A Generic Cryptographic Deep-Learning Inference Platform for Remote Sensing Scenes

Qian Chen, *Student Member, IEEE*, Yulin Wu, Xuan Wang, *Member, IEEE*, Zoe L. Jiang, *Member, IEEE*, Weizhe Zhang, *Senior Member, IEEE*, Yang Liu, and Mamoun Alazab, *Fellow, IEEE*

***Abstract*—Deep learning plays an essential role in multidisciplinary research of remote sensing. We will encounter security problems during the data acquisition, processing, and result generation stages. Therefore, secure deep-learning inference services are one of the most important links. Some theoretical progress has been made in cryptographic deep-learning inference, but it lacks a general platform that can be realized in reality. Constantly modifying the corresponding models to approximate the plaintext results reveal the model information to a certain extent. This article proposes a generic post-quantum platform named the PyHENet, which perfectly combines cryptography with plaintext deep learning libraries. Second, we optimize the convolution, activation, and pooling functions and complete the ciphertext operation under floating point numbers for the first time. Moreover, the computation process is accelerated by single instruction multiple data streams and GPU parallel computing. The experimental results show that the PyHENet is closer to the plaintext inference platform than any other cryptographic model and has satisfactory robustness. The optimized PyHENet obtained a better accuracy of 95.05% in the high-resolution NaSC-TG2 database, which the Tiangong-2 space station received.**

***Index Terms*—Convolutional neural network (CNN), deep learning inference, fully homomorphic encryption (HE), privacy preserving, remote sensing scenes.**

Manuscript received 9 January 2023; revised 6 March 2023; accepted 20 March 2023. Date of publication 23 March 2023; date of current version 7 April 2023. This work was supported in part by the Basic Research Project of Shenzhen, China under Grant JCYJ20200109113405927 and Grant JCYJ20200109113427092, in part by the National Natural Science Foundation of China under Grant 61872109, Grant 62272131, and 62203134, in part by the National Science and Technology Major Project Carried on by Shenzhen under Grant CJGJZD20200617103000001, in part by the Guangdong Provincial Key Laboratory of Novel Security Intelligence Technologies under Grant 2022B1212010005, in part by the Science and Technology Project of Guangzhou under Grant 2020A1515010652, in part by the Key Fields R&D Project of Guangdong Province under Grant 2020B0101380001, and in part by the PINGAN-HITsz Intelligence Finance Research Center. *(Corresponding authors: Xuan Wang; Zoe L. Jiang.)*

Qian Chen and Yulin Wu are with the School of Computer Science and Technology, Harbin Institute of Technology, Shenzhen 518055, China (e-mail: qianchen@stu.hit.edu.cn; yulinwu@cs.hitsz.edu.cn).

Xuan Wang, Zoe L. Jiang, Weizhe Zhang, and Yang Liu are with the School of Computer Science and Technology, Harbin Institute of Technology, Shenzhen 518055, China, and also with the Peng Cheng Laboratory, Shenzhen 518066, China (e-mail: wangxuan@cs.hitsz.edu.cn; zoeljiang@hit.edu.cn; wzzhang@hit.edu.cn; liu.yang@hit.edu.cn).

Mamoun Alazab is with the College of Engineering, IT and Environment, Charles Darwin University, Casuarina, NT 0810, Australia (e-mail: alazab@ieee.org).

Digital Object Identifier 10.1109/JSTARS.2023.3260867

I. Introduction

With the continuous development of Big Data and deep learning, the combination of remote sensing systems and artificial intelligence is getting closer and closer, and the corresponding privacy problems [1], [2], [3] are becoming increasingly prominent. The Nature published literature "Map Opportunities" [4] and suggested the importance of geographic information, which is one of the three most promising technology areas. However, many feature information and private data are hidden in the acquired remote sensing images. Sun [5] proposed a Hashing method to explore the characteristics of RS images. Kang [6] overcame the limitation on the class discrimination. So how to effectively protect the security of the raw data and the deep learning models trained in the cloud are critical.

The structure diagram of the deep-learning inference framework with high-security level is shown in Fig. 1, which can be applied in various application scenarios [7], such as disaster detection [8], pest monitoring [9], [10], target location [11], [12] and personal GPS [13] under different devices [14]. Public key encryption scheme [15], symmetry algorithm [16], orthogonal decomposition [2], crypto-watermarking [17], and other encryption technologies are used for remote sensing image processing. However, their encryption level cannot resist quantum attacks and cannot provide a unified platform for different tasks. This is what the PyHENet is to be solved.

Cloud computing provides convenient transmission, storage, and sharing for image classification of remote sensing scenes but brings many security problems [18]. 1) The transmission of data and models in plaintext is vulnerable to reverse attacks, poisoning attacks, back door attacks [19], and many other security attacks. 2) With the continuous development of quantum computing [20], traditional cryptographic algorithms are no longer absolutely secure. Watermarking, differential privacy, partially homomorphic encryption (HE) [16], [17], [21], and other methods are poor insecurity, unable to resist quantum attacks, and cannot provide the ability of deep computing. 3) The encryption computation remains in the basic addition and multiplication operations, which cannot be combined with the standard deep learning libraries, such as TensorFlow or Pytorch to complete the generic secure deep learning computation.

Many countries and organizations are strengthening their protection of national security and data privacy. The general data protection regulation (GDPR) [22] of the European Union, which came into effect in 2018, has created a new era of

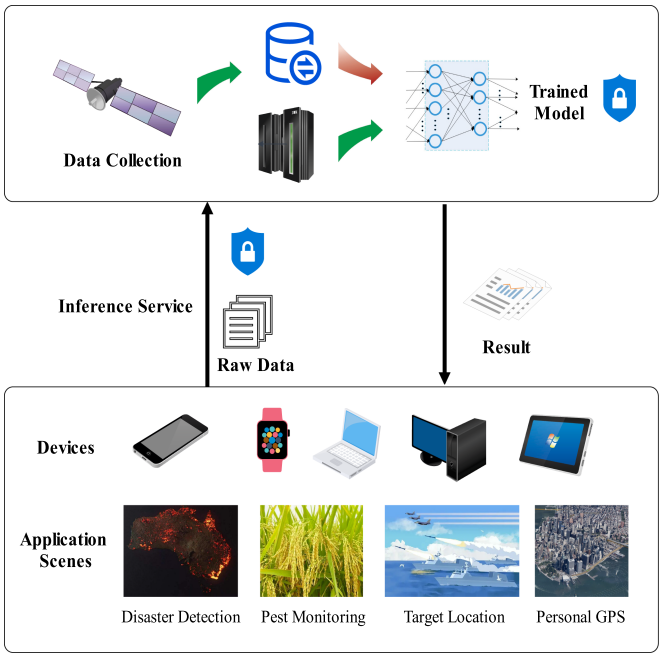

Fig. 1. Remote sensing system structure diagram with privacy preserving.

privacy-preserving. The users in Mozilla can delete their data from Mozilla's servers starting from 2020. As shown in Fig. 2, the RGB histograms of different residential images in the same class have different information features, which will reveal much private information. Chaudhari [17] combined watermarking and encryption for image copyright protection, but it only provides simple security for the image itself. Further research is needed for the more challenging post-quantum encryption for deep learning computing.

Since deep learning depends on huge computing power and massive datasets, individuals with limited computing power cannot accomplish such tasks. Inspired by the platform as a service (PaaS), deep learning as a service (DLaaS) [23], [24] has become a service with great application potential. Local users upload their personal data to the cloud, and the server outputs the results through the trained model and returns them to the client. This is called deep learning inference. In many business applications of remote sensing systems, the security of data and models is more urgent than that of ordinary applications [25]. Fig. 3 shows the encryption and decryption process of DLaaS under the condition of ciphertext.

Microsoft Research started research on privacy-preserving deep learning inference in 2016 [26], [27], based on fully HE (FHE) called the YASHE algorithm. The MiniONN framework [28] was then proposed. Meanwhile, Shokri [29] started his research on differential privacy. The follow-up research on privacy preserving deep learning inference can be divided into three categories, based on HE [30], [31], differential privacy [32], secure multiparty computing or combinations of these methods. It should be emphasized that differential privacy is the disturbance on the plaintext, which will reduce the accuracy. The GAZELLE framework [33] is based on the garbled circuit but will bring colossal communication overhead. Same problem as the BAYHENN framework [34].

According to state-of-the-art research [20], the difficult problems that traditional cryptography is based on are no longer difficult [35], [36]. But lattice-based fully HE has the ability to resist on quantum attacks, our research is also based on it.

For the privacy-preserving deep learning inference service required for remote sensing scenes, this article presents the PyHENet platform for the first time which is shown in Fig. 6. We optimize the shortcomings of traditional fully HE in convolutional layer, max pooling, sigmoid, and other computations.

In summary, the contributions of this article are as follows.

1) We develop a generic deep-learning inference platform PyHENet to protect the security of the raw data and models with fully HE.
   It can work with the plaintext library and does not require modifying the trained model. And use single instruction multiple data streams (SIMD) and GPU parallel calculation to accelerate the calculation.
2) Compared with the most advanced FHE-based framework, the complexity and generality of PyHENet can be realized without reducing the accuracy.
   We optimize and achieve the convolution, nonlinear sigmoid, and max pooling functions in FHE.
3) We optimize AlexNet model that used in state-of-the-art article of the NaSC-TG2, which collected by Tiangong-2[1] remote sensing system. We not only ensure the security but also improve the accuracy to 95.03%.

