Advancing Privacy and Accuracy with Federated Learning and Homomorphic Encryption

Tuy Nguyen 1

1Northern Arizona University

October 31, 2023

Abstract

In this paper, we present an integrated framework that combines Federated Learning (FL) with Homomorphic Encryption (HE) using the Artificial Intelligence (AI) models and the Cheon-Kim-Kim-Song (CKKS) algorithm to address the challenges of privacy and accuracy. FL facilitates collaborative training of Convolutional Neural Network (CNN) and Multi-Layer Perceptron (MLP) models across decentralized devices, allowing for data privacy preservation without sharing raw data. The integration of the CKKS algorithm for HE ensures secure computation on encrypted data during the FL process. Our experimental results on three diverse datasets demonstrate the efficacy of this approach, achieving an impressive highest average accuracy of 97.3%. Additionally, the CKKS algorithm is used to achieve efficient computation, making it a promising solution for privacy-conscious machine learning applications, and paving the way for practical deployment in various real-world scenarios, thereby revolutionizing the landscape of privacy-preserving machine learning.
Advancing Privacy and Accuracy with Federated Learning and Homomorphic Encryption

Quoc Bao Phan, Student Member, IEEE, Dinh C. Nguyen, Member, IEEE,
and Tuy Tan Nguyen, Member, IEEE

Abstract—In this paper, we present an integrated framework that combines Federated Learning (FL) with Homomorphic Encryption (HE) using the Artificial Intelligence (AI) models and the Cheon-Kim-Kim-Song (CKKS) algorithm to address the challenges of privacy and accuracy. FL facilitates collaborative training of Convolutional Neural Network (CNN) and Multi-Layer Perceptron (MLP) models across decentralized devices, allowing for data privacy preservation without sharing raw data. The integration of the CKKS algorithm for HE ensures secure computation on encrypted data during the FL process. Our experimental results on three diverse datasets demonstrate the efficacy of this approach, achieving an impressive highest average accuracy of 97.3%. Additionally, the CKKS algorithm is used to achieve efficient computation, making it a promising solution for privacy-conscious machine learning applications, and paving the way for practical deployment in various real-world scenarios, thereby revolutionizing the landscape of privacy-preserving machine learning.

Index Terms—Federated learning, Artificial neural network, Homomorphic Encryption, CKKS.

I. INTRODUCTION

FEDERATED Learning (FL) has emerged as a promising paradigm for training machine learning models in a decentralized manner, harnessing the collective intelligence of multiple devices or entities without compromising data privacy. By enabling local model training on user devices and sharing only model updates with a central server, FL has addressed significant privacy concerns while facilitating knowledge sharing across a distributed network. This distributed approach has proven particularly valuable in scenarios where data privacy and ownership are paramount, such as in healthcare, finance, and Internet of Things (IoT) applications.

While FL offers substantial advantages in preserving data privacy, concerns still arise when clients share model updates containing sensitive information. As models are iteratively updated based on local data, privacy breaches may occur when the aggregated updates reveal patterns or information about individual clients. To mitigate this privacy challenge, in recent years, homomorphic encryption (HE) has gained attention as a potential solution. HE allows computations to be performed on encrypted data without the need for decryption, providing an added layer of confidentiality during the FL process.

This paper explores the exciting synergy between FL and HE to achieve a robust framework for privacy-preserving machine learning. By combining the strengths of both technologies, we aim to enhance the privacy and security of FL, allowing it to flourish in diverse applications ranging from healthcare and finance to IoT and more.

In this work, we investigate the challenges, benefits, and potential trade-offs of this hybrid approach. One key challenge lies in efficiently encrypting and decrypting model updates and gradients during the FL process to maintain model accuracy while ensuring data privacy. Additionally, we explore the computational overhead introduced by HE and evaluate its impact on the overall FL system’s performance and convergence rates.

