Iraqi Journal for Computer Science and Mathematics

Journal Homepage: http://journal.esj.edu.iq/index.php/IJCM
e-ISSN: 2788-7421 p-ISSN: 2958-0544

# Healthcare Privacy-Preserving Federated Transfer Learning using CKKS-Based Homomorphic Encryption and PYHFEL Tool

**Anmar A. Al-Janabi [1,2]*, Sufyan Al-Janabi [2], Belal Al-Khateeb[2]**

[1]University of Technology – Iraq, Baghdad, Iraq,
Computer Science Department, University of Technology – Iraq, Baghdad, 10066, IRAQ
[2] University of Anbar, Ramadi, Iraq,
College of Computer Science & Information Technology, Al Ramadi, Anbar, 31001, IRAQ

*Corresponding Author: Anmar A. Al-Janabi

DOI: https://doi.org/10.52866/ijcsm.2024.05.03.029
Received March 2024; Accepted May 2024; Available online August 2024

**ABSTRACT:** Digitization of healthcare data has shown an urgent necessity to deal with privacy concerns within the field of deep learning for healthcare organizations. A promising approach is federated transfer learning, enabling medical institutions to train deep learning models collaboratively through sharing model parameters rather than raw data. The objective of this research is to improve the current privacy-preserving federated transfer learning systems that use medical data by implementing homomorphic encryption utilizing PYthon for Homomorphic Encryption Libraries (PYFHEL). The study leverages a federated transfer learning model to classify cardiac arrhythmia. The procedure begins by converting raw Electrocardiogram (ECG) scans into 2-D ECG images. Then, these images are split and fed into the local models for extracting features and complex patterns through a finetuned ResNet50V2 pre-trained model. Optimization techniques, including real-time augmentation and balancing, are also applied to maximize model performance. Deep learning models can be vulnerable to privacy attacks that aim to access sensitive data. By encrypting only model parameters, the Cheon-Kim-Kim-Song (CKKS) homomorphic scheme protects deep learning models from adversary attacks and prevents sensitive raw data sharing. The aggregator uses a secure federated averaging method that averages encrypted parameters to provide a global model protecting users' privacy. The system achieved an accuracy rate of 84.49% when evaluated using the MIT-BIH arrhythmia dataset. Furthermore, other comprehensive performance metrics were computed to gain deeper insights, including a precision of 72.84%, recall of 51.88%, and an F1-score of 55.13%, reflecting a better understanding of the adopted framework. Our findings indicate that employing the CKKS encryption scheme in a federated environment with transfer cutting-edge technology achieves relatively high accuracy but at the cost of other performance metrics, which is lower in the encrypted settings when compared to the plain one, an acceptable trade-off to ensure data privacy through encryption with achieving an optimal model performance.

**Keywords:** Electrocardiogram, Federated transfer learning, Homomorphic encryption, CKKS scheme, Data privacy.

## 1. INTRODUCTION

In recent years, healthcare data analytics has gained popularity as more data becomes available from clinical institutions, patients, and pharmaceutical industries, which introduces a new possibility of computational approaches revealing data-driven insights and potentially improving services offered by healthcare systems [1], [2].

The complex nature of the healthcare system and its related operations has led to fragmented data on the system. For example, a hospital can only access its own medically private records that belong to its patients. Usually, these files are extremely sensitive and susceptible because of the Private Health Information (PHI) that belongs to people. Strict regulations and legislation were enacted to control accessibility and data analysis, as in the Data Protection Regulation (GDPR) [3] and the Health Insurance Portability and Accountability Act (HIPAA) [4]. These present a challenge for state-of-the-art Machine Learning (ML) and its subsidiary Deep Learning (DL) [5] technologies, which require a massive amount of data for training.

The amount and variety of training data play a crucial role in the results of DL within various medical applications [6], [7]. Labeling data is important, as some organizations might possess unlabeled data while others have very limited data. The labeling process within the medical domain is expensive and requires human expertise. At the same time, due to the regulatory constraints of medical data, it is impractical to collect and share private patient information in a central data lake or server. Enabling the development of effective and accurate DL models for applications with limited samples, features, or labels while complying with data privacy and security regulations is a difficult challenge [8].

