Efficient Artificial Intelligence With Novel Matrix Transformations and Homomorphic Encryption

Quoc Bao Phan, *Graduate Student Member, IEEE*, and Tuy Tan Nguyen, *Member, IEEE*

Abstract—**This paper addresses the challenges of data privacy and computational efficiency in artificial intelligence (AI) models by proposing a novel hybrid model that combines homomorphic encryption (HE) with AI to enhance security while maintaining learning accuracy. The novelty of our model lies in the introduction of a new matrix transformation technique that ensures compatibility with both HE algorithms and AI model weight matrices, significantly improving computational efficiency. Furthermore, we present a first-of-its-kind mathematical proof of convergence for integrating HE into AI models using the adaptive moment estimation optimization algorithm. The effectiveness and practicality of our approach for training on encrypted data are showcased through comprehensive evaluations of well-known datasets for air pollution forecasting and forest fire detection. These successful results demonstrate high model performance, with nearly 1 R-squared for air pollution forecasting and 99% accuracy for forest fire detection. Additionally, our approach achieves a reduction of up to 90% in data storage and a tenfold increase in speed compared to models that do not use the matrix transformation method. Our primary contribution lies in enhancing the security, efficiency, and dependability of AI models, particularly when dealing with sensitive data.**

Index Terms—**Artificial intelligence, homomorphic encryption, matrix transformation, convergence analysis.**

I. Introduction

The development of methods leveraging artificial intelligence (AI) is demonstrably powerful, as evidenced by various studies [1], [2]. These advancements span multiple fields, including time series forecasting, audio signal processing, wireless communications, and computer vision applications. In each domain, AI's ability to extract valuable insights from these vast datasets is driving significant improvements in both theoretical understanding and practical applications. However, a significant challenge arises: as AI models become increasingly sophisticated and capable, their computational demands and size also grow exponentially. This poses new hurdles for researchers and practitioners who must grapple with efficient training and deployment of these powerful, but resource-intensive, models [3].

Received 6 June 2024; revised 24 July 2024 and 9 September 2024; accepted 19 September 2024. Date of publication 24 September 2024; date of current version 13 December 2024. This work was supported by the National Science Foundation under Grant 2348464. This article was recommended by Guest Editor Pin-Yu Chen. *(Corresponding author: Tuy Tan Nguyen.)*

The authors are with the School of Informatics, Computing, and Cyber Systems, Northern Arizona University, Flagstaff, AZ 86011 USA (e-mail: bqp7@nau.edu; tuy.nguyen@nau.edu).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/JETCAS.2024.3466849.

Digital Object Identifier 10.1109/JETCAS.2024.3466849

To address these challenges, third-party providers have emerged, offering specialized services such as large-scale model training and high-performance model deployment. These providers leverage advanced infrastructure and expertise to handle the computational intensity of modern AI applications. While most guarantee high privacy and security for customer data, concerns remain regarding data leakage during communication and the potential for unauthorized access to the data, especially as the volume of big data explodes [4].

Numerous research efforts have addressed these security concerns by utilizing homomorphic encryption (HE) to protect data integrity and enable computations on encrypted data. Recent studies [5], [6] have proposed methodologies showcasing a significant improvement in data security through this integration. By allowing computations to be performed directly on encrypted data without compromising its confidentiality, HE offers a promising solution to data privacy issues. However, HE also introduces a trade-off, as it increases the complexity and computational demands of AI algorithms [6]. Balancing security and performance remains an ongoing challenge in this area.

In this study, we propose a novel approach that combines matrix transformation techniques with the Cheon-Kim-Kim-Song (CKKS) algorithm [7] within the AI field. To validate the correctness and efficiency of the proposed model, we demonstrate its mathematical convergence and perform simulations on two representative AI problems: air pollution forecasting and forest fire detection. By successfully merging techniques from cryptography and machine learning, this project has the potential to significantly advance secure machine learning. We hope that our research will pave the way for the development and implementation of HE algorithms in AI models in the near future. Our contributions are as follows:

1) We propose two hybrid models combining HE and AI optimized for various real-world problems.
2) We introduce a novel matrix transformation method for both CKKS algorithms and the weight matrices of the proposed AI models.
3) We demonstrate the convergence of integrating HE into AI using the adaptive moment estimation (ADAM) optimization algorithm, which has not been clearly proven mathematically in previous research.
4) We conduct simulations based on real-world data, including air pollution forecasting and wildfire detection, to demonstrate the effectiveness of AI models when trained on encrypted datasets.

The remaining parts of the paper are structured as follows: Section II presents related research, and Section III covers preliminary knowledge of HE algorithms and neural networks. Section IV outlines our proposed developments and implementations. In Section V, we describe the datasets and evaluation criteria, followed by results for two AI problems. Lastly, we conclude the paper in Section VI.

II. Related Work

The field of AI is developing extremely rapidly with many significant advancements. Recent studies in areas such as computer vision and time series forecasting have introduced new proposals to enhance model productivity and efficiency [8], [9], [10]. In [11], the authors introduced a method to transform time series data and partition them based on thresholds to improve the accuracy of forecasting models. Another example in the smart grid sector [12] demonstrated that combining AI models increased accuracy to 98% with real-world data. In the computer vision field, in [13], the authors introduced an optimized framework for patient care, showcasing AI's practical contributions to human health. Another example of AI application in signal recognition was presented in [14], also achieving an accuracy of 98%.

A well-established method for securing data in machine learning is federated learning [15], [16]. In this approach, individual clients participate in the training process without sharing their raw data or a central server. Instead, they exchange AI models with the server, which aggregates and updates the models before sending them back to each client. This technique effectively safeguards sensitive data, ensuring that personal or proprietary information remains with the clients and is not exposed during training. As the need for even greater security has grown, researchers have turned to more advanced methods. Studies such as [17], [18], and [19] have introduced secure multi-party computation (SMPC) to enhance federated learning. SMPC allows multiple parties to jointly compute a function over their inputs while keeping those inputs private, thereby defending against side-channel attacks—a vulnerability in traditional federated learning.

Despite these advancements, a critical challenge remains: the risk of malicious clients. These rogue participants might infiltrate the training process to degrade model accuracy or steal sensitive information from the server. In such cases, encryption algorithms are essential, offering robust protection by ensuring that even if a malicious actor gains access to the process, the encrypted data remains secure and intact. This convergence of federated learning, SMPC, and encryption represents the cutting edge of secure AI model training. However, these advancements come with their own challenges, particularly in terms of increased model complexity and computational bottlenecks [20]. To address this, a novel transformation method proposed in [21] converts model weights into a binary form, significantly reducing the model size and training time without compromising effectiveness. This approach presents a highly practical and innovative solution.

Additionally, a major security concern with AI models is the reliance on servers for training, which necessitates secure communication protocols to maintain data integrity during transmission. To further address security concerns, researchers have developed encryption algorithms combined with AI. Many authors have proposed hybrid models that tackle various AI-related issues, dividing the encryption process into two types: encryption during information transmission and reception, and encryption during model training. In studies such as [22], [23], and [24], encryption methods were introduced for the transmission and reception process, preventing external attacks on information shared between clients and servers. This ensures that third parties cannot access the data post-training without the secret key holder's permission, effectively preventing the leakage of critical information.

Another promising approach in secure AI training is in-training encryption, where both the model and data are encrypted throughout the training process. This method relies on third-party cloud platforms with substantial storage capacity to handle the encrypted data. While involving a third party raises privacy concerns, advancements in SMPC provide promising solutions to these issues. Researchers have begun exploring the possibility of training AI models directly on encrypted datasets, an area of research that has shown significant potential. The study in 2019 [25] introduced a partial machine learning model that operates on encrypted data, integrated with SMPC. This model specifically addressed a critical limitation of many encryption schemes: the inability to perform the final thresholding operation used for classification. The authors proposed an innovative training method designed to protect selected sensitive features from leaking, adversarially optimizing the network to defend against an adversary attempting to identify these features. However, the model's application was limited to the MNIST dataset, and its effectiveness with more complex, real-world data remains to be fully validated.

Further advancements in this area were showcased in the study [26], where they introduced an approximate model capable of performing effectively in image recognition tasks, even when operating on encrypted data. This model represents a significant step forward, as it demonstrates that robust image classification can still be achieved without compromising data security. Building on this foundation, authors in [27] extended these innovative techniques to the realm of audio signal processing. Their work achieved an impressive accuracy, with an error deviation of only around 1% compared to traditional methods that do not employ HE. This achievement highlights the potential of HE to maintain high levels of performance across different types of data, reinforcing its versatility and effectiveness in securely handling sensitive information. Despite these successes, the use of HE algorithms introduces a notable challenge: a substantial increase in computation time. The encryption process significantly enlarges the dataset, which in turn leads to considerable delays and increased resource demands. This computational overhead remains a critical obstacle, as it can hinder the practical deployment of HE in real-world applications where speed and efficiency are paramount. As the field continues to evolve, finding ways to mitigate these performance bottlenecks will be essential for the broader adoption of HE-based solutions in secure AI model training. Moreover, despite ongoing research, there remains a gap in establishing theoretical guarantees for the convergence of some of these models. This lack of formal proof makes it challenging to definitively validate their effectiveness, particularly as they scale to more complex and varied datasets.