By thoroughly examining the integration of FL with HE, we aim to pave the way for a comprehensive understanding of its applicability and efficiency in privacy-preserving machine learning. Our research contributes to the growing body of knowledge on decentralized machine learning methods, addressing the critical need for data privacy while fostering collaborative model training across distributed networks.

Furthermore, we showcase experimental results on real-world datasets to demonstrate the effectiveness and performance of our proposed method. The experiments aim to validate the feasibility and practicality of the FL and HE hybrid approach, highlighting its potential for secure and privacy-preserving machine learning in practical scenarios. Through this study, we hope to provide valuable insights and guidelines for researchers interested in leveraging the powerful combination of FL and HE for privacy-preserving machine learning. Ultimately, our work contributes to the advancement of decentralized machine learning techniques and strengthens the foundation for secure and privacy-conscious AI applications.

The remainder of this paper is structured as follows: Section 2 provides an overview of related research on FL and HE. Section 3 details the technical aspects of combining these two technologies, discussing encryption schemes, secure aggregation, and communication protocols. In Section 4, we present our experimental setup and system implementation, followed by a discussion of the results in Section 5. Finally, we conclude the paper in Section 6, summarizing our findings in this dynamic field.
II. RELATED WORKS

In recent years, the rise of edge computing has attracted increasing attention from both industry and academia, leading to significant advancements in the field [1]–[5]. Edge nodes, situated between central servers and mobile devices, play a crucial role in collecting data from end devices and returning immediate results after analysis. However, the transmission of data in edge computing introduces privacy risks, prompting researchers to focus on privacy preservation in this context [3], [6]–[8]. In particular, the Alwarafy team conducted a survey on security and privacy issues in edge computing-assisted IoT environments [9].

One effective method for addressing privacy concerns in edge computing is FL [10]–[12]. FL allows individual devices to train local models on their respective data sets and transmit only the model parameters to update the global model. While this decentralized approach keeps individual data localized, the communication process still presents a potential vulnerability. Adversaries may exploit updated parameters to infer original information. Researchers have observed that a small portion of gradients may reveal information on local data, and gradients can be proportional to the original data [13].

To enhance privacy in FL, two widely adopted techniques are Secure Multiparty Computing (SMC) and Differential Privacy (DP) [14]. DP involves adding random noise to updated parameters to preserve privacy [15]–[17]. For example, Sun et al. [18] proposed a double disturbance localized DP (DDLDP) algorithm to disrupt edge device location information and upload sensing data to the blockchain via edge nodes. Similarly, Yin et al. [19] introduced a location data collection method that satisfies local DP requirements. The SMC technique encompasses various cryptographic approaches, including HE, secret sharing, and pairwise masking. Among these, HE has gained prominence by enabling complex mathematical operations on encrypted data without the need for decryption [20]. One of the notable HE schemes for privacy-preserving machine learning is the cryptography for HE applications over the approximate number system (CKKS). With CKKS, data can be encrypted in such a way that mathematical operations can be performed on the ciphertext directly, preserving the privacy of sensitive information throughout the computation process [21], [22].

Efficiency and effectiveness play a pivotal role in the successful deployment of privacy-preserving systems on a large scale. In this context, Li et al. [23] introduced a privacy-preserving FL framework named chain-PPFL, which leverages the chained SMC technique. By organizing participants into a chained structure and incorporating unique tokens as masks to each user’s output, chain-PPFL achieves efficient aggregation of model updates on the central server. This innovative approach significantly reduces the communication and computation complexities compared to conventional SMC-based schemes, making it more suitable for deployment on resource-constrained devices.

In light of the importance of security, this work prioritizes the utilization of the HE algorithm and proposes an enhancement to the CKKS algorithm to reduce computation time. By striking a balance between accuracy and latency in the FL-based edge computing environment, the proposed approach aims to improve privacy preservation and system efficiency simultaneously.

