Privacy-Preserving Transfer Learning using Learnable Encryption

Ijaz Ahmad[a], Joongheon Kim[a], and Seokjoo Shin[b]
[a]Dept. of Electrical and Computer Engineering, Korea University, Seoul, Korea
[b]Dept. of Computer Engineering, Chosun University, Gwangju, Korea
ijaz@korea.ac.kr, joongheon@korea.ac.kr, sjshin@chosun.ac.kr (*Corresponding author*)

***Abstract*—In a cloud-edge collaboration for Internet-of-Things (IoT), transfer learning can significantly enhance personalized context learning. This approach allows models trained on broader datasets in the cloud to be adapted for specific devices and scenarios at the edge, leading to more accurate and efficient Artificial Intelligence (AI) applications within the IoT ecosystem. However, to comply with privacy regulations, the data must be protected throughout the AI development lifecycle, from collecting data initially to storing and using it in the final model. This necessitates privacy-preserving computation techniques to minimize the exposure of sensitive information. Therefore, we implement a lightweight privacy-preserving transfer learning scheme based on learnable encryption to adapt a pre-trained model to local data characteristics in a privacy-preserving manner. Simulation analysis shows that personalized context can be learned within learnable encryption ciphertext, achieving significant improvements in pre-trained model performance i.e., the improvements range from 16% to 30%.**

***Index Terms*—artificial intelligence, cloud-edge collaboration, internet-of-things, learnable encryption, transfer learning.**

I. Introduction

Cloud-edge collaboration architecture is a distributed computing paradigm that optimizes the allocation of computational tasks between powerful, centralized cloud servers and geographically closer, resource-constrained edge devices. This architecture leverages the distinct advantages of both environments to achieve reduced latency, improved accuracy, enhanced privacy, scalability and resource efficiency for various applications, especially those involving artificial intelligence (AI). In general, AI has two main phases: a training phase, which is computationally intensive, and an inference phase, which requires less power. For Internet-of-Things (IoT), it is efficient to train AI models on powerful cloud servers and then deploy them for inference on edge devices, in the proximity of where data is generated. In addition, pre-trained models can be fine-tuned on the edge to be adapted to the evolving context of IoT. For example, [1] proposed a strategy that involved creating tailored models for individual edge devices by deriving sub-models from a larger, pre-trained model for several applications such as, human activity recognition, image classification and speech recognition. However, a significant challenge remains in ensuring privacy when handling sensitive data on edge devices.

Towards this, privacy-preserving personalized context learning techniques are proposed based on federated learning (e.g, [2] and [3]), homomorphic encryption (e.g, [4]), differential privacy (e.g, [5]) and learnable encryption (e.g., [6]). These methods allow for personalized model training without directly sharing raw data, enhancing both privacy and model performance. However, they still face limitations such as high communication costs and the need for encryption in federated learning, high computational costs in homomorphic encryption, challenges in directly applying differential privacy to images, and the need to balance privacy and usability in learnable encryption.

Despite its privacy-utility tradeoff, learnable encryption [6]–[12] offers a lightweight data manipulation approach to privacy-preserving deep learning, making it suitable for resource-constrained devices like IoT end-devices. It hides plaintext contents by modifying only selected pixel values while retaining its semantics to enable DL model computation in the encrypted domain. This approach is particularly useful for scenarios where strict cryptographic security is not paramount, but resource efficiency and compatibility with existing DL models are important. Existing works in learnable encryption mainly focus on ensuring privacy during the initial model training and inference phases, often ignoring the crucial aspect of adapting the trained model to the local data characteristics. Therefore, in this study we implement and analyze several learnable encryption algorithms to design a lightweight privacy-preserving transfer learning framework for personalized context learning in cloud-edge assisted IoT.

II. Methods

A. Learnable Encryption

The core idea of learnable encryption is to hide plaintext contents by modifying only selected pixel values while retaining its semantics to enable privacy-preserving computation of various applications. Based on this principle several learnable encryption algorithms such as [9], [10] and [6] are proposed. Let $\mathbf{P}$ is a plaintext image and $\mathbf{C}$ is its ciphtertext image, each with $N = H \times W$ pixels, H rows and W columns. Also, their elements are indexed as $\left(\lfloor \frac{i}{H} \rfloor, i \bmod W\right)$. Then, learnable encryption function proposed in [9], [10] and [6] can be defined as below.