To address this challenge, Google introduced a Federated Learning (FL) system in 2016 [9], wherein a global model is updated through the collaboration of distributed participants while keeping their data locally stored. Their framework necessitates all contributors to share the same feature space. These approaches are only applicable within the context of data possessing either common features or common samples under federation.

The focus has shifted towards Transfer Learning (TL) as an approach to enhance the performance of a target model by utilizing the pre-trained source model. Typically, the source and target datasets exhibit similarities in domains and modalities. Challenges in Electrocardiogram (ECG) classification, such as biased classification and small-sized datasets, are evident. With limited training data, employing deep learning algorithms to enhance classification model performance becomes essential to prevent model overfitting. Training the target ECG classification models through fine-tuning with the large-scale dataset of pre-trained source models could help alleviate model overfitting and biased classification [10].

This study employs ResNet50V2, a commonly used pre-trained TL model architecture. After converting ECG signals into 2-D grayscale images, the pre-trained model can analyze them to aid physicians in diagnosing cardiac arrhythmia. Given that hospitals own the data and adhere to strict privacy regulations that forbid the disclosure of any information beyond the model results, Homomorphic Encryption (HE) emerges as a viable approach. It encrypts intermediate parameters while allowing mathematical computations to be performed [11]. This methodology is capable of safeguarding the DL model from adversarial attacks.

This research introduces an innovative framework for ECG data classification, leveraging TL for feature extraction, FL for collaborative training, and HE implemented through PYthon For Homomorphic Encryption Libraries (PYFHEL) [12] for secure aggregation of model updates. In addition to enhancing patients' privacy protection, the integration of these state-of-the-art technologies improves the efficiency and accuracy of ECG data analysis.

The main contributions of this work are as follows:

- Implementation and evaluation, to the best of our knowledge, of the first Privacy-Preserving Federated Transfer Learning (PPFTL) framework for 2-D ECG arrhythmia classification.
- Integration of transfer knowledge to reduce the substantial margin of computation run times between non-encrypted and encrypted versions of the proposed system, leading to a more efficient computational process and significantly improving the overall runtime performance.
- Development of an end-to-end privacy-preserving healthcare system demonstrating that sensitive ECG data can be securely shared among multiple participants without compromising patient data.
- Highlighting how healthcare agencies can collaborate using Federated Transfer Learning (FTL) and a data-protected shared model.
- Use of a real-world ECG dataset to verify the validity of the proposed system.

The remainder of the study is organized as follows: Section 2 provides a comprehensive review of existing research and the current state of medical privacy-preserving FL, highlighting the significance of the suggested framework and its contributions to the field. Section 3 elucidates the preliminaries of HE, including the CKKS scheme utilized, FL, and TL, delineating how it enhances privacy and security. The system model encompasses the architecture and essential processes, commencing from client initialization, the secure aggregation step at the server side, to the client-side decryption processes is described in Section 4. Section 5 discusses the dataset, preprocessing techniques, and evaluation results. The encrypted FTL scheme is compared with the plain one, alongside locally trained models and the computation costs of the proposed models. Section 6 extensively discusses the results. Finally, in Section 7, we conclude the study by summarizing our findings and contributions and addressing the implications of privacy-preserving FL implementations in the medical and healthcare domains.

## 2. RELATED WORK

In healthcare, ML models offer significant advantages, particularly when coupled with private medical data. These models are usually trained locally to safeguard the privacy of sensitive healthcare data. However, developing effective models without extensive, diverse datasets covering a wide range of health conditions can pose challenges. Previous studies have suggested that employing FL approaches may offer a viable solution to address this issue. For instance, studies [13]–[15] have explored the potential use of FL in the digital health domain. Specifically, it draws attention to the obstacles and concerns that must be addressed for FL to be successfully adopted and to secure disseminated and private biomedical data. The impact of FL is examined by multiple stakeholders, encompassing patients, doctors, and healthcare organizations.