III. PRELIMINARY THEORY

A. Federated Learning

The concept of FL was first introduced by Google researchers in a seminal paper in 2015 [24]. Since then, it has gained significant attention and has been adopted and further developed by various organizations and researchers worldwide. The rise of mobile devices and the increasing need for privacy-conscious machine learning applications have contributed to the growing interest in this field.

The FL process, which is shown in Figure 1, involves three main components: the central server, the local devices, and the FL algorithm.

- Central server: initiates the training by distributing an initial model to the local devices. After receiving model updates from these devices, it aggregates them to create an improved global model. The central server is responsible for managing the communication between the local devices and ensuring the integrity of the FL process.
- Local devices: such as smartphones, tablets, or IoT devices, act as participants in the FL process. Each device holds its own local data and contributes to the model training without sharing the raw data with the central server or other devices. The local devices perform model training on their data using the received global model and then send encrypted model updates back to the central server. This distributed approach ensures data privacy and minimizes communication overhead.

B. Homomorphic Encryption

HE is a revolutionary cryptographic technique that allows computations on encrypted data without requiring decryption. The concept of HE was first introduced in 1978 by Rivest, Adleman, and Dertouzos [25], and it laid the foundation for
secure computation while preserving data privacy. The primary goal of HE is to enable authorized parties to process sensitive information in an encrypted form, ensuring that data remains confidential throughout the computation process.

The CKKS algorithm, introduced in 2017 by Cheon, Kim, Kim, and Song [21], signifies a remarkable advancement in the field of HE. This algorithm is specifically designed to handle computations on complex numbers efficiently, making it well-suited for various real-world applications that involve processing real-valued data. One of the notable strengths of the CKKS algorithm lies in its utilization of the mathematical concept of the Ring Learning with Errors (RLWE) problem, operating within the ring of polynomials modulo a prime number \( q \). This choice of the RLWE problem contributes to the security and efficiency of the CKKS scheme.

Efficiency is a crucial aspect of the CKKS algorithm, as it strives to strike a balance between precision and computation speed. By employing approximate operations, the CKKS algorithm optimizes computation time while still retaining an acceptable level of accuracy. This precision trade-off becomes particularly valuable in scenarios where strict accuracy is not the primary concern, and faster computations are preferred. As a result, the CKKS algorithm demonstrates improved performance in various real-world settings, such as machine learning tasks, secure multi-party computations, and privacy-preserving data analysis.

IV. SYSTEM DESIGN AND IMPLEMENTATION

A. Secure Computation Using CKKS

The CKKS scheme is utilized in FL, a privacy-preserving approach where multiple client devices \( (C_1, C_2, \ldots, C_n) \) hold private data and collaborate with a central server \( (S) \) to train a machine learning model without sharing raw data. The CKKS scheme operates on a cyclotomic polynomial ring \( \mathbb{R} = \mathbb{Z}[\chi]/(\chi^N + 1) \), where \( N \) is the polynomial degree, a power of two. With a Modulus \( q \) and a scaling factor \( \gamma = 2^p \), where \( p \) determines the level of precision.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Polynomial Degree ( (N) )</td>
<td>The number of coefficients in the encrypted ciphertext</td>
</tr>
<tr>
<td>Modulus ( (q) )</td>
<td>The size of the ciphertext elements</td>
</tr>
<tr>
<td>Base scale ( (\gamma) &amp; (\Delta) )</td>
<td>Used to encode and represent real numbers</td>
</tr>
<tr>
<td>Security Level</td>
<td>A measure of how resistant the scheme is to attacks</td>
</tr>
<tr>
<td>Number of Levels ( (L) )</td>
<td>The multiplicative depth of computations</td>
</tr>
<tr>
<td>Precision Parameters</td>
<td>The trade-off between the range and accuracy of encrypted computations</td>
</tr>
</tbody>
</table>

The parameters of the CKKS scheme for FL are shown in Table 1. and the CKKS’s workflow is shown in Figure 2. The core steps of the CKKS algorithm for FL are as follows:

1) Setup\( (1^1) \):
   - Initializes the CKKS encryption scheme with a security parameter \( \lambda \).
   - Outputs the ring dimension \( N \), which is a power of two, and sets up small distributions \( \chi_{\text{key}}, \chi_{\text{err}} \), and \( \chi_{\text{enc}} \) over the ring \( \mathbb{R} \) for secret key, error, and encryption respectively.