Sirichotedumrong *et al.*'s scheme [9]. This is a two-step learnable encryption scheme where in the first step half of the pixel values of the image $\mathbf{P}$ are modified as given in (1) to produce an intermediate output $\hat{\mathbf{C}}$.

$$\hat{\mathbf{C}}_{(i)} = \begin{cases} \mathbf{P}_{(i)} \oplus 255, & \text{if } \mathbf{K}^1_{(i)} = 1, \\ \mathbf{P}_{(i)}, & \text{if } \mathbf{K}^1_{(i)} = 0, \end{cases} \tag{1}$$

where $\mathbf{K}^1 \in \mathbb{Z}_2$ is a random binary key. For a color image, this process is performed in each color component using the same key. Then, in the next step, color channels are shuffled using a random key $\mathbf{K}^2 \in \mathbb{Z}_6$. Each element of the key $\mathbf{K}^2$ represents a unique permutation. This color channel shuffling function can be defined as,

$$\mathbf{C}_{(i)} = f\left(\hat{\mathbf{C}}_{(i)}, \mathbf{K}^2_{(i)}\right). \tag{2}$$

Huang *et al.*'s scheme [10]. The simple operations of [9]'s scheme makes it vulnerable to data reconstruction attacks as demonstrated in [13]. Therefore, Huang *et al.* [10] proposed to further process the intermediate output $\hat{\mathbf{C}}$ produced in (1) by dividing it into square blocks and perform smoothing on each block using different filters as follows,

$$\mathbf{C}_{(\mathbf{b})} = g(\mathbf{B}_{(\mathbf{b})}, m, \mathbf{K}^2_{(i)})\}^B_{b=0}, \tag{3}$$

where $\mathbf{B}_{(b)} \in \hat{\mathbf{C}}$ is the b^{th} image block, and $m \in \mathcal{M}$ is a set that consists of *identity*, *mean*, *median*, *max* and *min* filters. Also, $\mathbf{K}^2 \in \mathbb{Z}^B_5$ is a key to randomly choose a filter.

Ahmad *et al.*'s scheme [6]. Alternatively, Ahmad *et al.* [6] proposed a learnable encryption algorithm that consists of two steps. First, selected pixel values are mixed with a random key $\mathbf{K}^1 \in \mathbb{Z}^N_{256}$ and previously encrypted pixel value c to produce an intermediate output $\hat{\mathbf{C}}$ as,

$$\hat{\mathbf{C}}_{(i)} = \begin{cases} \mathbf{K}^1_{(i)} \oplus \left\{ \begin{matrix} \left(\mathbf{P}_{(i)} + \mathbf{K}^1_{(i)}\right) \\ \bmod 256 \end{matrix} \right\} \oplus c, & \text{if } \mathbf{K}_{(i)} > 0, \\ \mathbf{P}_{(i)} \oplus \gamma, & \text{if } \mathbf{K}_{(i)} = 0, \end{cases} \tag{4}$$

where the parameter $\gamma \in [0, 255]$, and $\gamma = 0$ leaves a pixel value unmodified while $\gamma = 255$ fully inverts a pixel value. Consequently, γ controls the visibility of the image contents. For additional security, c value can be randomly chosen from the previously encrypted or plaintext pixel values using $\mathbf{K}^2 \in \mathbb{Z}^N_N$ as follows,

$$c = \begin{cases} \hat{\mathbf{C}}_{\left(\mathbf{K}^2_{(i)}\right)}, & \text{if } \mathbf{K}^2_{(i)} < i, \\ \mathbf{P}_{\left(\mathbf{K}^2_{(i)}\right)}, & \text{if } \mathbf{K}^2_{(i)} \geq i. \end{cases} \tag{5}$$