In response to these challenges, our study aims to advance the field by demonstrating the effectiveness of combining a specific HE scheme, the CKKS scheme, with AI models across various scenarios. Our objectives are threefold: (1) to minimize the time and space complexity of these algorithms through the use of matrix transformations, (2) to mathematically prove the convergence of AI algorithms within the encrypted domain, and (3) to validate these theoretical proofs through rigorous experiments on two different real-world datasets.

III. Preliminaries

A. Homomorphic Encryption

Homomorphic encryption was first proposed by Craig Gentry in 2009 [28], allowing operations to be performed on encrypted data. Let's denote the encryption function as $\mathsf{Enc}(\cdot)$ and the decryption function as $\mathsf{Dec}(\cdot)$. For a plaintext message m, the encryption results in a ciphertext $c = \mathsf{Enc}(m)$. A key feature of HE is that it supports specific operations on ciphertexts which correspond to operations on the plaintexts. For example, if we can perform addition on the encrypted data, it will correspond to addition on the plaintext data. Similarly, multiplication on ciphertexts corresponds to multiplication on plaintexts. This means that computations can be carried out on encrypted data without decrypting it first.

The CKKS algorithm [7] is a specific HE scheme designed for approximate arithmetic on real numbers. In CKKS, a plaintext message m (which is a vector of real numbers) is encrypted to produce ciphertext $c = \mathsf{Enc}(m)$. The decryption process approximately recovers the original message, $m \approx \mathsf{Dec}(c)$.

CKKS supports both addition and multiplication of encrypted data. For plaintext vectors m_1 and m_2, with their corresponding ciphertexts $c_1 = \mathsf{Enc}(m_1)$ and $c_2 = \mathsf{Enc}(m_2)$, the following operations are supported:

- Addition:

$$\mathsf{Dec}(c_1 + c_2) \approx m_1 + m_2 \tag{1}$$

- Multiplication:

$$\mathsf{Dec}(c_1 \cdot c_2) \approx m_1 \cdot m_2 \tag{2}$$

CKKS uses polynomial approximations to represent real numbers, allowing efficient operations. It introduces a scaling factor to maintain precision. Specifically, for a plaintext message m scaled by a factor Δ, the encrypted message is $c = \mathsf{Enc}(m \cdot \Delta)$. During homomorphic operations, the ciphertexts are scaled accordingly, and after decryption, the resulting plaintext is divided by the scaling factor to retrieve the approximate result. For example, the product of two plaintext messages is approximately recovered by dividing the decrypted ciphertext by the scaling factor:

$$m_1 \cdot m_2 \approx \frac{\mathsf{Dec}(c_1 \cdot c_2)}{\Delta} \tag{3}$$

where Δ is typically chosen from 2^{40} to 2^{60}. This range is selected to balance precision and efficiency. A larger Δ improves the precision of the approximate equality by reducing the relative error introduced during encryption and decryption. However, it also increases computational complexity and storage requirements.

B. Neural Networks

Neural networks (NNs), conceived by Warren McCullough and Walter Pitts in 1943 [29], emulate the organization and functioning of the human brain, serving as a foundational pillar of machine learning. Composed of interconnected nodes (neurons), these networks analyze input data to discern patterns crucial for predictions and data classification across diverse domains like image recognition, natural language processing, and time series forecasting. A typical NN structure comprises an input layer, one or more hidden layers, and an output layer, with neurons within each layer processing inputs through weighted sums, biases, and activation functions. Through iterative training, where weights and biases are adjusted based on prediction errors, NNs optimize their predictive capabilities, embodying a dynamic tool for data analysis and prediction in numerous applications.

1) Feedforward Neural Network: In a feedforward neural network, information moves in one direction: from the input layer, through the hidden layers, to the output layer. The output of each neuron is given by $z_i = \sum_{j=1}^{n} \theta_{ij} x_j + b_i$, where z_i is the input to the activation function of the i-th neuron, θ_{ij} is the weight between the j-th input and the i-th neuron, x_j is the j-th input, and b_i is the bias of the i-th neuron.

The activation function σ applies non-linearity to the neuron's output: $a_i = \sigma(z_i)$.

2) Backpropagation: During training, the neural network uses backpropagation to minimize the error between predicted and actual outputs. The error (loss) is calculated using a loss function L, such as mean squared error (MSE) for regression tasks:

$$L(y, \hat{y}) = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \tag{4}$$

where y_i is the actual value and $\hat{y}_i$ is the predicted value.

The network updates its weights θ_{ij} and biases b_i by computing the gradients of the loss function with respect to each weight and bias. The updates are performed using an optimization algorithm:

$$\theta_{ij} \leftarrow \theta_{ij} - \alpha \frac{\partial L}{\partial \theta_{ij}} \tag{5}$$

and

$$b_i \leftarrow b_i - \alpha \frac{\partial L}{\partial b_i} \tag{6}$$

where α is the learning rate, controlling the size of the update steps.

Through multiple iterations of forward passes and backpropagation, the neural network learns the optimal weights and biases that minimize the loss function, enabling it to make accurate predictions on new dataset.

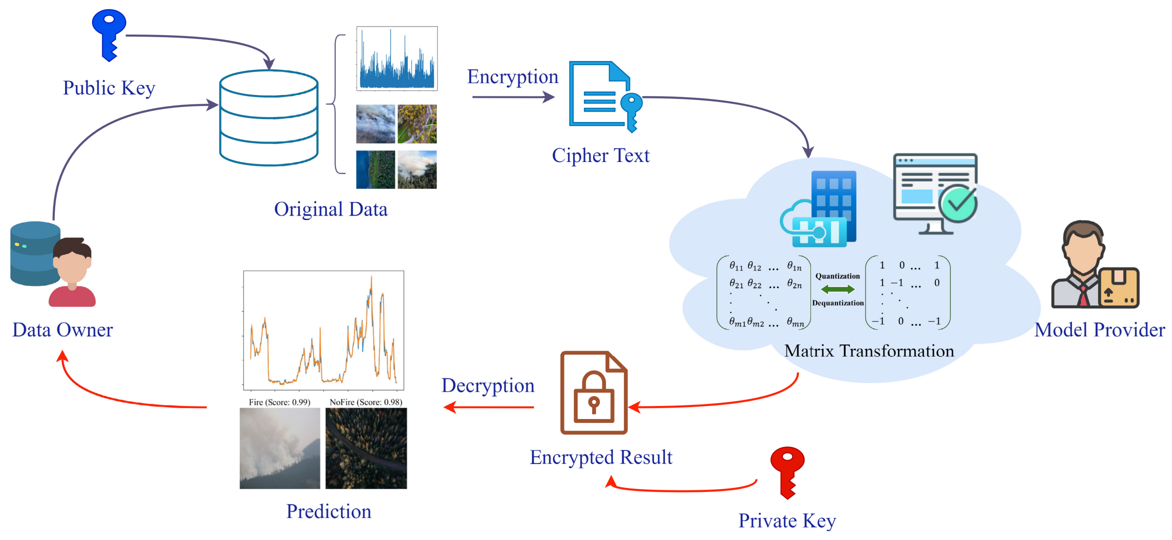

Fig. 1. Detailed architecture of the proposed system.

IV. Proposed Secure AI Models

A. Matrix Transformer

In examining a small example of both algorithms (CKKS and NNs), we observe the following: the time complexity of a neural network is primarily influenced by factors such as the number of layers, the quantity of neurons within each layer, and the computational tasks conducted during both the forward and backward passes. For the forward pass, the complexity for one layer is $\mathcal{O}(n \cdot m)$, where n is the number of inputs and m is the number of neurons in the layer. For a network with L layers, the overall complexity of the forward pass is $\mathcal{O}\left(\sum_{i=1}^{L} n_i \cdot m_i\right)$. The backward pass, which involves backpropagation, has a similar complexity. Hence, the total time complexity for training a neural network over E epochs is $\mathcal{O}\left(E \cdot \sum_{i=1}^{L} n_i \cdot m_i\right)$. On the other hand, the CKKS scheme involves polynomial arithmetic and FFT operations. The encryption and decryption processes have a time complexity of $\mathcal{O}(N \log N)$, where N is the degree of the polynomial. Homomorphic addition is efficient with $\mathcal{O}(N)$ complexity, while homomorphic multiplication, involving polynomial multiplication and scaling, has a complexity of $\mathcal{O}(N \log N)$. Additionally, the main factor leading to this is the matrix multiplication operations in the algorithms. Therefore, we propose transforming real-number matrices into matrices whose coefficients only include $\{-1, 0, 1\}$, whereby matrix multiplications are simplified into coefficient-wise additions.

