backdooring," Proc. 27th USENIX Conference on Security Symposium, ser. SEC'18, pp.1615–1631, USENIX Association, Aug. 2018.

[45] J. Zhang, Z. Gu, J. Jang, H. Wu, M.P. Stoecklin, H. Huang, and I. Molloy, "Protecting intellectual property of deep neural networks with watermarking," Proc. 2018 on Asia Conference on Computer and Communications Security, ser. ASIACCS'18, pp.159–172, Association for Computing Machinery, May 2018.

[46] S. Sakazawa, E. Myodo, K. Tasaka, and H. Yanagihara, "Visual decoding of hidden watermark in trained deep neural network," 2019 IEEE Conf. Multimedia Information Processing and Retrieval (MIPR), pp.371–374, 2019.

[47] E. Le Merrer, P. Pérez, and G. Trédan, "Adversarial frontier stitching for remote neural network watermarking," Neural Computing and Applications, vol.32, no.13, pp.9233–9244, 2019.

[48] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," Proc. 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pp.4171–4186, Association for Computational Linguistics, June 2019.

[49] I. Tolstikhin, N. Houlsby, A. Kolesnikov, L. Beyer, X. Zhai, T. Unterthiner, J. Yung, A.P. Steiner, D. Keysers, J. Uszkoreit, M. Lucic, and A. Dosovitskiy, "MLP-mixer: An all-MLP architecture for vision," Advances in Neural Information Processing Systems, A. Beygelzimer, Y. Dauphin, P. Liang, and J.W. Vaughan, eds., 2021.

[50] H. Touvron, P. Bojanowski, M. Caron, M. Cord, A. El-Nouby, E. Grave, A. Joulin, G. Synnaeve, J. Verbeek, and H. Jégou, "ResMLP: Feedforward networks for image classification with data-efficient training," CoRR, vol.abs/2105.03404, 2021.

[51] S. Chen, E. Xie, C. GE, R. Chen, D. Liang, and P. Luo, "CycleMLP: A MLP-like architecture for dense prediction," Int. Conf. Learning Representations, 2022. [Online]. Available: https://openreview.net/forum?id=NMEceG4v69Y

[52] H. Liu, Z. Dai, D. So, and Q.V. Le, "Pay attention to MLPs," Advances in Neural Information Processing Systems, A. Beygelzimer, Y. Dauphin, P. Liang, and J.W. Vaughan, eds., 2021.

[53] Q. Hou, Z. Jiang, L. Yuan, M.-M. Cheng, S. Yan, and J. Feng, "Vision Permutator: A permutable MLP-like architecture for visual recognition," IEEE Trans. Pattern Anal. Mach. Intell., 2022.

[54] A. Trockman and J.Z. Kolter, "Patches are all you need?" arXiv preprint arXiv:2201.09792, 2022. [Online]. Available: https://arxiv.org/abs/2201.09792

[55] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," Advances in Neural Information Processing Systems 32, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, eds., pp.8024–8035, Curran Associates, Inc., 2019.

[56] M. Tanaka, "Learnable image encryption," 2018 IEEE Int. Conf. Consumer Electronics-Taiwan (ICCE-TW), pp.1–2, 2018.

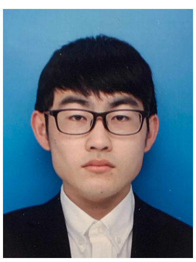
[57] W. Sirichotedumrong, T. Maekawa, Y. Kinoshita, and H. Kiya, "Privacy-preserving deep neural networks with pixel-based image encryption considering data augmentation in the encrypted domain," 2019 IEEE Int. Conf. Image Process. (ICIP), pp.674–678, 2019.

[58] W. Sirichotedumrong and H. Kiya, "A gan-based image transformation scheme for privacy-preserving deep neural networks," 2020 28th European Signal Processing Conference (EUSIPCO), pp.745–749, 2021.

[59] H. Ito, Y. Kinoshita, M. Aprilpyone, and H. Kiya, "Image to perturbation: An image transformation network for generating visually protected images for privacy-preserving deep neural networks," IEEE Access, vol.9, pp.64629–64638, 2021.

[60] B. Schneier and P. Sutherland, Applied Cryptography: Protocols, Algorithms, and Source Code in C, 2nd ed. USA: John Wiley and Sons, Inc., 1995.

Hitoshi Kiya received his B.E and M.E. degrees from the Nagaoka University of Technology in 1980 and 1982, respectively, and his Dr. Eng. degree from Tokyo Metropolitan University in 1987. In 1982, he joined Tokyo Metropolitan University, where he became a Full Professor in 2000. From 1995 to 1996, he attended the University of Sydney, Australia as a Visiting Fellow. He is a Fellow of IEEE, IEICE, AAIA and ITE. He served as President of APSIPA from 2019 to 2020 and as Regional Director-at-Large for Region 10 of the IEEE Signal Processing Society from 2016 to 2017. He was also President of the IEICE Engineering Sciences Society from 2011 to 2012, and he served there as Editor in-Chief for IEICE Society Magazine and Society Publications. He has been an Editorial Board Member of eight journals, including IEEE Trans. on Signal Processing, Image Processing, and Information Forensics and Security, Chair of two technical committees, and Member of nine technical committees including the APSIPA Image, Video, and Multimedia Technical Committee (TC) and IEEE Information Forensics and Security TC. He has organized a lot of international conferences in such roles as TPC Chair of IEEE ICASSP 2012 and as General Co-Chair of IEEE ISCAS 2019. He has received numerous awards, including 12 best paper awards.



Ryota Iijima received his B.C.S degree from Tokyo Metropolitan University, Japan in 2022. Since 2022, he has been a Master course student at Tokyo Metropolitan University. His research interests include deep neural networks and their protection.

AprilPyone MaungMaung received a BCS degree from the International Islamic University Malaysia in 2013 under the Albukhary Foundation Scholarship, MCS degree from the University of Malaya in 2018 under the International Graduate Research Assistantship Scheme, and Ph.D. degree from the Tokyo Metropolitan University in 2022 under the Tokyo Human Resources Fund for City Diplomacy Scholarship. He is currently working as a Project Assistant Professor in the Tokyo Metropolitan University and as a researcher in rinna Co. Ltd. He received an IEEE ICCE-TW Best Paper Award in 2016. His research interests are in the area of adversarial machine learning and information security. He is a member of IEEE.

Yuma Kinoshita received his B.Eng., M.Eng., and the Ph.D. degrees from Tokyo Metropolitan University, Japan, in 2016, 2018, and 2020 respectively. In April 2020, he started to work with Tokyo Metropolitan University, as a project assistant professor. He moved to Tokai University, Japan, as an associate professor/lecturer in April 2022. His research interests are in the area of signal processing, image processing, and machine learning. He is a Member of IEEE, APSIPA, IEICE, and ASJ. He received the IEEE ISPACS Best Paper Award, in 2016, the IEEE Signal Processing Society Japan Student Conference Paper Award, in 2018, the IEEE Signal Processing Society Tokyo Joint Chapter Student Award, in 2018, the IEEE GCCE Excellent Paper Award (Gold Prize), in 2019, and the IWAIT Best Paper Award, in 2020. He was a Registration Chair of DCASE2020 Workshop.