The study in [6] presented a Privacy-Preserving Federated Learning (PPFL) method for brain tumor segmentation using the BraTS 2018 dataset. The proposed FL system is based on the client-server model and applies a federated averaging algorithm. The server coordinates the clients' local Stochastic Gradient Descent (SGD) updates and manages the global Deep Neural Network (DNN) model. In addition, the Differential Privacy (DP) technique is used to ensure the privacy of medical records. Experimental results show that the FL method can preserve sensitive patient information while delivering high accuracy. The study does not evaluate how privacy-preserving methods affect the accuracy and performance of the brain tumor segmentation model. Furthermore, it does not examine the system's scalability for more complex medical imaging applications or larger datasets.

In [16], the authors introduced the use of FL for multi-institutional collaboration in medical imaging, enabling DL modeling without sharing sensitive patient data. They evaluate and compare two alternative forms of collaborative training, namely Incremental Institutional Learning (IIL) and Cyclic Institutional Learning (CIIL), with FL, using the same BraTS 2018 dataset previously utilized in [6]. Both IIL and CIIL involve sequentially training a shared model, with CIIL incorporating a cycling loop across institutions. Results indicate that FL produces comparable dice scores to models trained with shared data. FL outperforms IIL and CIIL due to their susceptibility to catastrophic forgetting and increased complexity. Furthermore, the authors mention DP as a means to prevent data leakage during model updates, deferring it to future research.

Encryption methods, such as HE, play a crucial role in safeguarding sensitive medical data. In [17], the authors introduced a novel approach to bolster the security of E-healthcare systems by integrating secure Multiparty Computation (MPC) with the Paillier encryption scheme. This methodology guarantees the confidentiality and security of critical patient information. Moreover, there is a growing interest in leveraging IoT-enabled health equipment to provide accurate medical data. The proposed methodology could also find applications in E-auctions and E-voting systems. However, the study lacks comprehensive details on the scalability of the suggested technique, particularly when dealing with a large number of patients.

The studies conducted by Wibawa et al. [18], [19] introduced PPFL for medical data, utilizing HE to safeguard sensitive data against privacy attacks. Employing a secure MPC protocol further fortifies the model against adversaries. The proposed method's model accuracy was evaluated using a real-world medical dataset of COVID-19 radiography images with two classifications, COVID and Normal, achieving accuracy scores above 80% with both encrypted and plaintext data. However, the implementation of HE leads to a significant increase in the system's running time; for two clients, the increase is approximately 629.37%, for three clients, it's about 690.91%, and for seven clients, it rises to approximately 872.20%, thereby limiting its practical application in real-world scenarios.

Bocu et al. [20] developed a HE scheme integrated with a heart rate-based personal health information system. The results indicate that the described method met the criteria for secure data processing for 500 patients with anticipated storage and network challenges. The study needs to address the potential challenges and limitations of combining the system with other wearable medical devices and data sources.

Authors in [21] proposed a unique approach for efficient cloud computing Fully Homomorphic Encryption (FHE). The suggested solution secures encrypted data processing with twin-key encryption and magic number fragmentation. Cryptanalytic attacks are used to evaluate the efficiency of the suggested method, and a cognitive smart city scenario proves its applicability. When tested against brute-force attacks, the system proved to be highly resilient. The study does not discuss the limitations of the presented FHE scheme, which utilizes twin-key encryption and magic number fragmentation. Also, there are no explicit details about the dataset used in the study.

Ali et al. [22] developed an innovative DL model with a searchable blockchain system and HE, allowing users to gain access to distributed data via search securely. They evaluated and compared the suggested access control mechanisms on an Internet of Things (IoT) dataset to industry standards. The stated approaches are implemented using Hyperledger smart contracts. The proposed method significantly improves security, anonymity, and user behavior monitoring in comparison to baseline approaches, hence improving the efficiency of a blockchain-based IoT system. The study acknowledges several limitations, including the need for a privacy-preserving technique to prevent unauthorized access, issues with data transfer and integrity across IoT networks, and the need for a responsive security mechanism to distinguish between normal and attack instances in IoT settings.