Then, in the second step, bitplane manipulation is performed to modify all pixel values using $\mathbf{K}^3 \in \mathbb{Z}^N_N$ as follows,

$$\mathbf{C}_{(i)} = \begin{cases} \hat{\mathbf{C}}_{(i)} \oplus \left(\mathbf{C}_{\left(\mathbf{K}^3_{(i)}\right)} \bmod \beta\right), & \text{if } \mathbf{K}^3_{(i)} < i, \\ \hat{\mathbf{C}}_{(i)} \oplus \left(\hat{\mathbf{C}}_{\left(\mathbf{K}^3_{(i)}\right)} \bmod \beta\right), & \text{if } \mathbf{K}^3_{(i)} \geq i, \end{cases} \tag{6}$$

where the parameter β controls the number of least significant bits to be modified in a pixel. Specifically, its value is chosen based on the number of bits required to represent the pixel $\hat{\mathbf{C}}_{(i)}$ to restrict the loss of information for learnability.

B. Proposed Privacy-Preserving Transfer Learning

1) Privacy-preserving training: For an image classification task, let $\mathcal{D} := \{[\mathbf{X}_i], [\mathbf{y}_i]\}$ be a dataset of N samples where $\mathcal{X} := [\mathbf{X}_1, \ldots, \mathbf{X}_N] \in \mathbb{R}^{H \times W}$ is the input space containing images as high dimensional vectors and $\quad := [y_1, \ldots, y_N] \in \mathbb{R}^k$ is the output space containing k image categories. In general, the dataset $\mathcal{D}$ is divided into two disjoint subsets $\mathcal{D}^s$ and $\mathcal{D}^t$, where $(x, y) \in \mathcal{D}^s$ is used for training a model, while $(x, y) \in \mathcal{D}^t$ is used for evaluating the trained model. During training, a parameterized neural network function f_θ updates its parameters θ for mapping the input data $\mathcal{X}$ to the output as closely as possible, i.e., $\{f_\theta : \mathcal{X} \rightarrow \quad | f_\theta(x_i) \approx y_i\}$, by minimizing a loss function L. The loss function $L(f_\theta)$ measures the distance of the model output $\hat{} \leftarrow f_\theta(\mathcal{X})$ from the actual output $\quad$. The training of a neural network can be represented as a tuple of four elements $(\mathcal{D}^s, f, \theta_0, g)$, where f is the neural network architecture, $\mathcal{D}^s$ is the training data, θ_0 is the set of initial parameters, and g is a training function. The function g is a process of learning a model based on $\mathcal{D}^s$ given the initial parameters θ_0 to obtain a trained neural network f_θ with learned parameters θ as $f_\theta \leftarrow f_{\theta_0}(\mathcal{D}^s)$. On the other hand, the samples $(x, y) \in \mathcal{D}^t$ are used to evaluate the trained model f_θ performance in how closer the predicted values $f_\theta(x)$ are to the actual output y, which can be defined by the probability given as follows,

$$\Pr\left[f_\theta(x) \rightarrow y\right], \text{where } (x, y) \in \mathcal{D}^t. \tag{7}$$

For privacy-preserving training, let h be an encryption function that encrypts the dataset $\mathcal{D}$ with a secret key $\mathcal{K}$ as $\mathcal{C} = \{(h(x, \mathcal{K}), y) : (x, y) \in \mathcal{D}\}$. Subsequently, $\mathcal{C}^s$ and $\mathcal{C}^t$ are the training and test sets, respectively. Then, the privacy-preserving training of a neural network is as a tuple $(\mathcal{C}^s, f, \theta^*_0, g)$, indicating that a trained neural network f_{θ^*} with learned parameters θ^* is obtained in a privacy-preserving manner, i.e., $f_{\theta^*} \leftarrow f_{\theta^*_0}(\mathcal{C}^s)$. Similar to non-secure training, the accuracy of f_{θ^*} can be measured by computing its probability on the encrypted test set $\mathcal{C}^t$ as follows,

$$\Pr\left[f_{\theta^*}(x) \rightarrow y\right], \text{where } (x, y) \in \mathcal{C}^t. \tag{8}$$