Let us consider two matrices ($A(n \cdot m)$ and $B(m \cdot p)$). In the typical case (FP16, commonly used by programming languages like Python or C), we need to apply matrix multiplication formulas, resulting in a matrix C with elements $C_{i,j}$ calculated by (7):

$$C_{ij} = \sum_{k=1}^{m} a_{ik} \cdot b_{kj} \tag{7}$$

where $1 \le i \le n, 1 \le j \le p$. Therefore, the time complexity of the matrix multiplication operation would be $\mathcal{O}(n \cdot m \cdot p)$.

Algorithm 1 Matrix Transformer

```
Input: Weight matrix $\theta$ of size $n \cdot m$
Output: Quantized weight matrix $\theta_f$ and dequantized
        weight matrix $\theta_{\text{approx}}$
1  Compute average absolute value $\gamma = \frac{1}{nm}\sum_{ij}|\theta_{ij}|$
2  /* Quantization Function */
3  for $i \leftarrow 1$ to $n$ do
4      for $j \leftarrow 1$ to $m$ do
5          $\theta_f[i,j] \leftarrow \text{RoundClip}\left(\frac{\theta[i,j]}{\gamma+\epsilon}, -1, 1\right)$
6      end for
7  end for
8  /* Dequantization Function */
9  for $i \leftarrow 1$ to $n$ do
10     for $j \leftarrow 1$ to $m$ do
11         $\theta_{\text{approx}}[i,j] \leftarrow \theta_f[i,j] \cdot (\gamma + \epsilon)$
12     end for
13 end for
14 return $\theta_f$, $\theta_{\text{approx}}$
```

In this paper, we introduce a novel approach designed to improve the efficiency of matrix multiplication in terms of both time and space utilization, as outlined in Algorithm 1 with a small positive error ϵ to avoid division by zero. Our method entails converting matrix weights into a $\{-1, 0, 1\}$ format using a quantization function, similar to the technique proposed in [21]. Subsequently, the multiplication operations involving weighted matrices are simplified into polynomial additions, leading to a significant reduction in computational overhead, resulting in a notable reduction in computational overhead and time complexity is reduced to $\mathcal{O}(n \cdot m)$.

B. Proposed Secure Neural Networks

The proposed system is fundamentally based on the integration of HE and AI, as detailed in Algorithm 2, with its structural visualization depicted in Fig. 1. This cryptographic technique enables arithmetic operations on ciphertexts,

Algorithm 2 Proposed Training Process

```
Input: Training data $D$, number of iterations $T$, number
       of layers $L$, number of heads $H$, sequence length
       $S$, dropout rate, learning rate $\alpha$
Output: Trained model weights $\theta_{enc}$
1  Initialize training iterations $T$, number of layers $L$,
   number of heads $H$, original data $D$, sequence length $S$,
   dropout rate, learning rate $\alpha$.
2  Initialize model weights $\theta$ (including embedding matrix
   $\theta_e$ and transformer weights).
3  /* Encrypt model weights and data */
4  $\theta_{enc} \leftarrow \mathsf{Enc}(\theta)$
5  $D_{enc} \leftarrow \mathsf{Enc}(D)$
6  /* Start training */
7  for $i$ = 1 to $T$ do
8      for each batch $X_{enc}$ in encrypted data $D_{enc}$ do
9          $E_{enc} \leftarrow \mathsf{HMult}(\theta_{enc}, \theta_e)$
10         $E_{enc} \leftarrow \mathsf{encoding}(E_{enc})$
11         for $l$ = 1 to $L$ do
12             for $h$ = 1 to $H$ do
13                 $Q, K, V \leftarrow \mathsf{split\_heads}(E_{enc})$
14                 $Q, K, V \leftarrow \mathsf{binarize}(Q, K, V)$
15                 $A_{enc} \leftarrow$ Approximate Active Function
16                 $A_{enc} \leftarrow \mathsf{dropout}(A_{enc})$
17                 $O_{h,enc} \leftarrow \mathsf{HMult}(A_{enc}, V)$
18             end for
19             $O_{enc} \leftarrow \mathsf{concat}([O_{1,enc}, \ldots, O_{H,enc}])$
20             $O_{enc} \leftarrow \mathsf{linear\_transformation}(O_{enc})$
21             $E_{enc} \leftarrow \mathsf{add\_and\_norm}(E_{enc}, O_{enc})$
22             $F_{enc} \leftarrow \mathsf{feed\_forward}(E_{enc})$
23             $F_{enc} \leftarrow \mathsf{binarize}(F_{enc})$
24             $E_{enc} \leftarrow \mathsf{add\_and\_norm}(E_{enc}, F_{enc})$
25         end for
26         /* Output */
27         $output_{enc} \leftarrow \mathsf{linear\_transformation}(E_{enc})$
28         $predicted \leftarrow output_{enc}$
29         $loss \leftarrow \mathsf{compute\_loss}(predicted, actual)$
30         $gradients \leftarrow \mathsf{backpropagate}(loss)$
31         /* Update weights in encrypted
              domain */
32         $\theta_{enc} \leftarrow \theta_{enc} - \alpha * gradients$
33     end for
34 end for
35 return $\theta_{enc}$
```

generating encrypted outcomes that align with results obtained from operations on plaintexts once decrypted. Such a property guarantees the confidentiality of data throughout the training process. Furthermore, the utilization of a matrix transformer aids in minimizing computational and memory requirements by binarizing specific operations.

1) Initialization: The training process begins with the initialization of the model parameters. This includes setting the number of training iterations T, the number of transformer layers L, the number of attention heads H, the original data D, the sequence length S, the dropout rate, and the learning rate α. The model weights, including the embedding matrix θ_e and transformer weights, are initialized and subsequently encrypted using CKKS scheme, resulting in encrypted model weights θ_{enc}. Similarly, the training data D and labels Y are encrypted to produce D_{enc} and Y_{enc}.

2) Training Process: In the proposed training process, the input batch X_{enc} undergoes an embedding operation where it is multiplied with the encrypted embedding matrix θ_e. This HE-based multiplication preserves the confidentiality of the input data while allowing the model to learn representations. Additionally, positional encoding is incorporated into the embedded input to capture sequence information, which is crucial for tasks involving sequential data.

Each transformer layer within the model architecture consists of a multi-head attention mechanism followed by a feed-forward network. In the multi-head attention component, computations are performed on the query (Q), key (K), and value (V) matrices, which are binarized to enhance computational efficiency. Attention scores are calculated using a ninth-degree polynomial approximation of the activation function, which is clearly defined in our previous research [27], ensuring compatibility with HE. Dropout is applied to the attention scores, maintaining the security of the mechanism even under encryption constraints. The resulting scores are then utilized to compute the output of each attention head, which is concatenated and subjected to linear transformation. Addition and normalization operations are employed to stabilize the learning process. Subsequently, the feed-forward network further processes the output, followed by binarization and another round of addition and normalization operations operation, enhancing the model's representation capabilities.

The final output of the model is generated through a linear transformation of the processed embeddings. To assess the model's performance, the output is decrypted to compare it with the actual values. In the encrypted domain, a polynomial approximation serves as the activation function, ensuring compatibility with HE. The loss is computed by comparing the decrypted predictions with the actual labels. This loss guides the backpropagation process, facilitating the refinement of model parameters. Furthermore, during backpropagation, gradients are computed on the encrypted data. These gradients are then used to update the encrypted model weights, ensuring that the weights remain confidential and secure throughout the training process. The weight update rule is defined as $\theta_{enc} \leftarrow \theta_{enc} - \alpha \cdot \text{gradients}$, where α represents the learning rate.

C. Model Convergence Proof

In this paper, we use the ADAM algorithm [30] as the optimizer for the proposed models. ADAM maintains two moving averages of the gradients: the first moment (mean) and the second moment (uncentered variance). The update rules for ADAM are as follows:

- Compute gradients:

$$g_t = \nabla L(\theta_t) \tag{8}$$

- Update biased first-moment estimate:

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t \tag{9}$$

- Update biased second-moment estimate:

$$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2 \tag{10}$$

- Compute bias-corrected first-moment estimate:

$$\hat{m}_t = \frac{m_t}{1-\beta_1^t} \tag{11}$$

- Compute bias-corrected second-moment estimate:

$$\hat{v}_t = \frac{v_t}{1-\beta_2^t} \tag{12}$$

- Update parameters:

$$\theta_{t+1} = \theta_t - \alpha \frac{\hat{m}_t}{\sqrt{\hat{v}_t}} \tag{13}$$

Now, to prove the model convergence with the ADAM optimizer, we first introduce the convex function:

Lemma 1 (Convex Function): Let $f : \mathbb{R}^n \to \mathbb{R}$ be a differentiable function. If and only if f satisfies the inequality:

$$f(y) \geq f(x) + \nabla f(x)^T (y-x)$$

for all $x, y \in \mathbb{R}^n$, then f is convex.