The rest of this article is organized as follows. Section II introduces the basic algorithms about fully HE and deep learning. Section III explains the contributions and optimizations of the PyHENet platform in detail. In Section IV, we finish the experimental comparison and give a detailed analysis. Finally, Section V concludes this article.

II. Preliminaries and Related Work

Remote sensing scenes usually involve people's livelihood, the economy, the military, and many other fields. Its security is far beyond simple image classification tasks [37], such as ImageNet. First, deep learning models for remote sensing tasks are trained based on many valuable datasets, so the models must be secure. In addition, the raw data of the inference service carries the private data of the remote sensing task. Our PyHENet platform can be completed under encryption conditions that can resist quantum attacks with higher security. This section introduces the relevant work in the following four aspects.

A. Privacy-Preserving Deep-Learning Inference

Inspired by platform as a service (PaaS), deep learning as a service (DLaaS) came into being in 2018 [38], [39] with great application potential. The high accuracy rate of deep learning models depends on many datasets and hardware devices.

[1]Tiangong-2 is China's first space laboratory. Where 19.6 TB of high-quality observation data have been obtained, covering a total area of 119.1 million square kilometers.

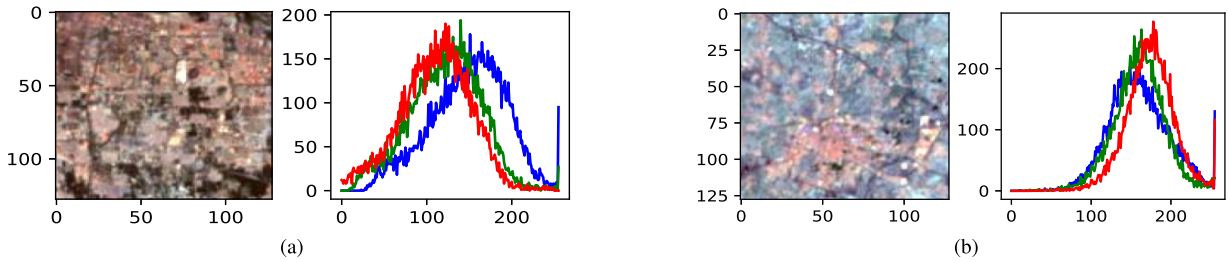

Fig. 2. Different RGB histograms of different residential images in NaSC-TG2. (a) Residential Area A. (b) Residential Area B.

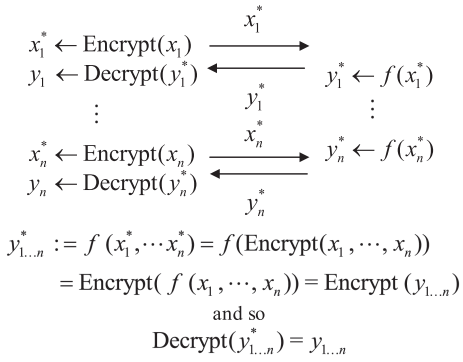

Fig. 3. DLaaS over encrypted data.

This makes datasets and trained models constantly becoming valuable. More and more researchers are exploring privacy-preserving deep-learning inference tasks. In addition, customers who use cloud services can obtain the desired results and solve the problem of weak computing power.

1) Data Perturbation-Based Privacy-Preserving: The basic idea of data perturbation is to increase the noise of the data so that the raw data is difficult to recover [40], [41]. The goal is to preserve each row of records in the database while allowing the analysis of the entire database

Differential privacy is frequently used [42], [43] for inference services. The accuracy of the results will be reduced due to the addition of noise. Also, the noise-added data is visible to anyone. Therefore, its security level cannot meet the requirements of remote sensing scene applications.

2) Cryptography-Based Privacy-Preserving: Cryptography-based methods can satisfy the invisibility and losslessness of the result accuracy. Commonly used methods include secure multiparty computation (MPC) [44], [45] and HE [34], [46]. But partial HE can only support one kind of operation, the additive or multiplicative. With the deepening of quantum computing research [20], researchers believe it will significantly impact existing cryptography. Therefore, it is necessary to research the lattice-based fully HE [47], the only methods that can resist quantum attack and support various operations.

The lattice-based fully HE is well matched with the diversity of deep learning operations and the high security of remote sensing applications. Section IV will explain how to optimize the fully HE to adapt to the general PyHENet platform. A typical application of encrypted DLaaS is shown in Fig. 3. The client provides encrypted information, and the server calculates in encrypted and returns the result to the client.

B. Lattice-Based FHE of Floating Point Computation

The security of cryptography is mainly based on some challenging problems in mathematics, such as the problem of prime factorization. Although short algorithms in quantum can solve this problem in polynomial time, other more complex mathematical problems need to be researched.

Ajtai gave proof of the lattice-based difficult problem from the worst case to the general case. Learning with Error (LWE) and variants [48], [49] are the lattice-based hard problems in the general case.

Definition 1: Learning with Errors Problem (LWE)

Give uniformly randomly generated matrices $A \in \mathbb{Z}_q^{m \times n}$, $s \in \mathbb{Z}_q^n$ and $e \in \mathbb{Z}_q^m$, obeying distribution χ and $b_i = A_i s + e_i$. Given multiple sets of (A_i, b_i), finding s is difficult.

The RLWE problem is a variant of the LWE problem, which reduces communication overhead and accelerates the encryption and decryption process. At the same time, the FHE algorithm in this article is based on ring learning with error problems. And the critical function of fully HE [47], [50] is to support the addition and multiplication operations under ciphertext

$$c_1 + c_2 = Enc_{pk}(m_1 + m_1) \tag{1}$$

$$c_1 * c_2 = Enc_{pk}(m_1 * m_1). \tag{2}$$

The process of HE is as above. The public and private keys (pk, sk) are generated from the key generation algorithm $Gen(1^n)$, and then plaintexts m_1, m_2 which are from the plaintext space M are encrypted with the public key pk. And obtain two ciphertexts $c_1 = Enc_{pk}(m_1)$ and $c_2 = Enc_{pk}(m_2)$ which are from ciphertext space C.

CKKS is an approximate computational fully HE algorithm proposed by Cheon in 2017, which supports floating-point operations through rescaling techniques. As shown in Fig. 4, a large amount of ciphertext space is saved by rescaling technology, which provides theoretical support for the computation for deep learning. In this way, the magnification of the ciphertext can be quickly reduced, thus avoiding the problems caused by the enlargement. In the previous methods, such as BFV, GSW [51], or BGV [52], the magnification gradually increases but cannot be decreased, which leads to the rapid accumulation and expansion

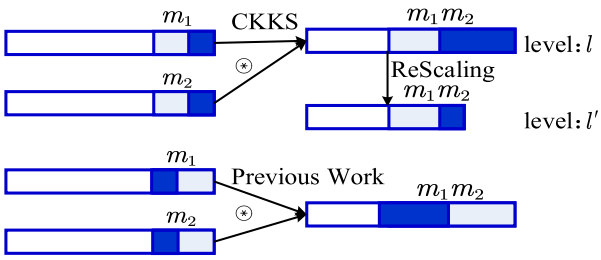

Fig. 4. ReScaling in CKKS of FHE.

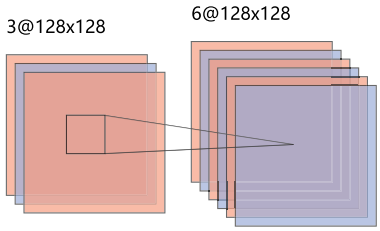

Fig. 5. Calculation of convolutional layers.

of magnification beyond the modulus of the plaintext polynomial coefficient. It also doubles the computational complexity.

The FHE can perform simple algebraic operations on the cloud directly. Although it is still a long way from cryptographic deep learning calculation, it is the security guarantee for the PyHENet platform calculation of this article.

C. Convolutional Neural Network

The convolutional neural network (CNN) is a milestone in the development of deep learning [53]. It has an excellent performance in large-scale image processing and is also the research hotspot in remote sensing [54], significantly improving image classification accuracy.

CNN has significant advantages in various applications because of its shared weight and local perception. As shown in Fig. 5, it can perform convolution calculations using convolution kernels. In addition, the sigmoid function is often used as an activation function for neural networks, and the max pool function is used to achieve this advantage of local perception. These basic neural network functions are optimized and realized in the cryptographic PyHENet platform of this article

$$x_j^l = f\left(\sum_{i \in M_j} x_i^{l-1} * w_{ij}^l + b_j^l\right) \tag{3}$$

$$\text{Sigmoid}(x) = 1/(1 + e^{(-x)}) \tag{4}$$

$$\text{MaxPooling}\begin{bmatrix} x_{00} & x_{01} & x_{02} \\ x_{10} & x_{11} & x_{12} \\ x_{20} & x_{21} & x_{22} \end{bmatrix} = \begin{bmatrix} \text{Max}(x_{00}, x_{01}, x_{10}, x_{11}) & \text{Max}(x_{01}, x_{02}, x_{11}, x_{12}) \\ \text{Max}(x_{10}, x_{11}, x_{20}, x_{21}) & \text{Max}(x_{11}, x_{12}, x_{21}, x_{22}) \end{bmatrix}. \tag{5}$$

Our CNN model in the PyHENet was based on AlexNet [55], and some modifications and optimizations have been made to adapt to our cryptography applications. Combining with the standard PyTorch library, the inference services can be implemented without modifying the external code of the training model.