2) Privacy-preserving transfer learning: In general, a model consists of an input and output layer, and at least one hidden layer. The dimension of the input layer is the input size $H \times W$, and the dimension of the output layer is the output size k. For image classification, the model f usually has a feature extractor module consisting of multiple convolution layers, followed by a classifier module, comprised of fully connected layers. The classifier module is inherently task-specific while the feature extractor module can be shared across different tasks. Thus allowing us to reuse the feature extractor, having learned general representations from a large dataset on a source task $\ _1$, for a relevant target task $\ _2$. The ability to reuse a pre-trained feature extractor in this way is often known as *transfer learning*. For a set of tasks $\quad = [\ _1, \ _2]$, the training set can be defined as, $\mathcal{C}^{s,\mathcal{T}} := \{[\mathcal{X}^{s,\mathcal{T}}], [\ ^{s,\mathcal{T}}]\}$ and test set as, $\mathcal{C}^{t,\mathcal{T}} := \{[\mathcal{X}^{t,\mathcal{T}}], [\ ^{t,\mathcal{T}}]\}$. Here, the dataset is encoded using

one of the learnable encryption functions outlined in Section II-A, to preserve privacy. Also, the objective function can be defined as follows,

$$\min_{f_\theta^{\mathcal{T}} \in F_\theta^{\mathcal{T}}} \mathcal{L}\left(f_\theta^{\mathcal{T}}\right) = \min_{f_\theta^{\mathcal{T}} \in F_\theta^{\mathcal{T}}} \mathbb{E}\left[L_{\mathcal{T}}\left(\ \ _{\mathcal{T}}, f_\theta^{\mathcal{T}}\left(\mathcal{X}_{\mathcal{T}}\right)\right)\right], \quad (9)$$

where $f_\theta^{\mathcal{T}} \in F_\theta^{\mathcal{T}}$ is the best estimator such that $\{f_\theta^{\mathcal{T}} : \mathcal{X}^{\mathcal{T}} \rightarrow \ ^{\mathcal{T}} | f_\theta^{\mathcal{T}}(x_i^{\mathcal{T}}) = y_i^{\mathcal{T}}\}$ for a task , and $\mathcal{L}_{\mathcal{T}}(f_\theta^{\mathcal{T}})$ is the loss function that measures a model $f_\theta^{\mathcal{T}} : \mathcal{X}_{\mathcal{T}} \rightarrow \ _{\mathcal{T}}$ for a given task . The goal here is to leverage the pre-trained model $f_\theta^{\mathcal{T}_1}$ from the source task to find the best estimator $f_\theta^{\mathcal{T}_2}$ for the target task. However, prior to this, transformation of the input and output is necessary because the input $\mathcal{X}^{\mathcal{T}}$ and output $^{\mathcal{T}}$ dimensions may differ for each task for example, the input image resolution and the number of categories may vary from task $_1$ to task $_2$. Therefore, based on the mathematical framework proposed in [14], we describe the transfer learning procedure in the following three steps.

(Step 1) Input transformation. When using a pre-trained model, the input in the target task must be processed to match the properties such as, image resolution, pixel value normalization etc., of the source task input. Therefore, an input transformation function $g^{\mathbf{X}}$ can be defined as mapping a sample from the target input space $\mathcal{X}^{\mathcal{T}_2}$ into the source input space $\mathcal{X}^{\mathcal{T}_1}$ such that $g^{\mathbf{X}}(\mathbf{X}^{\mathcal{T}_2}) \in \mathcal{X}^{\mathcal{T}_1}$. For example, the $g^{\mathbf{X}}$ can be a resizing function to meet the spatial and/or spectral dimension of the model input. It may also include the encryption function $h(.,\mathcal{K})$ used for protecting $\mathcal{C}^{\mathcal{T}_1}$.