Proof: For the forward implication, if f is convex, then we have the definition [31]:

$$\begin{aligned} f(\lambda y + (1-\lambda)x) &\leq \lambda f(y) + (1-\lambda) f(x) \\ \Leftrightarrow g(\lambda) &= f(\lambda y + (1-\lambda)x) \\ &\quad - \lambda f(y) - (1-\lambda) f(x) \leq 0 \end{aligned}$$

for all x, y, and $\lambda \in [0, 1]$. Therefore,

$$\begin{aligned} g'(0) &= \lim_{\lambda \to 0^+} \frac{g(\lambda) - g(0)}{\lambda} \leq 0 \\ &\Leftrightarrow \nabla^T f(x)(y-x) - f(y) + f(x) \leq 0 \\ &\Leftrightarrow f(y) \geq \nabla^T f(x)(y-x) + f(x). \end{aligned}$$

For the backward implication, if

$$f(y) \geq \nabla^T f(x)(y-x) + f(x)$$

then we now consider $z = \lambda x + (1-\lambda) y$. We have 2 following inequalities:

$$\begin{aligned} &f(y) \geq \nabla^T f(z)(y-z) + f(z) \\ \Leftrightarrow\ &f(y) \geq \lambda \nabla^T f(z)(y-x) + f(\lambda x + (1-\lambda) y) \end{aligned}$$

and

$$\begin{aligned} &f(x) \geq \nabla^T f(z)(x-z) + f(z) \\ \Leftrightarrow\ &f(x) \geq (1-\lambda) \nabla^T f(z)(x-y) + f(\lambda x + (1-\lambda) y) \end{aligned}$$

Multiplying the first inequality by $1-\lambda$, the second one by λ, and adding them we get:

$$(1-\lambda) f(y) + \lambda f(x) \geq f(\lambda x + (1-\lambda) y)$$

which shows that f is convex. □

We then recap the loss function definition:

Definition 1: Let $\theta^* := \arg\min_{\theta \in \chi} \sum_{t=1}^{T} L_t(\theta)$ with χ as the set of θ which will occur in the ADAM algorithm. The loss is then defined as:

$$\mathcal{L}(T) := \sum_{t=1}^{T} (L_t(\theta_t) - L_t(\theta^*))$$

Now, we need to prove that the loss converges with the timestamp T, and once this proof is completed, our task will be accomplished.

Theorem 1: The loss function can be approximated in the HE domain as:

$$\begin{aligned} \mathcal{L}(T) \leq\ &\frac{h H_\infty^2 \sqrt{T \hat{v}_{T,i}}}{2\alpha(1-\beta_1)} + \frac{h \beta_1 H_\infty^2 G_\infty}{2\alpha(1-\beta_1)} \\ &+ \frac{\alpha h \beta_1}{2(1-\beta_1)} \sum_{t=1}^{T} \frac{\hat{m}_{t-1,i}^2}{\sqrt{(t-1)\hat{v}_{t-1,i}}} \\ &+ \frac{\alpha}{2(1-\beta_1)} \sum_{t=1}^{T} \frac{\hat{m}_{t,i}^2}{\sqrt{t \hat{v}_{t,i}}} \end{aligned}$$

where, g_t be bounded with $\|g_t\|_2 \leq G$ and $\|g_t\|_\infty \leq G_\infty$, $\alpha \geq \alpha_t$, $\|\theta_n - \theta_m\|_2 \leq H$ and $\|\theta_n - \theta_m\|_\infty \leq H_\infty$ with $n, m \in \{1, \ldots, T\}$.

Proof: From (9), (11), and (13), we can express the update function as:

$$\begin{aligned} \theta_{t+1} &= \theta_t - \alpha_t \frac{\hat{m}_t}{\sqrt{\hat{v}_t}} \\ &= \theta_t - \alpha_t \left(\frac{\beta_{1,t} m_{t-1} + (1-\beta_{1,t}) g_t}{(1-\beta_1^t)\sqrt{\hat{v}_t}} \right) \end{aligned}$$

The convex differentiable of $L(\theta)$ can be written follow lemma 1:

$$\begin{aligned} L_t(\theta^*) &\geq L_t(\theta_t) + \mathbf{g}_t^T (\theta^* - \theta_t) \\ &\Leftrightarrow L_t(\theta_t) - L_t(\theta^*) \leq \mathbf{g}_t^T (\theta_t - \theta^*) \end{aligned}$$

Let us consider the i-th coefficient of weight set in $\mathbb{R}^h$:

$$\begin{aligned} &\theta_{t+1,i} - \theta_{,i}^* = \theta_{t,i} - \theta_{,i}^* - \alpha_t \frac{\hat{m}_{t,i}}{\sqrt{\hat{v}_{t,i}}} \\ \Leftrightarrow\ &(\theta_{t+1,i} - \theta_{,i}^*)^2 = (\theta_{t,i} - \theta_{,i}^*)^2 - 2(\theta_{t,i} - \theta_{,i}^*)\alpha_t \frac{\hat{m}_{t,i}}{\sqrt{\hat{v}_{t,i}}} \\ &\qquad + \alpha_t^2 \left(\frac{\hat{m}_{t,i}}{\sqrt{\hat{v}_{t,i}}}\right)^2 \end{aligned}$$

Multiply both sides by $\frac{(1-\beta_1^t)\sqrt{\hat{v}_{t,i}}}{2\alpha_t(1-\beta_{1,t})}$, we have:

$$\begin{aligned} \frac{(1-\beta_1^t)\sqrt{\hat{v}_{t,i}}}{2\alpha_t(1-\beta_{1,t})} (\theta_{t+1,i} - \theta_{,i}^*)^2 = \frac{(1-\beta_1^t)\sqrt{\hat{v}_{t,i}}}{2\alpha_t(1-\beta_{1,t})} \Bigg(&(\theta_{t,i} - \theta_{,i}^*)^2 \\ &- 2(\theta_{t,i} - \theta_{,i}^*)\alpha_t \frac{\hat{m}_{t,i}}{\sqrt{\hat{v}_{t,i}}} \\ &+ \alpha_t^2 \left(\frac{\hat{m}_{t,i}}{\sqrt{\hat{v}_{t,i}}}\right)^2 \Bigg) \end{aligned}$$

From this equation, (9), (11), and simplify it, we have:

$$g_{t,i}(\theta_{t,i}-\theta^*_{,i}) = \underbrace{\frac{(1-\beta_1^t)\sqrt{\hat{v}_{t,i}}}{2\alpha_t(1-\beta_{1,t})}((\theta_{t,i}-\theta^*_{,i})^2-(\theta_{t+1,i}-\theta^*_{,i})^2)}_{(a)} + \underbrace{\frac{-\beta_{1,t}}{(1-\beta_{1,t})}m_{t-1,i}(\theta_{t,i}-\theta^*_{,i})}_{(b)} + \underbrace{\frac{\alpha_t(1-\beta_1^t)\sqrt{\hat{v_{t,i}}}}{2(1-\beta_{1,t})}\left(\frac{\hat{m}_{t,i}}{\sqrt{\hat{v}_{t,i}}}\right)^2}_{(c)}$$

For each term (a), (b), and (c), we consider the coefficient in the HE domain (H) and the time constrain T:
For (a), the chosen $\beta_1 \in (0,1)$ then $(1-\beta_1^t) \le 1 (\forall t \in T)$:

$$\begin{aligned}
&\frac{(1-\beta_1^t)\sqrt{\hat{v}_{t,i}}}{2\alpha_t(1-\beta_{1,t})}((\theta_{t,i}-\theta^*_{,i})^2-(\theta_{t+1,i}-\theta^*_{,i})^2)\\
&\le \frac{\sqrt{\hat{v}_{t,i}}}{2\alpha_t(1-\beta_{1,t})}((\theta_{t,i}-\theta^*_{,i})^2-(\theta_{t+1,i}-\theta^*_{,i})^2)\\
&\le \sum_{i=1}^{h}\sum_{t=1}^{T}\frac{\sqrt{\hat{v}_{t,i}}}{2\alpha_t(1-\beta_1)}((\theta_{t,i}-\theta^*_{,i})^2-(\theta_{t+1,i}-\theta^*_{,i})^2)\\
&= \sum_{i=1}^{h}\frac{(\theta_{t,i}-\theta^*_{,i})^2\sqrt{\hat{v}_{1,i}}}{2\alpha_1(1-\beta_1)} + \sum_{i=1}^{h}\sum_{t=2}^{T}\frac{(\theta_{1,i}-\theta^*_{,i})^2\sqrt{\hat{v}_{t,i}}}{2\alpha_t(1-\beta_1)}\\
&\quad - \sum_{i=1}^{h}\frac{(\theta_{T+1,i}-\theta^*_{,i})^2\sqrt{\hat{v}_{T,i}}}{2\alpha_T(1-\beta_1)}\\
&\quad - \sum_{i=1}^{h}\sum_{t=2}^{T}\frac{(\theta_{t,i}-\theta^*_{,i})^2\sqrt{\hat{v}_{t-1,i}}}{2\alpha_t(1-\beta_1)}\\
&= \sum_{i=1}^{h}\frac{\sqrt{\hat{v}_{1,i}}}{2\alpha_1(1-\beta_1)}\underbrace{(\theta_{1,i}-\theta^*_{,i})^2}_{\le H_\infty^2} - \underbrace{\sum_{i=1}^{h}\frac{(\theta_{T+1,i}-\theta^*_{,i})^2\sqrt{\hat{v}_{T,i}}}{2\alpha_T(1-\beta_1)}}_{\ge 0}\\
&\quad + \sum_{i=1}^{h}\sum_{t=2}^{T}\frac{1}{2(1-\beta_1)}\left(\frac{\sqrt{\hat{v}_{t,i}}}{\alpha_t}-\frac{\sqrt{\hat{v}_{t-1,i}}}{\alpha_{t-1}}\right)\underbrace{(\theta_{t,i}-\theta_{,i})^2}_{\le H_\infty^2}\\
&\le \frac{H_\infty^2}{2\alpha(1-\beta_1)}\sum_{i=1}^{h}\sqrt{T\hat{v}_{T,i}} = \frac{hH_\infty^2\sqrt{T\hat{v}_{T,i}}}{2\alpha(1-\beta_1)} \qquad ①
\end{aligned}$$