III. Proposed Approach of the PyHENet Platform

Although convolution, pooling, and activation functions are widely used as necessary operations in plaintext. However, for fully HE, how to guarantee the correctness of the computation, how to reduce the computation steps, and optimize the computation time are all things we need to consider. In previous FHE implementations, neural networks tried to avoid these challenges and used linear functions with similar results instead, which is detrimental to the application of the framework and cannot achieve generalized computation for complex application scenarios.

The importance of data and model security in deep learning inference services requires no reemphasis, especially for high-security requirements, such as remote sensing scenarios. The theory of fully HE and convolutional neural networks are also briefly introduced. Naturally, we explore combining them, but it is not as simple as one plus one equals two. This section focuses on the difficulties that must be solved and optimized to construct the general inference platform of deep learning, which we named the PyHENet.

A. Privacy-Preserving Deep Learning Inference Generic Platform

The privacy-persevering of the PyHENet platform in this article is realized by the encryption algorithms based on the provable security of lattice-based FHE. PyHENet completes more complex convolution calculations under rescaling strategy and provides security for the raw data and the trained model without loss of accuracy.

The generality of the PyHENet is reflected in its ability to combine with the popular PyTorch library. It allows developers to not need additional learning of fully HE. We modify the bottom functions of the PyTorch library, including convolution, sigmoid, and max pooling functions. Support deep learning inference calculation under ciphertext. Combining the advantages of the cryptography library and artificial intelligence library, we have integrated a neural network platform called PyHENet, which represents an FHE-based neural network platform combined with the PyTorch library.

The overall framework of PyHENet is shown in Fig. 6. Because CNN is prominent in deep learning, especially for image classification in remote sensing scenes, it is reasonable to improve the CNN network under ciphertext. The deep learning inference service can be divided into three parts. The client provides private data, the server provides the trained model and the communication layer. Due to the nature of fully HE, the inference service can compute between the data and the model and return the encrypted result to the client. Finally, the client decrypts the result by the private key.

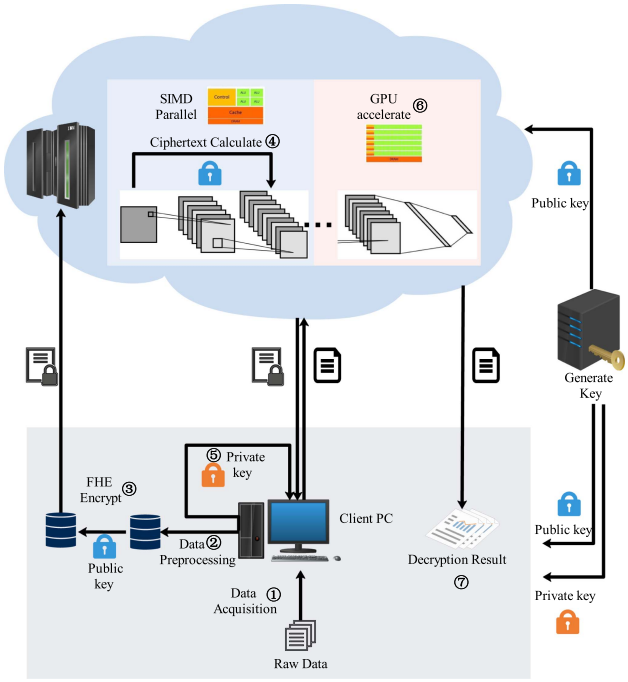

Fig. 6. Framework of the PyHENet.

On the other hand, different applications have different security requirements. Furthermore, the output after multilayer calculation can be insensitive to a certain extent. PyHENet can freely choose the depth of encryption calculation, interrupt password calculation, and balance security and calculation efficiency. The high-speed computing performance of the GPU can effectively reduce the overall computing time while ensuring security.

1) Convolution Calculation Under Ciphertext: The convolution calculation is different from the multiplication operation. It is a matrix operation. Under fully HE, the steps of the matrix dot operation under ciphertext are shown in Fig. 7

$$c_x \cdot c_w + c_b = \mathrm{Enc}_{pk}\left(x \cdot w + b\right). \tag{6}$$

We use integer p as a base for scaling in computation, and a modulus q_0 at the beginning. Let $q_l = p^l q_0$ for $0 < l \leq L$. Ciphertext vector c is defined as (c, l, v, B), where have the level $0 < l \leq L$, upper bound of message $v \in R$, upper bound of noise $B \in R$. Five basic functions are shown as follows:

$$\begin{aligned}
&\mathrm{Enc}_{pk}:\ m \longmapsto (c, L, v, B_{\text{clean}})\\
&\mathrm{Dec}_{sk}:\ (c, l, v, B) \longmapsto ((c \cdot sk)\,(\mathrm{mod}\ q_l), B)\\
&\mathrm{ReScaling}_{l\to l'}:\ (c', l, v, B) \longmapsto\\
&\left(c, l', p^{l'-l}v, p^{l'-l}B + B_{\text{scale}}\right)\\
&\mathrm{ADD}:\ ((c_1, l, v_1, B_1), (c_2, l, v_2, B_2))\\
&\mapsto (c_{\text{add}}, l, v_1 + v_2, B_1 + B_2)\\
&Mult:\ ((c_1, l, v_1, B_1), (c_2, l, v_2, B_2)) \mapsto\\
&(c_{\text{mult}}, l, v_1 v_1, v_1 B_2 + v_2 B_1 + B_1 B_2 + B_{\text{mult}}).
\end{aligned} \tag{7}$$

Algorithm 1: Convolution Calculation Based on FHE.

Input: w, x, b
Parameter: Optional list of parameters
Output: $f(x) = c_{xw+b}$

$c_x \leftarrow \mathrm{Enc}_{pk}(Encode(x\ to\ vector))$
$c_w \leftarrow \mathrm{Enc}_{pk}(Encode(w^T\ to\ vector))$
$c_b \leftarrow \mathrm{Enc}_{pk}(Encode(b\ to\ vector))$
$x \leftarrow 0$
for $i = 1$ to $RowNum(x)$ **do**
  for $j = 1$ to $ColwNum(x)$ **do**
    for $k = 1$ to $RowNum(w)$ **do**
      for $l = 1$ to $ColwNum(w)$ **do**
        $c_{xw} \leftarrow ReScaling(Mult(c_x[i][j], c_w[k][l]))$
      end for
      for $m = 1$ to $ColNum(x)$ **do**
        $c_{\text{add}} \leftarrow Mult(Rotate(c_{xw}, m), c_{10\ldots0})$
        $c_{\text{add}} \leftarrow ReScaling(c_{\text{add}})$
        $c_{xw} \leftarrow \mathrm{ADD}(c_{xw}, c_{\text{add}})$
      end for
      $c_{xw} \leftarrow ReScaling(Mult(c_{\text{add}}, c_{10\ldots0})$
      $c_{xw} \leftarrow \mathrm{ADD}(c_{xw}, \mathrm{Rotate}(c_{xw}, -k)$
    end for
  end for
end for
$c_{xw+b} \leftarrow \mathrm{ADD}(c_{xw}, c_b)$
return c_{xw+b}

The lattice-based FHE is an algebraic operation on the ring, so Algorithm 1 gives the pseudo-code for floating point convolutional computation based on the above functions. We can pack vectors into ciphertexts and perform parallel computation on the server based on the SIMD technique.

2) Proof of Lower Accuracy Loss of Convolutional Calculation: As shown in Algorithm 1 of the convolution calculation of ciphertext, it can be decomposed into addition and multiplication. Therefore, by proving that the addition and multiplication under ciphertext have lower accuracy loss, respectively, we can also prove that convolution calculation has lower accuracy loss correspondingly.

Lemma 1: Additive and Multiplicative Operations under Ciphertext are Similar to the Case of Unencrypted Float-Point Operations.

Define relative errors as $\beta = B/v$. Let (c_1, l, v_1, B_1) and (c_2, l, v_2, B_2) be the encryption of $m_1, m_2 \in S$. Let $c_a \leftarrow c_1 + c_2 (mod\ q_l)$ and $c_m \leftarrow c_{1*}c_2 (mod\ q_l)$. Then $(c_a, l, v_1 + v_1, B_1 + B_2)$ and $(c_m, l, v_1 v_1, v_1 B_2 + v_2 B_1 + B_1 B_2 + B_m)$ are valid encryptions of $m_1 + m_2$ and $m_1 m_2$.