(Step 2) Applying pretrained model. In this step, the transformed data $g^{\mathbf{X}}(\mathbf{X}^{\mathcal{T}_2})$ from the previous step is used as an input to the pre-trained model $f_\theta^{\mathcal{T}_1}$ that maps it into the source output space $^{\mathcal{T}_1}$ as $f_\theta^{\mathcal{T}_1}\left(g^{\mathbf{X}}(\mathbf{X}^{\mathcal{T}_2})\right) \in \ ^{\mathcal{T}_1}$.

(Step 3) Output transformation. A pre-trained model's classifier is task specific, which may not directly translate to a new task because of different label space. This discrepancy necessitates a transformation to adapt the pre-trained model's output for the different target task. An output transformation function g^{Y} can be defined as mapping Step 2's output into the target output space $^{\mathcal{T}_2}$ as $g^{Y}\left(f_\theta^{\mathcal{T}_1}\left(g^{\mathbf{X}}\left(\mathbf{X}^{\mathcal{T}_2}\right)\right)\right) \in \ ^{\mathcal{T}_2}$. This transformation may include attaching additional layers after the feature extractor of $f_\theta^{\mathcal{T}_1}$ and fine-tuning them for the target task $_2$.

For the transfer learning, the best estimator $f_\theta^{\mathcal{T}_2}$ to achieve objective in (9) for the target task $_2$ is given as follows,

$$f_\theta^{\mathcal{T}_2} = g^{Y}\left(.,\left(f_\theta^{\mathcal{T}_1} \quad g^{\mathbf{X}}\left(\mathbf{X}^{\mathcal{T}_2}\right)\right)\right), \quad (10)$$

and $f_\theta^{\mathcal{T}_1}$ is trained via direct learning for the source task $_1$.

III. Simulation Results

A. System Model

This study considers a cloud-edge collaboration that allows for a balance between resource utilization and performance. For example, the cloud can train large and complex models, while the edge can handle real-time inference and transfer learning closer to the data source. Therefore, this cloud-edge collaboration paradigm for AI in IoT ecosystem involves a system model with three entities i.e., IoT end devices, edge servers, and cloud servers. The *IoT end devices* are responsible for capturing data from the physical environment and performing initial preprocessing steps such as encryption prior to transmitting it to other entities. The *edge servers* provide the computational resources necessary to perform tasks such as model inference and transfer learning. Finally, the *cloud servers* are capable of handling the training of initial models.

B. Datasets

For the initial model training we use CIFAR10 dataset that consists of 60K color images divided among 10 distinct classes, each with a size of 32×32 pixels. Also, for privacy-preserving transfer learning analysis we use Kaggle Dogs vs. Cats dataset. Although the dataset consists of 25K color images of dimensions 224×224, we choose a subset of 2K images from this dataset to better simulate limited data scenario. Consequently, demonstrating a more realistic and effective simulation of the advantages and challenges of the transfer learning.

C. DL-based Classification Model

The primary advantage of learnable encryption is its direct compatibility with state-of-the-art deep learning models. Among the existing models, EfficientNet [15] (EfficientNetV1) is a family of lightweight convolutional neural network models optimized for high efficiency in terms of parameters and Floating Point Operations (FLOPs). It takes advantage of neural architecture search (NAS) to design a baseline EfficientNet-B0 while aiming to find an optimal tradeoff between accuracy and computational cost (FLOPs). Instead of independently scaling network depth, width, or resolution, EfficientNet introduces a simplified yet an effective compound scaling strategy that uniformly scales up all three dimensions to obtain a family of models B1-B7. Consequently, each subsequent model (B0 to B7) represents a scaled-up version of the baseline, offering a balance between higher accuracy and increase computational requirements. Compared to its predecessor, a key advantage of EfficientNet models lies in their efficiency i.e., they achieve superior accuracy with significantly fewer parameters and FLOPs. Therefore, in this work we consider EfficientNetV1 model to analyze the efficiency of different learnable encryption schemes in privacy-preserving transfer learning. For model training setup, we refer to [6].

D. Performance Analysis

This subsection divides the simulation analysis into two parts: privacy-preserving initial model training and privacy-preserving transfer learning. The privacy-preserving initial model training analyzes the performance of each learnable encryption in training the initial model. This trained model then supports the subsequent privacy-preserving transfer leaning analysis, which involves analyzing and adapting the initial model to the local data characteristics. Furthermore, for each learnable encryption technique, we use two sets of parameters.