For (b), we multiply with $\sqrt{\frac{(\alpha_{t-1})\sqrt{\hat{v}_{t-1}}}{(\alpha_{t-1})\sqrt{\hat{v}_{t-1}}}}$:

$$\begin{aligned}
&\frac{\beta_{1,t}}{(1-\beta_{1,t})}m_{t-1,i}(\theta^*_{,i}-\theta_{t,i})\sqrt{\frac{(\alpha_{t-1})\sqrt{\hat{v}_{t-1}}}{(\alpha_{t-1})\sqrt{\hat{v}_{t-1}}}}\\
&= \frac{\beta_{1,t}}{(1-\beta_{1,t})}\left[\left(\frac{\hat{v}_{t-1,i}^{1/4}}{\alpha_{t-1}^{1/2}}(\theta^*_{,i}-\theta_{t,i})\right)\left(\frac{\alpha_{t-1}^{1/2}m_{t-1,i}}{\hat{v}_{t-1,i}^{1/4}}\right)\right]
\end{aligned}$$

Using Cauchy-Schwarz inequality for two numbers, we have:

$$(b) \le \underbrace{\frac{\beta_{1,t}}{(1-\beta_{1,t})}}_{\le(\beta_1)/(1-\beta_1)}\left[\frac{\sqrt{\hat{v}_{t-1,i}}(\theta^*_{,i}-\theta_{t,i})^2}{2\alpha_{t-1}} + \frac{\alpha_{t-1}m^2_{t-1,i}}{2\sqrt{\hat{v}_{t-1,i}}}\right]$$

Add both term of the inequality with (c), and apply the constrain $(1-\beta_1^t) \le 1$:

$$\begin{aligned}
(b)+(c) &\le \frac{\beta_{1,t}}{(1-\beta_{1,t})}\left[\frac{\sqrt{\hat{v}_{t-1,i}}(\theta^*_{,i}-\theta_{t,i})^2}{2\alpha_{t-1}} + \frac{\alpha_{t-1}m^2_{t-1,i}}{2\sqrt{\hat{v}_{t-1,i}}}\right]\\
&\quad + \frac{\alpha_t\hat{m}^2_{t,i}}{2(1-\beta_{1,t})\sqrt{\hat{v}_{t,i}}}\\
&\le \underbrace{\frac{\beta_1\sqrt{\hat{v}_{t-1,i}}(\theta^*_{,i}-\theta_{t,i})^2}{2\alpha(1-\beta_1)}}_{(*)} + \underbrace{\frac{\beta_1\alpha_{t-1}m^2_{t-1,i}}{2(1-\beta_1)\sqrt{\hat{v}_{t-1,i}}}}_{(**)}\\
&\quad + \underbrace{\frac{\alpha_t\hat{m}^2_{t,i}}{2(1-\beta_{1,t})\sqrt{\hat{v}_{t,i}}}}_{(***)}
\end{aligned}$$

For $(*)$, we can conduct the following inequality:

$$\sum_{i=1}^{h}\sum_{t=1}^{T}\frac{\beta_1\sqrt{\hat{v}_{t-1,i}}(\theta^*_{,i}-\theta_{t,i})^2}{2\alpha(1-\beta_1)} \le \frac{\beta_1 hG_\infty H_\infty^2}{2\alpha(1-\beta_1)}$$

For $(**)$, we can deduce the inequality by using (9) and (11):

$$\begin{aligned}
(**) &= \frac{\beta_1\alpha}{2(1-\beta_1)}\sum_{i=1}^{h}\sum_{t=1}^{T}\frac{\hat{m}^2_{t-1,i}}{\sqrt{(t-1)\hat{v}_{t-1,i}}}\\
&\le \frac{\alpha h\beta_1}{2(1-\beta_1)}\sum_{t=1}^{T}\frac{\hat{m}^2_{t-1,i}}{\sqrt{(t-1)\hat{v}_{t-1,i}}}
\end{aligned}$$

Similar to $(**)$, for $(***)$, we can infer the following inequality:

$$(***) \le \frac{\alpha}{2(1-\beta_1)}\sum_{t=1}^{T}\frac{\hat{m}^2_{t,i}}{\sqrt{t\hat{v}_{t,i}}}$$

Then, we can combine all 3 terms $(*)$, $(**)$, and $(***)$ to get the inequality for (b) and (c):

$$\begin{aligned}
(b)+(c) &\le \frac{\beta_1 hG_\infty H_\infty^2}{2\alpha(1-\beta_1)} + \frac{\alpha h\beta_1}{2(1-\beta_1)}\sum_{t=1}^{T}\frac{\hat{m}^2_{t-1,i}}{\sqrt{(t-1)\hat{v}_{t-1,i}}}\\
&\quad + \frac{\alpha}{2(1-\beta_1)}\sum_{t=1}^{T}\frac{\hat{m}^2_{t,i}}{\sqrt{t\hat{v}_{t,i}}} \qquad ②
\end{aligned}$$

From definition 1 and the convex function of $L(\theta)$ with the constrain of time: $t \in \{1,\ldots,T\}$ and the gradient of the encrypted model in the HE domain: $i \in \{1,\ldots,h\}$, we have:

$$\begin{aligned}
\mathcal{L}(T) &= \sum_{t=1}^{T}(L_t(\theta_t)-L_t(\theta^*))\\
&\le \sum_{t=1}^{T}g^T(\theta_t-\theta^*) = \sum_{t=1}^{T}\sum_{t=1}^{h}g_{t,i}(\theta_{t,i}-\theta^*_{,i}) \qquad ③
\end{aligned}$$

From $\textcircled{1}$, $\textcircled{2}$, and $\textcircled{3}$, theorem 1 is proven:

$$\begin{aligned}\mathcal{L}(T) \le{}& \frac{hH_\infty^2\sqrt{T\hat{v}_{T,i}}}{2\alpha(1-\beta_1)} + \frac{h\beta_1 H_\infty^2 G_\infty}{2\alpha(1-\beta_1)} \\ &+ \frac{\alpha h\beta_1}{2(1-\beta_1)}\sum_{t=1}^{T} \frac{\hat{m}_{t-1,i}^2}{\sqrt{(t-1)\hat{v}_{t-1,i}}} \\ &+ \frac{\alpha}{2(1-\beta_1)}\sum_{t=1}^{T}\frac{\hat{m}_{t,i}^2}{\sqrt{t\hat{v}_{t,i}}}\end{aligned}$$

□

With the mentioned constraints, we need to prove that the loss function converges within the time domain. Since $T > 0$, we have the following equation:

$$\begin{aligned}\lim_{T->\infty}\frac{\mathcal{L}(T)}{T} \le{}& \lim_{T->\infty}\Bigg[\frac{hH_\infty^2\sqrt{\hat{v}_{T,i}}}{2\sqrt{T}\alpha(1-\beta_1)} + \frac{h\beta_1 H_\infty^2 G_\infty}{2T\alpha(1-\beta_1)} \\ &+ \frac{\alpha h\beta_1}{2T(1-\beta_1)}\sum_{t=1}^{T} \frac{\hat{m}_{t-1,i}^2}{\sqrt{(t-1)\hat{v}_{t-1,i}}} \\ &+ \frac{\alpha}{2T(1-\beta_1)}\sum_{t=1}^{T}\frac{\hat{m}_{t,i}^2}{\sqrt{t\hat{v}_{t,i}}}\Bigg]\end{aligned}$$

As the denominators of each term grow faster than their respective numerators, and under the assumption that the sums in the third and fourth terms remain bounded, all terms approach zero, leading to the convergence of the loss function to zero:

$$\lim_{T->\infty}\frac{\mathcal{L}(T)}{T} = 0 \tag{14}$$

V. Result and Discussion

A. Data Preparation and Evaluation Metrics

1) Data Preparation: In this study, we conducted a comprehensive evaluation of a proposed transformation methodology by applying it to two significant AI problems: time series forecasting for air pollution and image recognition for forest fire detection. The air pollution dataset [32] was collected hourly over five years at the US embassy in Beijing, China. It includes meteorological data (temperature, humidity, wind speed, atmospheric pressure), pollutant concentrations (PM2.5), and temporal information (date and time stamps). The objective was to assess the impact of these features on air pollution levels. We divided the dataset into 80% for training and 20% for testing.