Proof: There is a polynomial $e_1, e_2 \in R$ such that $c_1 \cdot sk = m_1 + e_1 (\mathrm{mod}\ \ q_l)$, $c_2 \cdot sk = m_2 + e_2 (\mathrm{mod}\ \ q_l)$, and $\|e_1\|_\infty^{\text{can}} \leq B_1$, $\|e_2\|_\infty^{\text{can}} \leq B_2$. It is obvious that $c_a \cdot sk = c_1 \cdot sk + c_2 \cdot sk = m_1 + m_2 + e_1 + e_2 (\mathrm{mod}\ \ q_l)$, $\|e_1 + e_2\|_\infty^{\text{can}} \leq B_1 + B_2$. We also have $c_m \cdot sk = m_1 m_2 + m_2 e_1 + m_1 e_2 + e_1 e_2 (mod\ q_l)$ and than after rescaling calculation, $\beta = \beta_1 + \beta_2 + \beta_1\beta_2 + \frac{B_{\text{mult}}(l) + p^{l-l'} B_{\text{scale}}}{v_1 v_2}$. It is very close to $\beta_1 + \beta_2$ similar to the case of unencrypted float-point multiplication under an

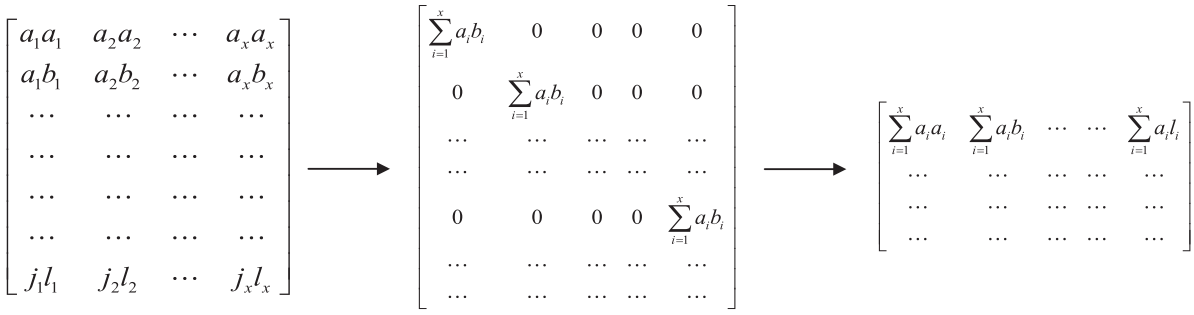

Fig. 7. Main steps of matrix dot operation in ciphertext.

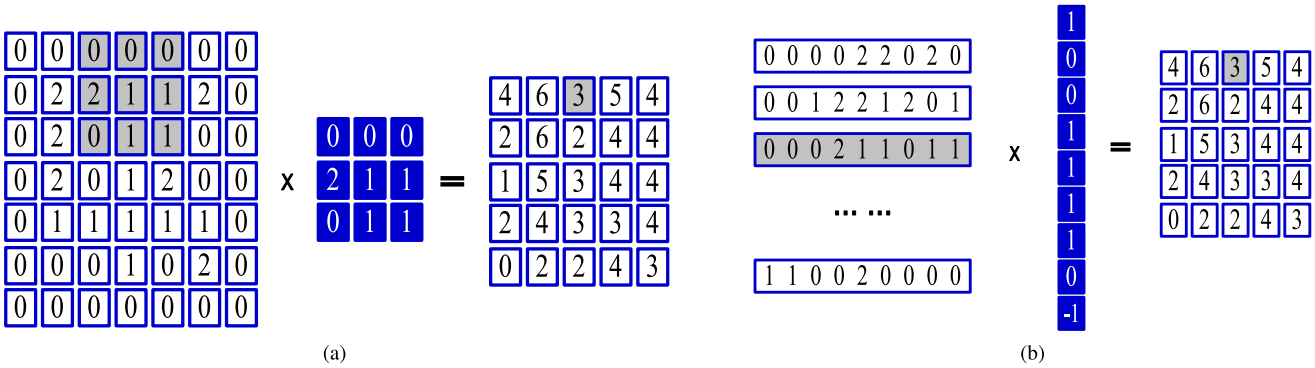

Fig. 8. Optimized convolution calculation. (a) Ordinary method. (b) Image-to-column (Im2col) method.

TABLE I
ADVANTAGE OF SIMD

	Using SIMD	Not Using SIMD
Number of Ciphertext	n	1

appropriate choice of parameter and level. Since the convolution operation consists of multiplication and addition under ciphertext, the lemma also holds. ■

B. Aided Parallel Computing Based on GPU and SIMD

As shown in Fig. 6, to balance security and computing performance, the PyHENet can freely select the number of network layers for GPU computing. On the other hand, we also use SIMD technology to package ciphertext to optimize ciphertext computation, which makes homomorphic computing faster and more accessible for the current scene. SIMD is a parallel computing technology that can significantly more quickly execute instructions. Table I shows the advantage of SIMD. It only needs one encryption and one homomorphic computation to operate on the plaintext vector.

C. Optimization of Neural Network Under Ciphertext

1) Linear Convolution Function: In the encryption convolution calculation, the computational cost should be reduced as much as possible. Except for SIMD parallel computation under ciphertext, compared with the traditional sliding window matrix computation, the image-to-column (im2col) [56] optimization method used in this article can accelerate and support homomorphic computation. The optimized schematic is shown in Fig. 8.

2) Nonlinear Sigmoid Activation Function: Different from other deep learning methods based on HE, this article uses the Taylor expansion to approximate sigmoid function, instead of replacing it with other functions.

The sigmoid function $f(x) = 1/(1 + \exp^{(-x)})$ is the most basic activation function in neural networks and is widely used. We cannot give up the original open-source system because of the difficulty of realizing complete homomorphism

$$f(x) = 1/2 + 1/4x - 1/48x^3 + 1/480x^5 - 17/86040x^7 + 31/1451520x^9 + O\left(x^{11}\right). \quad (8)$$

As HE does not support the nonlinear operation, we use Taylor expansion to approximate the Sigmoid function. Fig. 9 shows the comparison of the Taylor expansion in different orders with the Sigmoid function, and it can obtain linear approximation results. Of course, the deeper the order of Taylor expansion, the closer the effect will be to the actual value. The experiment shows that the sixth-order expansion can meet our requirements.

3) Max Pooling Function: Unlike other homomorphic algorithms that use average pooling instead of maximum pooling, the maximum pooling function is better suited for reducing the trained model. More importantly, for privacy-preserving deep

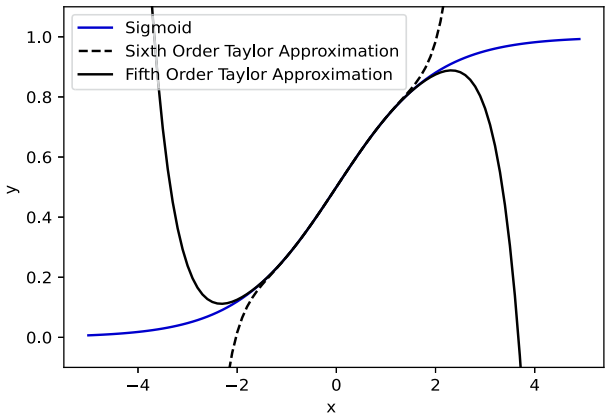

Fig. 9. Taylor expansion of sigmoid functions of different orders.

learning inference services, changing the model structure has already damaged the private data of the model, even if it does not damage the parameters. The trained model is precious for the collection of the training data. The average pool function is often used in fully HE. It only requires homomorphic addition to complete the pooling, as shown in the following equation. However, the maximum pool function measurement needs our optimization.

$$\text{Encrypt}\left(\sum_{i=1}^{n} x_i\right) = \sum_{i=1}^{n} \text{Encrypt}(x_i). \tag{9}$$

In the encryption condition, it is harder to complete than number size, so we modify the conditional operations, shown in the following:

$$\begin{aligned}\max(a,b) &= \begin{cases} b+a-b, & if\ a \geq b \\ b+0, & if\ a < b \end{cases} \\ &= b + \max(0, a-b) \\ &= b + \text{ReLU}(0, a-b). \end{aligned} \tag{10}$$

Correspondingly, the max pooling function in the ciphertext can be transformed into the following equation:

$$\begin{aligned}&\text{Enc}(\max(a,b)) = \text{Enc}(b) - \text{Enc}(\text{ReLU}(0, a-b)) \\ &\text{where Enc}(a-b) = \text{Enc}(a) - \text{Enc}(b). \end{aligned} \tag{11}$$

IV. Experiments and Evaluation

In this section, we implement the PyHENet platform with fully HE in the traditional PyTorch library, which is based on the rerealization of the various functions for deep learning inference services that need to be completed in the previous section.

The experimental analysis in this chapter is completed in three dimensions: 1) comparison with state-of-the-art ciphertext-based deep learning prediction models, 2) comparison with the corresponding plaintext accuracy, and 3) comparison in more complex remote sensing scenes.

Fig. 10. Dateset of MNIST.

A. Dataset and Experimental Settings

We deploy the generic cryptographic deep-learning inference platform with two NVIDIA A100 GPUs to satisfy the larger memory requirements for more complex security services. It not only improves the speed of the training process but also assists the PyHENet platform more effectively in the inference tasks mentioned in Section II, balancing security with computational speed.