2) KeyGen:
   - The central server \( (S) \) generates the public and secret keys for the CKKS encryption scheme.
   - The secret key, \( sk \), is a random polynomial \( s \) sampled from the secret distribution \( \chi_{\text{key}} \).
   - The public key, \( pk \), consists of a random polynomial \( p \) sampled from the ring \( \mathbb{R}_Q \) (for a given ciphertext modulus level \( l \)) and an error polynomial \( e \) sampled from the error distribution \( \chi_{\text{err}} \).
   - The public key \( pk \) is shared with the client devices \( (C) \) for encryption.

3) Encrypt \( (pk, m) \):
   - The plaintext \( m \) is first scaled and encoded into a polynomial \( \hat{m} \) using the Encode function with the scaling factor \( \gamma \).
   - The client samples random polynomials \( v \) from the encryption distribution \( \chi_{\text{enc}} \) and error polynomials \( e_0, e_1 \) from \( \chi_{\text{err}} \).
   - The ciphertext \( ct \) is computed as \( \hat{m} = v \cdot pk + (e_0 + e_1) \mod Q_l \), where \( Q_l \) is the ciphertext modulus for level \( l \).

4) Decrypt \( (sk, ct) \):
   - The decryption process is performed as \( \hat{m} = c_0 + c_1 \cdot s \mod Q_l \), where \( c_0 = (c_0, c_1) \).

5) Homomorphic operations:
   - Homomorphic addition: The central server can add two encrypted ciphertexts \( ct_1 \) and \( ct_2 \) as \( \hat{ct}_{\text{add}} = ct_1 + ct_2 \mod Q_l \).
   - Homomorphic multiplication: The central server can multiply two encrypted ciphertexts \( ct_1 \) and \( ct_2 \) as \( \hat{ct}_{\text{mul}} = ct_1 \times ct_2 \mod Q_l \).

6) Rescale \( (ct, \Delta') \):
   - To manage the noise accumulation during homomorphic computations, a rescaling step is performed using a rescaling factor \( \Delta' \). Rescale takes a ciphertext \( ct \) and rescales it as \( ct' \leftarrow \gamma(-\Delta') \times ct \mod Q_l - \Delta' \).

B. Federated Learning System

In this study, we introduce a secure learning approach with the system diagram depicted in Figure 1. The operation of this system includes the following steps:

1) Initialization: The process begins with the central server initializing an initial model and sending it to local devices to start the training. However, before sending the model and the dataset, the central server encrypts it using CKKS. This ensures that the model is not disclosed in plaintext form, maintaining its security during transmission and the pseudo-code is shown in Algorithm 1.

2) Local model training with HE: At the local devices, the data and model are received and ready to train the local model. This allows the devices to perform mathematical
operations on the encrypted data. While the idea is shown in Algorithm 2.

3) Model update encryption: After completing the model training, the local devices encrypt the updated model before sending it back to the central server. This encryption process ensures that sensitive information within the model is not disclosed during transmission.

4) Aggregation and global model update: At the central server, the locally encrypted models are decrypted and combined using cryptographic operations such as addition and averaging. This process generates an updated global model, incorporating knowledge from all local devices without revealing their actual data.

5) Iterative training: The training process is repeated iteratively by reiterating the above steps to create increasingly improved global model versions through contributions from local devices.

Additionally, the initialization parameters for the client side are shown in Table II. The datasets used for experimentation include Iris, Mnist, and Cifar, with the models employed being MLP and CNN.