The forest fire dataset [33] comprised 2388 images labeled “fire” and “no fire”, collected from real forests using drones in 2020. These images varied in size and resolution, capturing diverse environmental conditions and maintaining high-resolution details necessary for accurate fire detection. To process this dataset, we employed a vision transformer (ViT) model, known for its effectiveness in image recognition tasks. We also integrated CKKS to ensure data privacy, which is crucial for real-world applications. This combination of ViT and CKKS provides a robust foundation for developing advanced forest fire detection systems, enhancing model accuracy while protecting sensitive data.

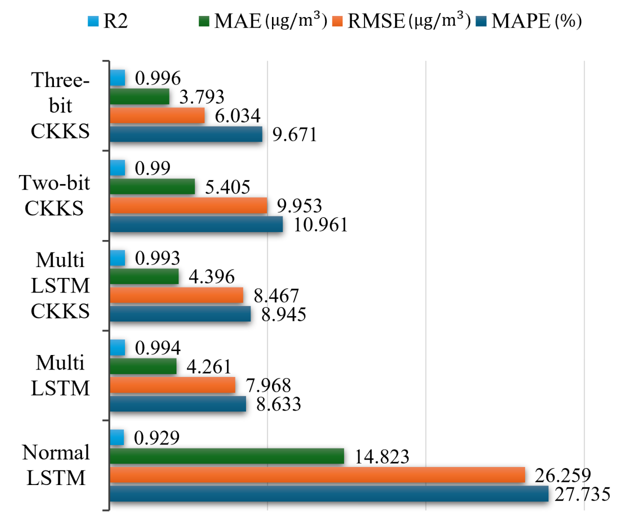

Fig. 2. Performance evaluation metrics for five different air pollution forecasting models.

2) Evaluation Metrics: We use 4 metrics to assess air pollution forecasting models. Mean absolute error (MAE) and root mean square error (RMSE) measure the difference between predicted and actual pollution levels, while mean absolute percentage error (MAPE) reveals the model’s bias. R-squared (R2) quantifies the variance explained by the model [34]. For forest fire detection, precision, recall, F1 score, and accuracy score [35] evaluate model performance, with precision measuring correct predictions, recall identifying actual fire instances, and F1 score balancing both. Additional ROC curve and confusion matrix assess detection system effectiveness and spatial agreement.

B. Air Pollution Forecasting

We explore forecasting air pollution by comparing outcomes across five distinct models. The initial model employs an LSTM framework with 64 units, predicting solely based on past air pollution data points, disregarding other features, and serving as the baseline assessment. Following is a multivariate LSTM model, featuring 128 units, which accounts for all specified dataset features. The third model integrates the multivariate LSTM with the CKKS algorithm, configured with a modulus degree of 8192 and 160 coefficient bits. The fourth model, a novel proposal, evolves from the multivariate LSTM and CKKS model merged with the matrix transformer algorithm, with weight matrix quantization within the range of $\{-1, 1\}$, named two-bit CKKS. Similarly, the fifth model, now named three-bit CKKS, applies weight matrix quantization within the range of $\{-1, 0, 1\}$. Parameters selected for models 4 and 5 align respectively with those of models 2 and 3. The training epoch for these forecasting models is set to 100, and the early-stopping technique is also employed to minimize processing time.

In Fig. 2, we observe differences between utilizing features compared to using a basic LSTM network. Moreover, transitioning the network to the encrypted domain has enhanced

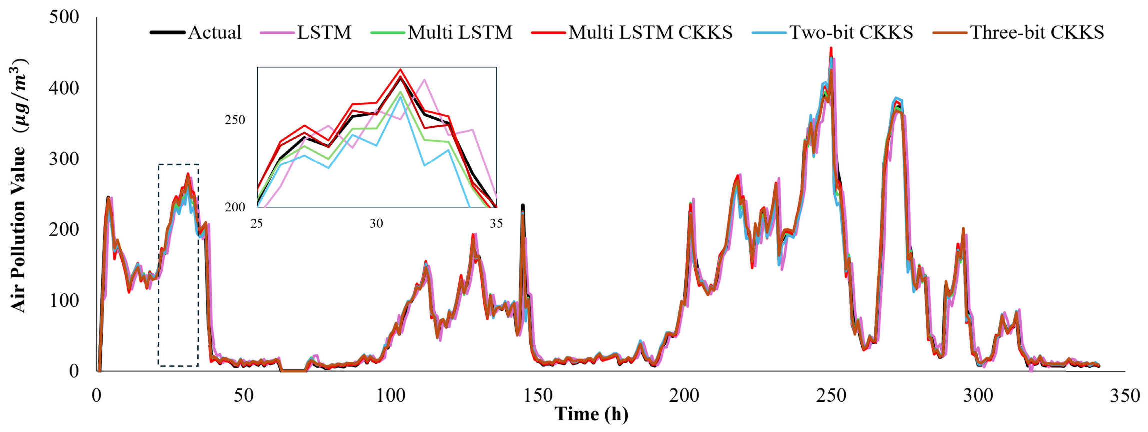

Fig. 3. Visualization of forecasting results from five different models for air pollution forecasting.

TABLE I
PERFORMANCE COMPARISON BETWEEN VARIOUS MODELS

Model Name	Precision	Recall	F1-Score	Accuracy
ViT	0.9879	0.9899	0.9889	0.9889
ViT CKKS	0.9798	0.9838	0.9818	0.9818
Two-bit CKKS	0.9819	0.9838	0.9828	0.9829
Three-bit CKKS	0.9899	0.9879	0.9889	0.9889

security for the AI model, albeit with a trade-off as accuracy is slightly diminished, specifically seen in the MAPE index increasing from 8.633% to 8.945%. Similarly, employing a matrix transformer also escalates the MAPE index to 10.961%. However, across other metrics such as R2, MAE, or RMSE, we notice minimal changes. Notably, the R2 score of the three-bit CKKS model reaches 0.996, indicating an exceptional fit of the model to the data, capturing almost all patterns and relationships within the dataset, resulting in highly precise predictions. Switching to visualization, as depicted in Fig. 3, we can clearly observe the accuracy of the proposed models. Among them, the three-bit CKKS model demonstrates remarkable adherence to the air pollution trend line.

C. Forest Fire Detection

For the forest fire detection task, we opted for the ViT model as our baseline for benchmarking against our novel composite architectures. The second model in consideration integrates ViT with CKKS encryption, featuring parameters (8192, 160). Our proposed innovations, namely two-bit CKKS and three-bit CKKS, are derived through the application of the matrix transformer algorithm outlined in Section IV. The training epoch for these detection models is fixed at 100, and the early-stopping technique is used to reduce processing time.

Upon analyzing Table I, Table III, Fig. 4, and Fig. 5, a nuanced understanding of each model's problem-solving prowess emerges. While the fundamental ViT model demonstrates superior accuracy metrics, this advantage comes at the expense of the enhanced security features offered by our proposed models. Moreover, the marginal declines in accuracy metrics such as precision, recall, F1, and overall accuracy are virtually negligible, all registering below the 1% threshold.

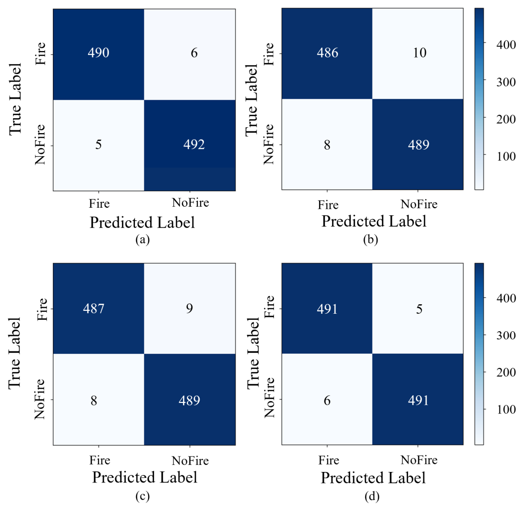

Fig. 4. Confusion matrix of (a) ViT model, (b) ViT CKKS model, (c) Two-bit CKKS model, and (d) Three-bit CKKS model.