Since previous deep learning inference experiments based on fully HE were done and compared in the MNIST[2] dataset, we must experiment on this dataset first. The MNIST is the classification task for gray images, which is the introduction dataset for deep learning. As shown in Fig. 10, it contains ten classes of images from 0–9, and the inference service is to classify them precisely, with 70 000 images of 28*28 pixels.

In MNIST-based experiments, we focus on the difference between the PyHENet with other cryptographic frameworks, the complexity of the model, or the similarity with the plaintext model rather than the accuracy itself. Since this dataset is primary, distinguishing the accuracy is meaningless. We try to make the model's parameters not optimal to achieve the best accuracy. The inference process with accuracy change can better show the rate of change.

After proving the generality of the PyHENet platform, we apply it to the more complex remote sensing image classification tasks. The NaSC-TG2[3] dataset is collected from the Tiangong-2 space lab [57], [58] in China, which has higher image quality and can enables richer remote sensing scenes compared with the experiments based on maps. It has ten natural scenes, each with 2000 color images of 128*128 pixels, as shown in Fig. 11. The PyHENet achieves the fully homomorphic encrypted deep learning inference service on remote sensing data first while gaining better accuracy and providing a general platform for high-security application services.

B. Experiment and Aanalysis of the PyHENet Platform

The main contribution of this article is to provide a more practical general security platform in complex scenarios with multiple data sources or multiple network models. The PyHENet realized convolutional, activation, and pooling functions, which are essential in standard neural networks. Of course, to obtain higher accuracy, the PyHENet requires scenario-oriented personalization, such as data preprocessing and function parameter tuning, like standard deep learning libraries, such as Pytorch. Meanwhile, the PyHENet platform makes secure inference service more in line with standard deep learning libraries and

[2]Dateset of MNIST: http://yann.lecun.com/exdb/mnist/

[3]Dateset of Natural Scene Classification With Tiangong-2 Remotely Sensed Imagery (NaSC-TG2): http://www.msadc.cn/main/setsubDetail?id=1370312964720037889

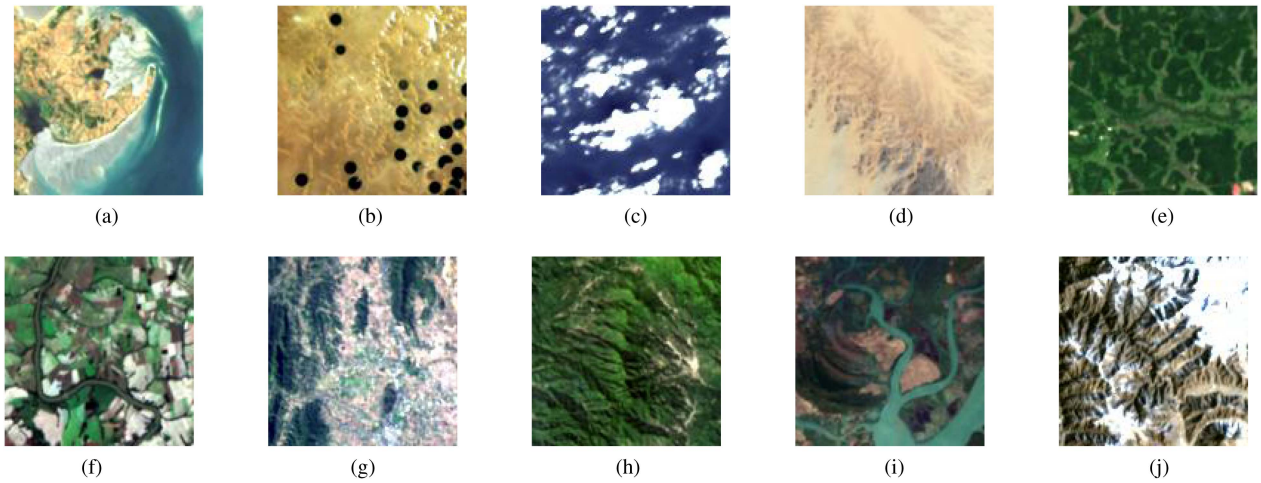

Fig. 11. Ten classes of remote sensing images in NaSC-TG2 dateset. (a) Beach. (b) Circular Farmland. (c) Cloud. (d) Desert. (e) Forest. (f) Mountain. (g) Rectangular Farmland. (h) Residential. (i) River. (j) Snowberg.

TABLE II
COMPARE THE PYHENET WITH THE CRYPTONETS IN MNIST

PyHENet 2	PyHENet 1	CryptoNets [26]
Conv1: kernel size: **3*3**, stride:**1**, output **6** channels	**Conv1:** kernel size: **3*3**, stride:**1**, output **6** channels	**Conv1:** kernel size: **5*5**, stride:**2**, output **5** channels
Sigmoid: highly approximate, outpus **4056** numbers	**Sigmoid:** highly approximate, outpus **4056** numbers	**Square Activation:** sequare each of **835** output numbers in Conv1
Max Pooling: generates **1014** output numbers frome 4056 above	/	**Average Pooling:** generates **100** output numbers frome 835 above
Conv2: kernel size: **3*3**, stride:**1**, output **7** channels, output **847** numbers	**Conv2:** kernel size: **3*3**, stride:**1**, output **7** channels, output **4032** numbers	/
FC1: input **847** numbers, output **128** numbers	**FC1:** input **4021** numbers, output **128** numbers	/
Sigmoid: highly approximate, output **128** numbers	**Sigmoid:** highly approximate, output **128** numbers	**Square Activation:** sequare each of **100** outputs numbers above
FC2: input **128** numbers, output **10** numbers	**FC2:** input **128** numbers, output **10** numbers	/
Output: generates **10** output numbers	**Output:** generates **10** output numbers	**Output:** generates **10** output numbers

[1]CryptoNets Framework: https://sealcrypto. codeplex.com/

enables no need to modify the external code after model training. This contribution is original and has a wide range of practical applications. The experiments were applied in remote sensing scenarios and obtained better accuracy.

1) Comparison With State-of-the-Art Fully Homomorphic Encryption Models: This subsection focuses on comparing with state-of-the-art fully homomorphic neural network models.

The CrypNets [26] is one of the few frameworks that expose the model parameters and code. At the same time, it has been widely recognized. Therefore, comparing it with PyHENet is more convincing.

As PyHENet is a general platform, it can be freely combined to generate different models. Table II gives two different models of it. Both the models in the PyHENet platform have more general and complex sigmoid and full connectivity functions than the CryptoNets. This undoubtedly requires more optimization methods. Moreover, the PyHENet can be used together with the PyTorch library.

In addition, PyHENet 2 has an increased max pooling function compared to PyHENet 1, which can reduce the matrix calculation under the ciphertext. Moreover, there is no doubt about the importance of pooling computation in traditional convolutional networks. It additionally increases the accuracy of the results, so implementing maximum pooling in the ciphertext is more relevant to standard deep-learning libraries.

We also compare the PyHENet with other state-of-the-art methods in Table III. Since our platform supports encryption computation under floating-point data, it allows deeper computation. More importantly, the PyHENet is much closer to the actual requirements and can be implemented with a general platform like Pytorch. Moreover, the PyHENet has three advantages in optimizing the neural network: optimized linear convolution calculation function, nonlinear sigmoid activation function, and max pool function, which have been described in detail in the previous section.

Based on the model parameters in Table II, the following subsection compares the experimental performance of the PyHENet.

2) Comparison With Deep Learning Inference Models Under Plaintext: The experiments in this subsection aim to compare the performance of PyHENet with the plaintext convolution

TABLE III
COMPARE WITH THE HOMOMORPHIC ENCRYPTION-BASED NEURAL NETWORKS IN MNIST

Framework	HE Algorithm	Security	Accuracy	Data Type	General Platform
PyHENet	**CKKS**	Fully HE	**99.9**	**Float**	√
BAYHENN [34]	BFV	Fully HE	98.93	Integer	×
LoLa [27]	BFV	Fully HE	98.95	Integer	×
GAZELLE [33]	BFV	Fully HE	99.05	Integer	×
Delphi [59]	BFV	Fully HE	/	Integer	×
Hunter [60]	BFV	Fully HE	99.37	Integer	×
CryptoDL [30]	HE	Fully HE	99.95	Integer	×
CryptoNets [26]	YASHE	Fully HE	99	Integer	×
Popcorn [61]	Paillier	Partial HE	99	Float	×

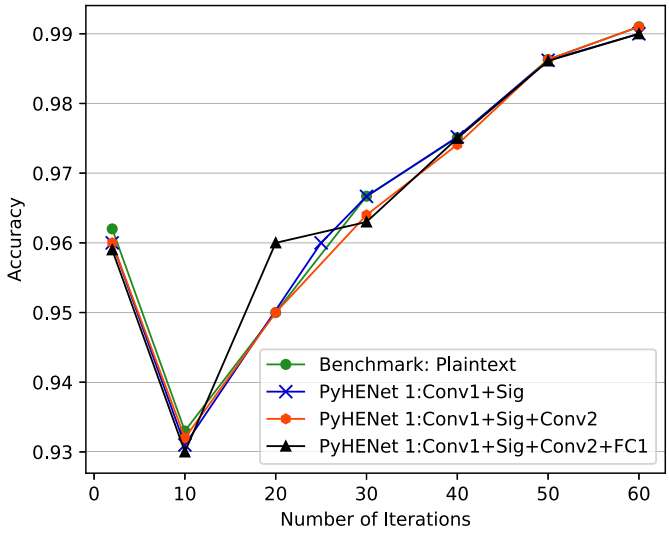

Fig. 12. Compare the accuracy of the PyHENet 1 in different security requirements.