<table>
<thead>
<tr>
<th>Notation</th>
<th>Definition</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>The number of clients</td>
<td>20</td>
</tr>
<tr>
<td>F</td>
<td>The fraction of local models</td>
<td>0.1</td>
</tr>
<tr>
<td>B</td>
<td>Batch size</td>
<td>32</td>
</tr>
<tr>
<td>lr</td>
<td>Learning rate</td>
<td>0.001</td>
</tr>
<tr>
<td>D</td>
<td>Dropout rate</td>
<td>0.1</td>
</tr>
<tr>
<td>E</td>
<td>Number of local epoch</td>
<td>5</td>
</tr>
</tbody>
</table>

V. RESULT AND COMPARISON

A. System Experimental Setup

1) Computer Specifications: The experiments were conducted on a Dell Inc. Precision 3660 desktop system running Microsoft Windows 10 Education. The system featured a 12th Gen Intel(R) Core(TM) i9-12900 processor with 16 cores and 24 logical processors, providing a base clock speed of 2400 Mhz. It was equipped with 64.0 GB of installed physical memory (RAM) to handle large datasets and complex computations efficiently.

2) Dataset Descriptions:

- Mnist: The Mnist dataset consists of 28x28 grayscale images of handwritten digits (0 to 9) along with their labels. It is a widely used benchmark in computer vision research. The dataset was collected from a combination of two sources: NIST (National Institute of Standards and Technology) and CENPARMI (Center for Pattern Recognition and Machine Intelligence). The dataset has been extensively used for testing various machine learning and deep learning models in image classification tasks.

- Cifar: The Cifar dataset includes 60,000 32x32 color images distributed across 10 classes. It was collected by the Canadian Institute for Advanced Research. Each class contains 6,000 images. The dataset serves as a challenging benchmark for evaluating image recognition models due to its diverse and complex images in various categories.
• Iris: The Iris dataset contains 150 samples of iris flowers, each with four features: sepal length, sepal width, petal length, and petal width. The dataset was collected by biologist Edgar Anderson [26] and is widely used in machine learning and pattern recognition. The data were gathered from three species of iris flowers: Iris setosa, Iris versicolor, and Iris virginica.

3) Data Preprocessing: For all datasets, careful data preprocessing was applied to ensure consistency and comparability across experiments. The Mnist and Cifar datasets were publicly available and preprocessed to normalize pixel values and apply data augmentation techniques such as random rotations and flips [13], [27]. For the Iris dataset, no additional preprocessing was required.

B. Evaluation Metric

This paper is focused on three main metrics, as flow:

• Encrypt and decrypt time: These metrics quantify the average time taken for encrypting and decrypting model updates during each communication round in FL-HE. These times are measured in seconds.

• Aggregation time: The aggregation time metric represents the average time taken to aggregate encrypted model updates from multiple client devices on the central server during a communication round in FL-HE. It includes the computation time for securely combining the encrypted model updates.

• Accuracy: The accuracy metric evaluates the quality of the trained models in FL-HE. It is defined as the ratio of correctly predicted samples to the total number of samples in the validation dataset. Accuracy is calculated as follows:

\[
ACC = \frac{\text{Number of correctly predicted}}{\text{Total number of samples}} \times 100
\]

VI. EXPERIMENTAL RESULTS

The evaluation of the FL-HE training losses on the Iris, Mnist, and Cifar datasets showcases its effectiveness in ensuring privacy preservation within edge computing environments:

• On the Iris dataset, the FL-HE method yielded impressive outcomes, achieving a remarkable reduction in training loss from 1.16 to 0.09 over 100 epochs. This considerable decrease in training loss indicates the capability of FL-HE to protect sensitive information effectively and maintain data privacy, even in the context of a relatively smaller dataset like Iris.