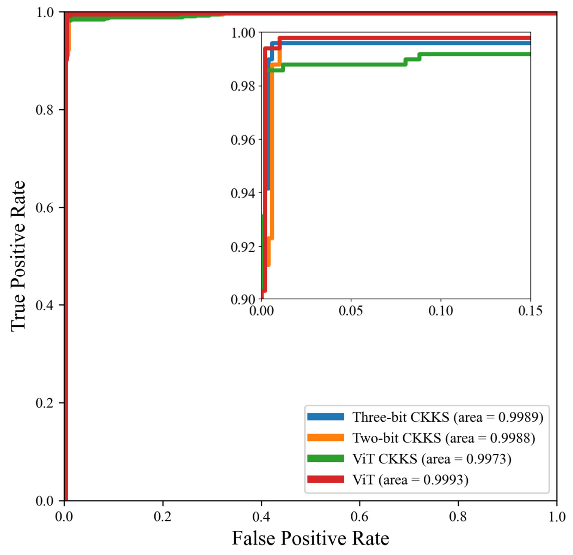

Fig. 5. Visualization of ROC curves for various models for forest fire detection.

Delving deeper, the transformative approach to the matrix not only amplifies accuracy but also yields substantial reductions in training duration and model dimensions when

TABLE II
PERFORMANCE COMPARISON BETWEEN VARIOUS WORKS

Model	Dataset	Data Dimension	Acceleration Technique	Best Accuracy (%)	Accuracy Gap (%)	Time per Epoch (s)
DCT-CryptoNets ResNet-20 [36]	CIFAR-10	16×16	Discrete Cosine Transform	91.6	1.0 - 2.5	565
LeNet-5 OpenFHE [37]	MNIST	32×32	None	95.0	-	1515.13
LeNet-5 Lancelot [37]	MNIST	32×32	Hardware Acceleration	93.0	-	63.20
ReLU approximation [27]	Audio MNIST	Signal	None	95.5	0.2 - 1	701
Two-Bit	Forest Fire	4057×3155	Matrix Transformer	98.3	<0.6	21.6
Three-Bit	Forest Fire	4057×3155	Matrix Transformer	98.9	<0.1	31.8

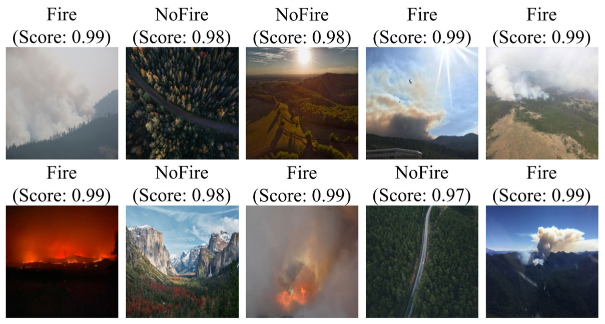

Fig. 6. Forest fire detection results of the three-bit model.

juxtaposed with the ViT CKKS algorithm. This noteworthy aspect will be elaborated upon in subsequent sections, elucidating the multifaceted benefits of our innovative approach. Furthermore, Fig. 6 illustrates the consistently high detection accuracy of the three-bit CKKS model across 10 randomly selected images.

D. Performance Comparison Between Several Works

The performance comparison in Table II provides a comprehensive analysis of how various models and techniques perform in terms of accuracy and computational efficiency, with datasets ranging from widely used benchmarks like CIFAR-10 and MNIST to more specialized domains such as forest fire detection. This comparison reveals important insights into how different acceleration techniques and encryption methods affect the trade-off between accuracy and processing time.

The DCT-CryptoNets ResNet-20 model [36] performs well on the CIFAR-10 dataset, achieving a best accuracy of 91.6%. However, this model comes with a significant computational cost, requiring 565 seconds per epoch. This highlights a notable trade-off between maintaining high accuracy and computational overhead, particularly when employing the discrete cosine transform (DCT) for encryption acceleration. While the accuracy is commendable, the relatively long time per epoch makes this approach less suitable for real-time or resource-constrained environments.

On the other hand, the LeNet-5 models (OpenFHE and Lancelot) [37], applied to the MNIST dataset, provide a striking example of how hardware acceleration can dramatically influence performance. OpenFHE's LeNet-5, which lacks any form of hardware acceleration, achieves a higher accuracy of 95.0% but at a considerable computational cost, requiring 1515.13 seconds per epoch. On the other hand, Lancelot's LeNet-5, which incorporates hardware acceleration, significantly reduces the time per epoch to just 63.20 seconds. However, this comes with a slight decrease in accuracy, dropping to 93.0%. This result highlights the efficiency gains possible with hardware acceleration, although it necessitates a small trade-off in accuracy.

The ReLU approximation model [27], applied to the audio MNIST dataset, achieves a strong accuracy of 95.5%, demonstrating its effectiveness for signal-based datasets. However, it also exhibits a high time per epoch of 701 seconds, reflecting the computational complexity associated with approximating non-linear activation functions in encrypted environments. While the model delivers competitive accuracy, its time demands may limit its applicability in scenarios requiring faster inference.

The Two-Bit and Three-Bit models, both utilizing matrix transformer acceleration and applied to the forest fire dataset, represent the highest-performing approaches in this comparison. The Two-Bit model achieves the best accuracy of 98.3%, while the Three-Bit model reaches an even higher accuracy of 98.9%. The Two-Bit model, for instance, completes an epoch in 21.6 seconds, making it the fastest model in this comparison.

A key contribution of this work is the introduction of the accuracy gap metric, which offers a new perspective on the impact of encryption algorithms on model performance. The accuracy gap measures the difference in accuracy between models using encryption techniques and their base versions without encryption. This metric is critical for assessing the trade-offs between maintaining model accuracy and incorporating encryption for data security. For example, the DCT-CryptoNets ResNet-20 model shows an accuracy gap of 1.0% to 2.5%, indicating a modest reduction in accuracy due to the encryption overhead introduced by the Discrete Cosine Transform. Despite this, the relatively small gap implies that the encryption method effectively preserves most of the model's accuracy.

TABLE III
PERFORMANCE EVALUATION OF AI MODELS IN AIR POLLUTION FORECASTING AND FOREST FIRE DETECTION

Evaluation Metric	Air Pollution Forecasting				Forest Fire Detection			
	Multi LSTM	Multi LSTM CKKS	Two-bit CKKS	Three-bit CKKS	Normal ViT	ViT CKKS	Two-bit CKKS	Three-bit CKKS
Training time per epoch (s)	0.48	121.9	1.21	2.13	13.2	541.1	21.6	31.8
Memory storage (Bytes)	12,480	1,241,061	15,033	16,055	370,159	12,281,154	241,401	489,410

In contrast, the Two-Bit and Three-Bit models, applied to the forest fire dataset, exhibit much smaller accuracy gaps of less than 0.6% and less than 0.1%, respectively. These findings demonstrate that when encryption is combined with efficient acceleration techniques such as matrix transformers, its impact on performance is minimal, allowing the models to maintain high accuracy while benefiting from encryption.

E. Training Time and Memory Storage Considerations

Table III showcases significant variations in training time and memory storage among different models for both air pollution forecasting and forest fire detection tasks. In air pollution forecasting, the Multi LSTM CKKS model notably increases training time from 0.48 seconds for the traditional Multi LSTM to 121.9 seconds. However, its memory storage requirement sharply rises from 12,480 bytes to 1,241,061 bytes, highlighting the computational overhead introduced by cryptographic techniques. Conversely, the two-bit CKKS and three-bit CKKS models exhibit reduced training times, at 1.21 seconds and 2.13 seconds, respectively, while maintaining relatively low memory storage needs of 15,033 bytes and 16,055 bytes. Similarly, in forest fire detection, integrating CKKS encryption escalates both training time and memory storage requirements. For instance, the ViT CKKS model's training time increases significantly from 13.2 seconds to 541.1 seconds, with a substantial rise in memory storage from 370,159 bytes to 12,281,154 bytes. In contrast, the two-bit CKKS and three-bit CKKS models demonstrate competitive training times of 21.6 seconds and 31.8 seconds, respectively, with modest memory storage needs of 241,401 bytes and 489,410 bytes.

Through evaluations of models' accuracy and resource consumption via simulations, we successfully demonstrate the convergence of the proposed model and assert the mathematical basis of this proof. Furthermore, we prove the effectiveness of the matrix transformation method in reducing model size. We affirm the performance of combining AI algorithms, HE techniques, and matrix transformers in two tasks: time series forecasting and image detection.

VI. CONCLUSION

This paper presents a novel approach to address the critical challenges of data security and computational efficiency in AI models by integrating HE. Through the development of two hybrid models optimized for real-world applications and the introduction of a novel matrix transformation method compatible with both HE algorithms and AI weight matrices, we offer a robust solution. Furthermore, the establishment of a mathematical proof of convergence for integrating HE with the ADAM optimization algorithm fills a significant gap in existing literature. Our evaluations on established datasets related to air pollution forecasting and wildfire detection demonstrate the effectiveness and practicality of training on encrypted data, achieving high performance while maintaining data privacy. These contributions signify a substantial advancement in enhancing the security, efficiency, and reliability of AI models in sensitive data environments, offering promising avenues for future research and application in privacy-sensitive domains, such as healthcare and finance.

REFERENCES

[1] M. Pinos, V. Mrazek, F. Vaverka, Z. Vasicek, and L. Sekanina, "Acceleration techniques for automated design of approximate convolutional neural networks," *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 13, no. 1, pp. 212–224, Mar. 2023.