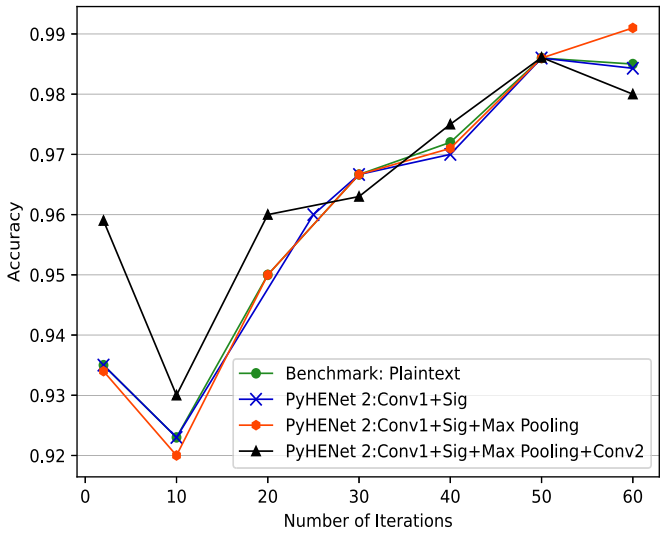

Fig. 13. Compare the accuracy of the PyHENet 2 in different security requirements.

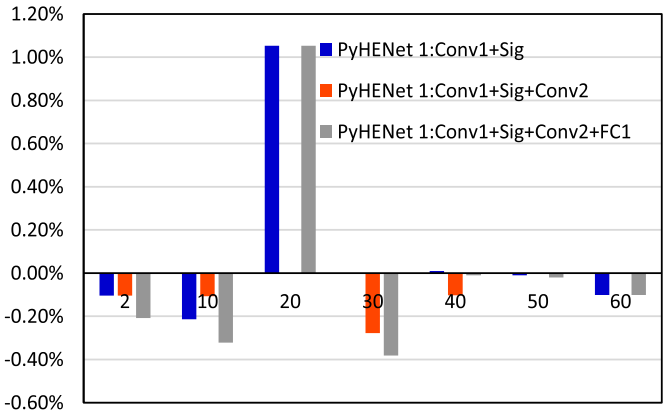

Fig. 14. Relative accuracy percentage of PyHENet 1 that compare with plaintext network.

neural network model rather than to obtain higher accuracy. Therefore, this experiment was conducted in different iterations of the model to get the relative performance of different models in different accuracy cases.

Figs. 12 and 13 compare the accuracy of the plaintext benchmark with PyHENet 1 and PyHENet 2 in different security levels (number of network layers for encryption computation). We modify the parameters so that the model cannot quickly obtain the optimum accuracy in the previous iterations. We found that PyHENet 1 and 2 have good robustness during 60 iterations. It can also be seen from the figures that the inference performance of the encrypted neural network has almost the same relative performance as that of the benchmark. In addition, different security-level models can obtain almost the same accuracy.

Figs. 14 and 15, respectively, compare the accuracy of PyHENet 1 and PyHENet 2 with that of the plaintext network and obtain the relative accuracy percentage. As can be seen from the above figures, when the number of iterations of the network is low, the relative accuracy of the network fluctuates greatly. This phenomenon is consistent with the actual situation. The neural network needs many iterations to obtain high accuracy. With the increased iteration times, no matter what kind of ciphertext network, it has reached the same accuracy as a plaintext network and tends to be stable.

In this part of the experiment, the relative accuracy under the impact condition can be obtained by changing the performance of the network, which can better reflect the robustness of the PyHENet platform.

3) Comparison in Remote Sensing Scenes: The experiments in the previous subsections have demonstrated the security, robustness, and high accuracy of the PyHENet platform. This has laid a foundation for exploring more complex remote sensing applications.

We conduct cryptographic deep-learning inference experiments on the NaSC-TC2 dataset provided by the Tiangong-2 space laboratory. In their latest research, Zhou et al. [57]

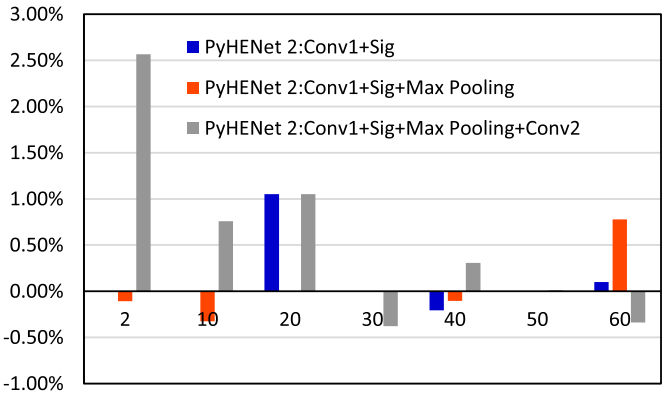

Fig. 15. Relative accuracy percentage of PyHENet 2 that compare with plaintext network.

TABLE IV
COMPARISON OF THE PYHENET PLATFORM IN NASC-TG2

PyHENet 3	Zhou's AlexNet [55], [57]
Conv1: kernel size: 7*7, stride:2, output **128** channels	**Conv1:** kernel size: 11*11, stride:4, output **96** channels / **ReLU** **Local Response Norm**
/	
Max Pooling: kernel size: **3*3**, stride:2	**Max Pooling:** kernel size: **3*3**, stride:2
Conv2: kernel size: **3*3**, stride:**2**, output **192** channels	**Conv2:** kernel size: **5*5**, stride:**1**, output **256** channels / **ReLU** **Local Response Norm**
/	
Max Pooling: kernel size: **3*3**, stride:2	**Max Pooling:** kernel size: **3*3**, stride:2
Conv3: kernel size: **3*3**, stride: **1**, output **128** channels **Sigmoid**	**Conv3:** kernel size: **3*3**, stride: **1**, output **384** channels **ReLU**
/	**Conv4:** kernel size: **3*3**, stride: **1**, output **384** channels / **ReLU**
/	**Conv5:** kernel size: **3*3**, stride: **1**, output **384** channels / **ReLU**
/	**Conv6:** kernel size: **3*3**, stride: **1**, output **256** channels / **ReLU**
/	**Max Pooling:** kernel size: **3*3**, stride:2
FC1: input **2048** numbers, output **1024** numbers **Sigmoid:**	**FC1:** input **4096** numbers, output **4096** numbers **ReLU**
FC2: input **1024** numbers, output **512** numbers /	**FC2:** input **4096** numbers, output **4096** numbers **ReLU**
FC3: input **512** numbers, output **10** numbers	**FC3:** input **4096** numbers, output **100** numbers
Output: generates **10** output numbers	**Output:** generates **100/10** output numbers

found that AlexNet could achieve 89.39% accuracy in this high-resolution remote sensing data classification task. It possesses higher accuracy than deeper neural networks, such as VGG.

Therefore, we adjust the model structure of PyHENet 3 to let it be similar to the AlexNet and replicate the experiment using our inference model based on fully HE. Table V carries out the ablation experiments of model PyHENet 3, which reflects the effectiveness of its structure. Table IV shows the comparison of PyHENet with Zhou's AlexNet. We upgrade the number of layers and the difficulty of the model in PyHENet once again.

TABLE V
ABLATION EXPERIMENT OF PYHENET 3 WITH DIFFERENT KINDS OF LAYERS

Model	Top-1 Accuracy
PyHENet 3	95.05%
(w/o) Fully homomorphic encryption	95.10%
(w/o) Max Pooling & Con3	82.05%
(w/o) Max Pooling & Con3 & Sigmoid	69.51%
(w/o) Two Max Pooling & Con3 & Sigmoid	54.66%

TABLE VI
COMPARE WITH STATE-OF-THE-ART NEURAL NETWORKS IN NASC-TG2

Model	Encryption	Number of Layers	Accuracy	
			20% Training	70% Training
PyHENet	√	**9**	**90.43%**	**95.05%**
Zhou's AlexNet [57]	×	22	89.39%	/

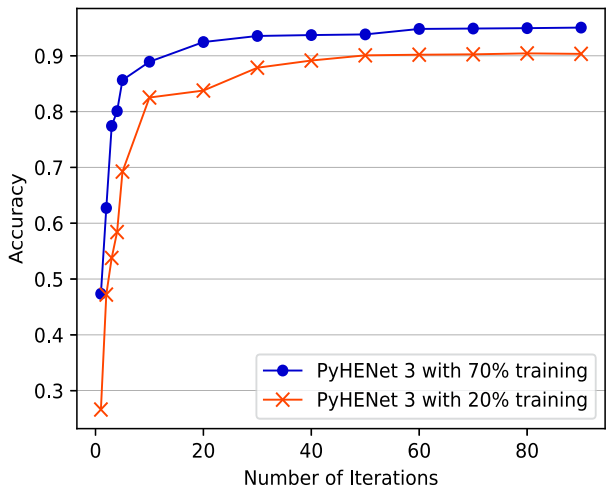

Fig. 16. Accuracy of PyHENet 3 in NaSC-TG2.