TABLE I
PERFORMANCE ANALYSIS OF LEARNABLE ENCRYPTION SCHEMES FOR PRIVACY-PRESERVING TRANSFER LEARNING ANALYSIS. $\mathcal{S} \in \{\mathcal{D}, \mathcal{C}\}$ DENOTES THE PLAINTEXT AND CIPHERTEXT DATASETS.

Methods	$f_\theta^{\tau_1}(\mathcal{S}^{\tau_1})$	$f_\theta^{\tau_1}(\mathcal{S}^{\tau_2})$	$f_\theta^{\tau_2}(\mathcal{S}^{\tau_2})$
Plain	93.60	76.56	95.31
[9]-1	86.93	70.31	86.46
[9]-2	83.70	59.38	83.33
[10]-1	83.93	54.69	84.38
[10]-2	73.97	33.33	77.08
[6]-P1	85.80	69.79	86.46
[6]-P4	78.80	65.10	85.94

For example, [9]-1 implements only one step while [9]-2 implements both steps of [9]'s learnable encryption method. Similarly, [10]-1 and [10]-2 use encryption block size 4×4 and 8×8, respectively. Also, we use the same pseudonym of [6]. These analyses are reported in Table I and discussed below.

1) Privacy-Preserving Initial Model Training: Table I presents the initial model ($f_\theta^{\tau_1}$) trained for task τ_1 performance on the plaintext images (i.e., $\mathcal{S}^{\tau_1} \in \mathcal{D}^{t,\tau_1}$) and ciphertext images (i.e., $\mathcal{S}^{\tau_1} \in \mathcal{C}^{t,\tau_1}$). Among the compared learnable encryption techniques, [9] achieved the best accuracy when only half of the pixel values were randomly inverted (i.e, [9]-1). However, incorporating color channel shuffling as in [9]-2 introduced a 3% error. On the other hand, the blockwise operation proposed in [10] to deal with the vulnerabilities of [9], preserved [9]-2's performance as in [10]-1 at best. However, using a larger block size for the improved security in [10]-2 drastically decreased the model performance. Nonetheless, [10] provided a better security and usability tradeoff compared to [9]. Furthermore, the parameterized learnable encryption [6]-P1 achieved a comparable performance to [9]-1 with better security. Although the performance degradation from [6]-P1 to [6]-P4 is not as drastic as [10], there is still a 7% accuracy difference. As suggested in [6] that a model learned on plain ImageNet dataset may not necessarily correspond to cipher-image feature space; therefore, the model's superior performance on plain-images i.e., $f_\theta^{\tau_1}(\mathcal{S}^{\tau_1} \in \mathcal{D}^{\tau_1})$, can be attributed to its initialization strategy.

2) Privacy-Preserving Transfer Learning: For this analysis, two sets of experiments were conducted. First, the initial trained model $f_\theta^{\tau_1}$ was directly used to infer labels in the target task domain without context learning i.e., $f_\theta^{\tau_1}(\mathcal{S}^{\tau_2} \in \mathcal{D}^{\tau_2})$ for non-secure inference and $f_\theta^{\tau_1}(\mathcal{S}^{\tau_2} \in \mathcal{C}^{\tau_2})$ for secure inference. Second, the model $f_\theta^{\tau_1}$ was adapted to task τ_2 using transfer learning as $f_\theta^{\tau_2}(\mathcal{S}^{\tau_2} \in \mathcal{D}^{\tau_2})$ and privacy-preserving transfer learning as $f_\theta^{\tau_2}(\mathcal{S}^{\tau_2} \in \mathcal{C}^{\tau_2})$. For both experiments, the results are given in Table I. It can be observed that in the first scenario the model performance drastically degraded for both non-secure and secure inference due to the context difference. Nonetheless, the model's accuracy error is completely eliminated after being adapted to local data characteristics in the second case. Thus, showing successful application of transfer learning in both non-secure and secure scenarios. Importantly, the learnable encryption techniques that provided higher security benefited the most from the transfer learning for example, the performance improvement is 3.1% for [10]-2 while 7.14% for [6]-P4 compared to the initial trained model.