[2] X. Wang, J. Lyu, J. Dinesh Peter, and B.-G. Kim, "Privacy-preserving AI framework for 6G-enabled consumer electronics," *IEEE Trans. Consum. Electron.*, vol. 70, no. 1, pp. 3940–3950, Feb. 2024.

[3] P. Panzade, D. Takabi, and Z. Cai, "Privacy-preserving machine learning using functional encryption: Opportunities and challenges," *IEEE Internet Things J.*, vol. 11, no. 5, pp. 7436–7446, May 2024.

[4] A. Golda et al., "Privacy and security concerns in generative AI: A comprehensive survey," *IEEE Access*, vol. 12, pp. 48126–48144, 2024.

[5] C. Zhou and N. Ansari, "Securing federated learning enabled NWDAF architecture with partial homomorphic encryption," *IEEE Netw. Lett.*, vol. 5, no. 4, pp. 299–303, Dec. 2023.

[6] D. Kim and C. Guyot, "Optimized privacy-preserving CNN inference with fully homomorphic encryption," *IEEE Trans. Inf. Forensics Security*, vol. 18, pp. 2175–2187, 2023.

[7] J. H. Cheon, A. Kim, M. Kim, and Y. Song, "Homomorphic encryption for arithmetic of approximate numbers," in *Proc. Int. Conf. Theory Appl. Cryptol. Inf. Secur.* Hongkong, China: Springer, 2017, pp. 409–437.

[8] Y. Yao, "SE-CNN: Convolution neural network acceleration via symbolic value prediction," *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 13, no. 1, pp. 73–85, Mar. 2023.

[9] W. Zhang, Z. Jiang, and Q. Yao, "DND: Deep learning-based directed network disintegrator," *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 13, no. 3, pp. 841–850, Mar. 2023.

[10] Z. Huang, C. Yuan, H. Ge, and T. Hou, "Optimization of substation siting and connection topology in offshore wind farm based on modified firefly algorithm," *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 13, no. 3, pp. 806–816, Mar. 2023.

[11] M. Naeem, M. Aamir, J. Yu, and O. Albalawi, "A novel approach for reconstruction of IMFs of decomposition and ensemble model for forecasting of crude oil prices," *IEEE Access*, vol. 12, pp. 34192–34207, 2024.

[12] X. Wang, S. Li, and M. Iqbal, "Live power generation predictions via AI-driven resilient systems in smart microgrids," *IEEE Trans. Consum. Electron.*, vol. 70, no. 1, pp. 3875–3884, Feb. 2024.

[13] K. M. Abubeker, S. Baskar, C. Prajitha, and P. Yadav, "Computer vision-assisted smart ICU framework for optimized patient care," *IEEE Sensors Lett.*, vol. 8, no. 1, pp. 1–4, Jan. 2024.

[14] D. R. Kothadiya, C. M. Bhatt, A. Rehman, F. S. Alamri, and T. Saba, "SignExplainer: An explainable AI-enabled framework for sign language recognition with ensemble learning," *IEEE Access*, vol. 11, pp. 47410–47419, 2023.

[15] U. Majeed, L. U. Khan, S. S. Hassan, Z. Han, and C. S. Hong, "FL-Incentivizer: FL-NFT and FL-tokens for federated learning model trading and training," *IEEE Access*, vol. 11, pp. 4381–4399, 2023.

[16] H. Lee and D. Seo, "CLSM-FL: Clustering-based semantic federated learning in non-IID IoT environment," *IEEE Internet Things J.*, vol. 11, no. 18, pp. 29838–29851, Aug. 2024.

[17] H. Akbari-Nodehi and M. A. Maddah-Ali, "Secure coded multi-party computation for massive matrix operations," *IEEE Trans. Inf. Theory*, vol. 67, no. 4, pp. 2379–2398, Apr. 2021.

[18] K. Iwamura and A. A. A. M. Kamal, "Communication-efficient secure computation of encrypted inputs using (κ, η) threshold secret sharing," *IEEE Access*, vol. 11, pp. 51166–51184, 2023.

[19] X. Li et al., "Publicly verifiable secure multi-party computation framework based on bulletin board," *IEEE Trans. Services Comput.*, vol. 17, no. 4, pp. 1698–1711, Aug. 2024.

[20] Y. Lu, B. Zhang, and K. Ren, "Maliciously secure MPC from semi-honest 2PC in the server-aided model," *IEEE Trans. Dependable Secure Comput.*, vol. 21, no. 4, pp. 3109–3125, Aug. 2024.

[21] S. Ma et al., "The era of 1-bit LLMs: All large language models are in 1.58 bits," 2024, *arXiv:2402.17764*.

[22] T. T. Nguyen, Q. B. Phan, T. X. Nghiem, C. daCunha, M. Gowanlock, and B. Cambou, "A video surveillance-based face image security system using post-quantum cryptography," *Proc. SPIE*, vol. 12544, pp. 148–155, Jun. 2023.

[23] T. T. Nguyen, Q. B. Phan, N. T. Bui, and C. daCunha, "High-secure data collection in IoT sensor networks using homomorphic encryption," *Peoc. SPIE*, vol. 12546, pp. 53–60, Jun. 2023.

[24] B. Maram, R. Majji, G. K. D. Gopisetty, A. Garg, T. Daniya, and B. S. Kumar, "Lightweight cryptography based deep learning techniques for securing IoT based E-healthcare system," in *Proc. 2nd Int. Conf. Autom., Comput. Renew. Syst. (ICACRS)*, Dec. 2023, pp. 1334–1341.

[25] T. Ryffel, E. Dufour-Sans, R. Gay, F. Bach, and D. Pointcheval, *Partially Encrypted Machine Learning Using Functional Encryption*. Red Hook, NY, USA: Curran Associates Inc., 2019.

[26] Q. B. Phan, D. C. Nguyen, and T. T. Nguyen, "Advancing privacy and accuracy with federated learning and homomorphic encryption," *TechRxiv*, Aug. 2023, pp. 1–7.

[27] L. Nguyen, B. Phan, L. Zhang, and T. Nguyen, "An efficient approach for securing audio data in AI training with fully homomorphic encryption," *TechRxiv*, Mar. 2024, pp. 1–10.

[28] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *Proc. 41st Annu. ACM Symp. Theory Comput. (STOC)*, 2009, pp. 169–178.

[29] W. S. Mcculloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *Bull. Math. Biophys.*, vol. 5, no. 4, pp. 115–133, Dec. 1943.

[30] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.

[31] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

[32] R. Roy. (2022). *Air Pollution Dataset*. Accessed: May 20, 2024. [Online]. Available: https://www.kaggle.com/datasets/rupakroy/lstm-datasets-multivariate-univariate/data

[33] E. A. Fri. (2024). *The Wildfire Dataset*. Accessed: May 5, 2020. [Online]. Available: https://www.kaggle.com/datasets/elmadafri/the-wildfire-dataset

[34] R. J. Hyndman and A. B. Koehler, "Another look at measures of forecast accuracy," *Int. J. Forecasting*, vol. 22, no. 4, pp. 679–688, Oct. 2006.

[35] T. G. Dietterich, "Approximate statistical tests for comparing supervised classification learning algorithms," *Neural Comput.*, vol. 10, no. 7, pp. 1895–1923, Oct. 1998.

[36] A. Roy and K. Roy, "DCT-CryptoNets: Scaling private inference in the frequency domain," 2024, *arXiv:2408.15231*.

[37] S. Jiang, H. Yang, Q. Xie, C. Ma, S. Wang, and G. Xing, "Lancelot: Towards efficient and privacy-preserving Byzantine-robust federated learning within fully homomorphic encryption," 2024, *arXiv:2408.06197*.

Quoc Bao Phan (Graduate Student Member, IEEE) received the bachelor's degree in electrical engineering from Hanoi University of Science and Technology, Vietnam, in 2022. He is currently pursuing the Ph.D. degree in informatics and computing with the School of Informatics, Computing, and Cyber Systems, Northern Arizona University, USA. His research interests include post-quantum cryptography, homomorphic encryption, and artificial intelligence.

Tuy Tan Nguyen (Member, IEEE) received the M.S. and Ph.D. degrees in information and communication engineering from Inha University, South Korea, in 2016 and 2019, respectively. He was a Senior Research Engineer with Conextt Inc., South Korea, from August 2019 to April 2021. From May 2021 to July 2022, he held the position of a Lecturer with the School of Global Convergence Studies and a Post-Doctoral Fellow with the Department of Electrical and Computer Engineering, Inha University. Currently, he is an Assistant Professor with the School of Informatics, Computing, and Cyber Systems, Northern Arizona University, USA. He is also a Technical Committee Member of the IEEE Circuits and Systems Society—Circuits and Systems for Communications. His research interests include the design, implementation, optimization, and applications of post-quantum cryptography, homomorphic encryption, artificial intelligence, and error correction codes in both software and hardware.