Through several experimental comparisons, in the task of NaST-TC2, we found that the traditional AlexNet network, which is computed with 22 functions, has redundant functions. So we reduce the neural network to the model with nine layers of functions. Moreover, the specific parameters of each layer are given in Table IV.

Since Zhou's experiments were implemented based on 20% of the training data, this article does the inference experiments on 20% and 70% of the trained models, respectively. As shown in Table VI, we can obtain higher accuracy even though the PyHENet model has fewer functions and is based on a complex computation of fully HE. It can be seen that high security does not trigger a decrease in accuracy. In addition, the generic model can be appropriately tuned to obtain higher accuracy.

From the graph in Fig. 16, we can see the accuracy trend of the inference model. The accuracy increases sharply in the 1st to 20th iterations, and the model accuracy stabilizes through the 20th to 40th iterations. In addition, the accuracy trend is the same for different amounts of training data.

With the encrypted scenario, we not only do not have less inference accuracy but also do not affect the the final results. It can be seen that the generic platform PyHENet proposed in this article can indeed provide a secure inference service for remote sensing scenes. The security of the remote sensing scenario is ensured and the platform is generic and easy to operate.

V. Conclusion

Remote sensing scenes have an increasing demand for security, especially inference services in deep learning. Moreover, privacy-preserving deep learning is a challenging but essential research topic. How to balance security and performance is critical. We design a general platform for deep learning inference called the PyHENet. It can secure both the client's data and the trained model based on post-quantum encryption theory. By implementing and optimizing the neural network under ciphertext, the security is further improved while reducing the development difficulty. In the future, we can further optimize the platform to satisfy more deep learning models, such as LSTM or GAN. The exploration of distributed secure remote sensing applications is another valuable research direction.

References

[1] X. Zhang, G. Zhang, X. Huang, and S. Poslad, "Granular content distribution for IoT remote sensing data supporting privacy preservation," *Remote Sens.*, vol. 14, no. 21, 2022, Art. no. 5574.

[2] L. Jiang, T. Niu, Z. Xu, and Y. Xu, "Integrating encryption and marking for remote sensing image based on orthogonal decomposition," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 8, no. 5, pp. 2232–2239, May 2015.

[3] L. Jiang, H. Zheng, H. Wang, and Z. Quan, "A multipermutation superposition coding-based fragile watermarking for probabilistic encryption," *Multimedia Tools Appl.*, vol. 81, no. 21, pp. 30025–30048, 2022.

[4] V. Gewin, "Mapping opportunities," *Nature*, vol. 427, no. 6972, pp. 376–377, 2004.

[5] Y. Sun et al., "Multisource data reconstruction-based deep unsupervised hashing for unisource remote sensing image retrieval," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, 2022, Art. no. 5546316.

[6] J. Kang, R. Fernández-Beltran, Z. Ye, X. Tong, P. Ghamisi, and A. Plaza, "Deep metric learning based on scalable neighborhood components for remote sensing scene characterization," *IEEE Trans. Geosci. Remote Sens.*, vol. 58, no. 12, pp. 8905–8918, Dec. 2020.

[7] J. Liang, Y. Deng, and D. Zeng, "A deep neural network combined CNN and GCN for remote sensing scene classification," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 13, pp. 4325–4338, 2020.

[8] Y. Cao et al., "Forest disaster detection method based on ensemble spatial–spectral genetic algorithm," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 15, pp. 7375–7390, 2022.

[9] J. Zhang et al., "Monitoring plant diseases and pests through remote sensing technology: A review," *Comput. Electron. Agriculture*, vol. 165, 2019, Art. no. 104943.

[10] Y. Dong et al., "Automatic system for crop pest and disease dynamic monitoring and early forecasting," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 13, pp. 4410–4418, 2020.

[11] Z. Sun, X. Li, W. Yi, G. Cui, and L. Kong, "A coherent detection and velocity estimation algorithm for the high-speed target based on the modified location rotation transform," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 11, no. 7, pp. 2346–2361, Jul. 2018.

[12] Y. Cai, Y. Ding, H. Zhang, J. Xiu, and Z. Liu, "Geo-location algorithm for building targets in oblique remote sensing images based on deep learning and height estimation," *Remote Sens.*, vol. 12, no. 15, 2020, Art. no. 2427.

[13] C. Gao, D. Yang, X. Hong, Y. Xu, B. Wang, and Y. Zhu, "Experimental results about traffic flow detection by using GPS reflected signals," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 11, no. 12, pp. 5076–5087, Dec. 2018.

[14] B. Zhang et al., "Progress and challenges in intelligent remote sensing satellite systems," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 15, pp. 1814–1822, 2022.

[15] X. Wang, J. Li, and H. Yan, "An improved anti-quantum MST_3 public key encryption scheme for remote sensing images," *Enterprise Inf. Syst.*, vol. 15, no. 4, pp. 530–544, 2021.

[16] Z. Yu and Z. Yang, "Method of remote sensing image detail encryption based on symmetry algorithm," *J. Ambient Intell. Humanized Comput.*, pp. 1–9, 2021. [Online]. Available: https://link.springer.com/article/10.1007/s12652-020-02818-x

[17] S. Zope-Chaudhari, P. Venkatachalam, and K. M. Buddhiraju, "Secure dissemination and protection of multispectral images using crypto-watermarking," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 8, no. 11, pp. 5388–5394, Nov. 2015.

[18] R. Sen, G. R. Heim, and Q. Zhu, "Artificial intelligence and machine learning in cybersecurity: Applications, challenges, and opportunities for MIS academics," *Commun. Assoc. Inf. Syst.*, vol. 51, pp. 179–209, 2022.

[19] J. Wu, *Cyberspace Mimic Defense - Generalized Robust Control and Endogenous Security* (ser. Wireless Networks). Berlin, Germany: Springer, 2020.

[20] F. Arute et al., "Quantum supremacy using a programmable superconducting processor," *Nature*, vol. 574, no. 7779, pp. 1948–1957, 2019.

[21] F. Zhou, S. Qin, R. Hou, and Z. Zhang, "Privacy-preserving image retrieval in a distributed environment," *Int. J. Intell. Syst.*, vol. 37, no. 10, pp. 7478–7501, 2022.

[22] R. Jones and D. Tahri, "An overview of EU data protection rules on use of data collected online," *Comput. Law Secur. Rev.*, vol. 27, no. 6, pp. 630–636, 2011.

[23] H. C. Tanuwidjaja, R. Choi, S. Baek, and K. Kim, "Privacy-preserving deep learning on machine learning as a service-A comprehensive survey," *IEEE Access*, vol. 8, pp. 167425–167447, 2020.

[24] C. Wang et al., "SOLAR: Services-oriented deep learning architectures-deep learning as a service," *IEEE Trans. Serv. Comput.*, vol. 14, no. 1, pp. 262–273, Jan./Feb. 2021.

[25] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. 20th Int. Conf. Artif. Intell. Statist.*, 2017, vol. 54, pp. 1273–1282.

[26] R. Gilad-Bachrach, N. Dowlin, K. Laine, K. Lauter, M. Naehrig, and J. Wernsing, "CryptoNets: Applying neural networks to encrypted data with high throughput and accuracy," in *Proc. 33rd Int. Conf. Mach. Learn.*, 2016, vol. 48, pp. 201–210.

[27] A. Brutzkus, R. Gilad-Bachrach, and O. Elisha, "Low latency privacy preserving inference," in *Proc. 36th Int. Conf. Mach. Learn.*, 2019, vol. 97, pp. 812–821.

[28] J. Liu, M. Juuti, Y. Lu, and N. Asokan, "Oblivious neural network predictions via miniONN transformations," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2017, pp. 619–631.

[29] R. Shokri and V. Shmatikov, "Privacy-preserving deep learning," in *Proc. 22nd ACM SIGSAC Conf. Comput. Commun. Secur.*, 2015, pp. 1310–1321.

[30] E. Hesamifard, H. Takabi, and M. Ghasemi, "CryptoDL: Deep neural networks over encrypted data," 2017, *arXiv:1711.05189*.

[31] H. Chen, W. Dai, M. Kim, and Y. Song, "Efficient multi-key homomorphic encryption with packed ciphertexts with application to oblivious neural network inference," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2019, pp. 395–412.

[32] B. D. Rouhani, M. S. Riazi, and F. Koushanfar, "Deepsecure: Scalable provably-secure deep learning," in *Proc. 55th Annu. Des. Automat. Conf.*, 2018, pp. 1–6.

[33] C. Juvekar, V. Vaikuntanathan, and A. Chandrakasan, "GAZELLE: A low latency framework for secure neural network inference," in *Proc. 27th USENIX Conf. Secur. Symp.*, 2018, pp. 1651–1668.