IV. CONCLUSION

This study proposed a privacy-preserving transfer learning strategy based on learnable encryption to adapt a pre-trained model to the IoT context, while simultaneously ensuring data privacy. The simulation analysis on two datasets shown that learnable encryption can cater to privacy requirements of AI applications within IoT environments. In the future, we are interested to consider a special case of transfer learning called *domain adaptation* in a privacy-preserving manner using learnable encryption.

ACKNOWLEDGMENT

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government. (MSIT) (RS-2023-00278294).

REFERENCES

[1] Y. Zhuang, Z. Zheng, Y. Shao, B. Li, F. Wu, and G. Chen, "ECLM: Efficient edge-cloud collaborative learning with continuous environment adaptation," *arXiv preprint arXiv:2311.11083*, 2023.

[2] Q. Wu, K. He, and X. Chen, "Personalized federated learning for intelligent IoT applications: A cloud-edge based framework," *IEEE Open Journal of the Computer Society*, vol. 1, pp. 35–44, 2020.

[3] S. Hu, J. Lin, Z. Lu, X. Du, Q. Duan, and S.-C. Huang, "CoLLaRS: A cloud–edge–terminal collaborative lifelong learning framework for AIoT," *Future Generation Computer Systems*, vol. 158, pp. 447–456, 2024.

[4] X. Yu, D. Tang, and W. Zhao, "Privacy-preserving cloud-edge collaborative learning without trusted third-party coordinator," *Journal of Cloud Computing*, vol. 12, no. 1, p. 19, 2023.

[5] K.-H. Le, K.-H. Le-Minh, and H.-T. Thai, "BrainyEdge: An AI-enabled framework for iot edge computing," *ICT Express*, vol. 9, no. 2, pp. 211–221, 2023.

[6] I. Ahmad, J. Kim, and S. Shin, "Privacy-preserving uncertainty calibration using perceptual encryption in cloud-edge collaborative artificial intelligence of things," *IEEE Internet of Things Journal*, 2025.

[7] M. Tanaka, "Learnable image encryption," in *2018 IEEE International Conference on Consumer Electronics-Taiwan (ICCE-TW)*. IEEE, 2018, pp. 1–2.

[8] K. Madono, M. Tanaka, M. Onishi, and T. Ogawa, "Block-wise scrambled image recognition using adaptation network," in *AAAI-20 Workshop on Artficial Intelligence of Things (AAAI-WAIoT)*, 2020.

[9] W. Sirichotedumrong, Y. Kinoshita, and H. Kiya, "Pixel-based image encryption without key management for privacy-preserving deep neural networks," *IEEE Access*, vol. 7, pp. 177 844–177 855, 2019.

[10] Q.-X. Huang, W. L. Yap, M.-Y. Chiu, and H.-M. Sun, "Privacy-preserving deep learning with learnable image encryption on medical images," *IEEE Access*, vol. 10, pp. 66 345–66 355, 2022.

[11] Ł. Krzywiecki, G. Zaborowski, and M. Zawada, "Too noisy, or not too noisy? a private training in machine learning," in *2023 IEEE 22nd International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*. IEEE, 2023, pp. 149–156.

[12] I. Ahmad, J. Kim, and S. Shin, "Learnable encryption with a diffusion property," in *2025 55th Annual IEEE/IFIP International Conference on Dependable Systems and Networks-Supplemental Volume (DSN-S)*. IEEE, 2025, pp. 259–260.

[13] A. H. Chang and B. M. Case, "Attacks on image encryption schemes for privacy-preserving deep neural networks," *arXiv preprint arXiv:2004.13263*, 2020.

[14] H. Cao, H. Gu, and X. Guo, "Feasibility of transfer learning: A mathematical framework," *arXiv preprint arXiv:2305.12985*, 2023.

[15] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *International conference on machine learning*. PMLR, 2019, pp. 6105–6114.