• Moving to the more complex Mnist dataset, FL-HE continued to exhibit excellent performance in privacy preservation. With a reduction in training loss from 0.81 to 0.15, the FL-HE method demonstrated its adaptability and robustness in safeguarding data privacy in scenarios involving larger and more diverse datasets.

• Even on the challenging Cifar dataset, known for its intricacies and higher dimensionality, FL-HE proved its ability to mitigate privacy risks. By reducing the training loss from 1.69 to 0.25, FL-HE showcased its strength in preserving privacy while training on sophisticated and resource-intensive datasets.

The results presented in Table III provide a comprehensive evaluation of the effectiveness of the CKKS algorithm with three different parameter sets. The analysis focuses on encryption time, aggregation time, decryption time, and average accuracy across the parameter variations.

For the parameter set \( N = 4096, \gamma = 2^{30} \) with coefficient mod bit sizes [60, 40, 40, 60], the CKKS algorithm demonstrated impressive performance. The encryption time was 0.01827 seconds, while the aggregation time took 1.59841 seconds, and the decryption time was 0.00977 seconds. Moreover, the average accuracy achieved was 91.6%, showcasing the algorithm’s ability to efficiently handle computations.

When using the parameter set \( N = 8192, \gamma = 2^{40} \) with coefficient mod bit sizes [60, 40, 40, 60], the CKKS algorithm continued to perform effectively. The encryption time increased to 0.03181 seconds, aggregation time rose to 3.77594 seconds, and decryption time reached 0.01287 seconds. However, the trade-off was worthwhile, as the average accuracy significantly improved to 95.9%, indicating better precision with slightly increased computational overhead.

For the parameter set \( N = 16384, \gamma = 2^{40} \) with coefficient mod bit sizes [60, 40, 40, 60], the CKKS algorithm demonstrated its ability to handle more substantial data sizes. The encryption time further increased to 0.08712 seconds, and the aggregation time reached 4.91896 seconds, while the decryption time increased to 0.06773 seconds. However, this was compensated by a remarkable average accuracy of 97.3%, showcasing the algorithm’s efficiency in dealing with more complex datasets.

The final evaluation in Figure 4 highlights the impressive performance of both the MLP and CNN models on the three datasets, Iris, Mnist, and Cifar. The accuracy results indicate that both models achieved substantial improvements in accuracy as the number of epochs increased from 10 to 100.

For the MLP model, on the Iris dataset, the accuracy increased significantly from 61.5% to 93.1% after 100 epochs. Similarly, on the Mnist dataset, the accuracy saw a remarkable
improvement from 58.9% to 92.5% after the same number of epochs. On the more complex Cifar dataset, the MLP model demonstrated commendable progress, with accuracy rising from 45.5% to 89.4%.

On the other hand, the CNN model showcased even more impressive results. On the Iris dataset, the accuracy surged from 73.5% to an outstanding 97.7% after 100 epochs. For the Mnist dataset, the CNN model achieved a remarkable leap in accuracy, increasing from 68.4% to an impressive 96.4%. Similarly, on the challenging Cifar dataset, the CNN model’s accuracy showed significant growth, increasing from 55.4% to an excellent 91.6%.

These accuracy results indicate that both the MLP and CNN models are highly effective in capturing complex patterns and representations from the data, resulting in significant improvements in performance as the number of epochs increased. The substantial increase in accuracy demonstrates the models’ ability to learn and generalize better with more training iterations.

VII. CONCLUSION

The integrated framework presented in this paper, combining FL with HE using the CKKS algorithm, provides an effective solution to the challenges of privacy and accuracy in machine learning. By enabling collaborative model training across decentralized devices while preserving data privacy through FL, and incorporating secure computation on encrypted data via HE and CKKS, the approach achieves an impressive highest average accuracy of 97.3% across diverse datasets. This promising solution holds significant potential for privacy-conscious machine learning applications, empowering the development of trustworthy and secure AI systems in data-driven domains.

REFERENCES