[34] P. Xie, B. Wu, and G. Sun, "BAYHENN: Combining Bayesian deep learning and homomorphic encryption for secure DNN inference," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, 2019, pp. 4831–4837.

[35] R. A. Hallman, M. H. Diallo, M. A. August, and C. T. Graves, "Homomorphic encryption for secure computation on Big Data," in *Proc. 3rd Int. Conf. Internet Things, Big Data Secur.*, 2018, pp. 340–347.

[36] M. S. Riazi and F. Koushanfar, "Privacy-preserving deep learning and inference," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Des.*, 2018, pp. 1–4.

[37] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[38] S. Lobo, "IBM rolls out deep learning as a service (DLaaS) program for ai developers," 2018.

[39] L. Cui, Z. Chen, S. Yang, R. Chen, and Z. Ming, "A secure and decentralized DLaaS platform for edge resource scheduling against adversarial attacks," *IEEE Trans. Comput.*, to be published, doi: 10.1109/TC.2021.3074806.

[40] Z. Liang, X. Chen, L. Zhang, J. Liu, and Y. Zhou, "Correlation classifiers based on data perturbation: New formulations and algorithms," *Pattern Recognit.*, vol. 100, 2020, Art. no. 107106.

[41] H. Kargupta, S. Datta, Q. Wang, and K. Sivakumar, "Random-data perturbation techniques and privacy-preserving data mining," *Knowl. Inf. Syst.*, vol. 7, no. 4, pp. 387–414, 2005.

[42] S. Truex, L. Liu, M. E. Gursoy, W. Wei, and L. Yu, "Effects of differential privacy and data skewness on membership inference vulnerability," in *Proc. IEEE 1st Int. Conf. Trust, Privacy Secur. Intell. Syst. Appl.*, 2019, pp. 82–91.

[43] V. Karwa and A. Slavkovic, "Inference using noisy degrees: Differentially private b-model and synthetic graphs," *Ann. Statist.*, vol. 44, no. 1, pp. 87–112, 2016.

[44] F. Boemer, R. Cammarota, D. Demmler, T. Schneider, and H. Yalame, "MP2ML: A mixed-protocol machine learning framework for private inference," in *Proc. 15th Int. Conf. Availability, Rel. Secur.*, 2020, pp. 1–10.

[45] M. Zanon and S. Gros, "Safe reinforcement learning using robust MPC," *IEEE Trans. Autom. Control*, vol. 66, no. 8, pp. 3638–3652, Aug. 2021.

[46] L. T. Phong, Y. Aono, T. Hayashi, L. Wang, and S. Moriai, "Privacy-preserving deep learning via additively homomorphic encryption," *IEEE Trans. Inf. Forensics Secur.*, vol. 13, no. 5, pp. 1333–1345, May 2018.

[47] D. J. Bernstein and T. Lange, "Post-quantum cryptography," *Nature*, vol. 549, no. 7671, pp. 188–194, 2017.

[48] M. J. Frank, B. S. Woroch, and T. Curran, "Error-related negativity predicts reinforcement learning and conflict biases," *Neuron*, vol. 47, no. 4, pp. 495–501, 2005.

[49] Z. Brakerski and V. Vaikuntanathan, "Efficient fully homomorphic encryption from (standard) LWE," in *Proc. IEEE 52nd Annu. Symp. Found. Comput. Sci.*, 2011, pp. 97–106.

[50] C. Gentry, "Computing arbitrary functions of encrypted data," *Commun. ACM(Earlier Version STOC 2009)*, vol. 53, no. 3, pp. 97–105, 2010.

[51] C. Gentry, A. Sahai, and B. Waters, "Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based," in *Proc. 33rd Annu. Int. Cryptology Conf.*, 2013, vol. 8042, pp. 75–92.

[52] Z. Brakerski, C. Gentry, and V. Vaikuntanathan, "(Leveled) fully homomorphic encryption without bootstrapping," *ACM Trans. Computation Theory*, vol. 6, no. 3, pp. 13:1–13:36, 2014.

[53] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[54] Y. Ren, C. Zhu, and S. Xiao, "Small object detection in optical remote sensing images via modified faster R-CNN," *Appl. Sci.*, vol. 8, no. 5, 2018, Art. no. 813.

[55] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. 26th Annu. Conf. Neural Inf. Process. Syst.*, 2012, pp. 1106–1114.

[56] K. Chellapilla, S. Puri, and P. Simard, "High performance convolutional neural networks for document processing," in *Proc. 10th Int. Workshop Front. Handwriting Recognit. Suvisoft*, 2006, pp. 1–6.

[57] Z. Zhou et al., "NaSC-TG2: Natural scene classification with Tiangong-2 remotely sensed imagery," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 14, pp. 3228–3242, 2021.

[58] H. Zhu, "Scientific experiments on Tiangong-2, the predecessor of the China space station," *Nat. Sci. Rev.*, vol. 9, no. 12, 2022, Art. no. nwac189.

[59] P. Mishra, R. Lehmkuhl, A. Srinivasan, W. Zheng, and R. A. Popa, "Delphi: A cryptographic inference service for neural networks," in *Proc. 29th USENIX Secur. Symp. Assoc.*, 2020, pp. 2505–2522.

[60] Y. Cai, Q. Zhang, R. Ning, C. Xin, and H. Wu, "Hunter: HE-friendly structured pruning for efficient privacy-preserving deep learning," in *Proc. ACM Asia Conf. Comput. Commun. Secur.*, 2022, pp. 931–945.

[61] J. Wang, C. Jin, S. Meftah, and K. M. M. Aung, "Popcorn: Paillier meets compression for efficient oblivious neural network inference," 2021, *arXiv:2107.01786*.

Qian Chen (Student Member, IEEE) received the master's degree in computer science, from the Harbin Institute of Technology, Shenzhen, China, in 2018, where she is currently working toward the Ph.D. degree in computer science.

She is an Experimentalist with the Experimental and Practical Research Center, Harbin Institute of Technology, from 2018 to 2019, teaching experimental courses on compilation principle, operating system, and high-level language programming. Her research interests include homomorphic encryption, remote sensing, and game theory.

Yulin Wu received the Ph.D. degree in computer science from the Harbin Institute of Technology, Shenzhen, China, in 2021.

She is currently an Assistant Professor with the School of Computer Science and Technology, Harbin Institute of Technology. Her research interests include secure multiparty computation, remote sensing, and cloud security.

Xuan Wang (Member, IEEE) received the Ph.D. degree from Harbin Institute of Technoloy, Harbin, China, in 1997.

He is one of the inventors of Microsoft Pinyin, and once worked with Microsoft headquarter in Seattle due to his contribution to Microsoft Pinyin. He is currently the deputy Director of the Computing Department, Harbin Institute of Technology, and Project Leader with the Artificial Intelligence Research Center, Pengcheng Laboratory, Shenzhen. His research interests include cybersecurity and artificial intelligence.

Zoe L. Jiang (Member, IEEE) received the Ph.D. degree in computer science from The University of Hong Kong, Hong Kong, in 2010.

She is currently an Associate Professor with the School of Computer Science and Technology, Harbin Institute of Technology, Shenzhen, China. Her research interests include secure multiparty computation, homomorphic encryption, and cloud security.

Weizhe Zhang (Senior Member, IEEE) received the Ph.D. degree from Harbin Institute of Technology, Harbin, China, in 2006.

He is currently a Professor with the School of Computer Science and Technology, Harbin Institute of Technology, Shenzhen, China, and Director of the Department of New Networks, Pengcheng Laboratory, Shenzhen. He has authored more than 130 academic papers in journals, books, and conference proceedings. His research interests include cyberspace security, cloud computing, and high-performance computing.

Dr. Zhang is a Lifetime Member of ACM.

Yang Liu received the B.Eng. degree in computer science from the Ocean University of China, Qingdao, China, in 2010, the M.Sc. degree in software engineering from Peking University, Beijing, China, in 2013, and the D.Phil. (Ph.D.) degree in computer science from the University of Oxford, Oxford, U.K., in 2018, advised by Prof. Andrew Simpson.

He is currently an Assistant Professor with the School of Computer Science and Technology, Harbin Institute of Technology, Shenzhen, China. His research interests include security and privacy problems, and in particular, the privacy issues related to mobile and IoT devices.

Mamoun Alazab (Fellow, IEEE) received the Ph.D. degree in computer science from the School of Science, Information Technology and Engineering, Federation University of Australia, Mount Helen, VIC, Australia, in 2012.

He is currently an Associate Professor with the College of Engineering, IT and Environment, Charles Darwin University, Casuarina, NT, Australia. He is also a Cyber Security Researcher and a Practitioner with industry and academic experience. He works closely with government and industry on many projects, including the Northern Territory (NT) Department of Information and Corporate Services, IBM, Trend Micro, the Australian Federal Police (AFP), etc. His research interests is multidisciplinary that include cyber security and digital forensics of computer systems with a focus on cybercrime detection and prevention, including cyber terrorism and cyber warfare.

Dr. Alazab is the Founder and the Chair of the IEEE Northern Territory Subsection Detection and Prevention.