In [23], the authors developed a deep 1-Dimensional Convolutional Neural Network (1-D CNN) based heartbeat classification model. This model classifies five AAMI EC57 arrhythmias. The authors additionally propose a method for transforming the knowledge acquired through this task to the challenge of classifying Myocardial Infarction (MI). The proposed method achieves a classification accuracy of 93.4% in arrhythmia and 95.9% in MI.

Gao et al. [24] introduced Heterogeneous Federated Transfer Learning (HFTL), which uses TL to handle different feature spaces. They devised a privacy-preserving transfer learning approach to eliminate covariate shifts in homogeneous feature spaces and combine heterogeneous feature spaces from different data owners. Two secure multi-party learning protocol variants based on the HE and Secret Sharing (SS) techniques demonstrate the HFTL's security, efficiency, and scalability over five benchmark datasets. Nevertheless, a thorough investigation is required to assess the scalability of the suggested approach in scenarios involving a more significant number of clients.

The study [8] introduced FTL to improve statistical modeling in data federations. The research also presents novel methods for Two-Party Computation (2PC) with Neural Network (NN) within the FTL framework, integrating

additively HE and SS via beaver triples to achieve almost lossless accuracy with little NN adjustments. This research applies an FTL system to MNIST, CIFAR-10, and CIFAR-100 datasets to demonstrate its effectiveness. The study found that the FTL framework outperforms conventional approaches while preserving data privacy.

The majority of previous work refrained from incorporating numerous deep layers, instead focusing on utilizing simple models coupled with privacy-preserving techniques such as DP, MPC, and HE. This choice stems from the noise generated by deep convolutional layers. Convolutional layers heavily rely on matrix computations, leading to an exponential increase in noise through multiplication processes in HE. To address this issue, the proposed privacy approach employs TL, where the weights of the deep model are frozen, particularly in the feature extraction part of the architecture, where most computationally intensive convolutions occur. Moreover, TL reduces computation costs across distributed models, as training for the suggested approach does not need to commence from scratch. The integration of FL, TL, and HE represents cutting-edge technologies that strike a balance between model performance, noise management, and data privacy.

## 3. PRELIMINARIES

In the following subsections, we present the fundamental theoretical foundations for this research. We start by exploring cryptographic methods such as CKKS. Subsequently, we delve into an analysis of healthcare data privacy regulations. Finally, we investigate Federated and Transfer Learning methods.

### 3.1 HOMOMORPHIC ENCRYPTION

Data encryption is commonly employed in both enterprise and individual environments to safeguard data while at rest or in transmission. However, traditional approaches pose a security vulnerability during computational operations, particularly in highly sensitive sectors such as healthcare.

In response to this issue, HE emerged, allowing mathematical operations to be performed directly on encrypted data without the need for decryption beforehand. The decryption results remain encrypted and yield the same or nearly identical results consistently. By enabling secure data processing without revealing actual data, HE meets the stringent privacy standards of today's digitally interconnected world, aligning with the strict provisions of GDPR and HIPAA. If we define encryption as $Enc$, decryption as $Dec$, $\odot$ representing homomorphic addition or multiplication operations over ciphertext, and $f()$ as a function applied to plaintext values x and y using encryption key pk. In that case, the property of HE can be stated as follows:

$$f(x,y) = Dec(Enc(\mathbf{pk}, x) \odot Enc(\mathbf{pk}, y)) \quad (1)$$

HE schemes facilitate outsourcing data storage and processing while preserving clients' privacy. They allow data encryption and outsourcing to commercial cloud platforms while the data remains encrypted. HE can be classified into several categories based on the operations they support [25]:

1. Partially Homomorphic Encryption (PHE) supports only one type of mathematical operation, either addition or multiplication, but not both infinitely. While these schemes are interesting in their own right, their functionality is limited.
2. Somewhat Homomorphic Encryption (SHE) enables both additive and multiplicative operations. However, each operation introduces error noise into the system, necessitating bounding the number of computations for accurate evaluation. Message decryption fails if the noise exceeds a certain threshold.
3. FHE enables arbitrary computations involving unlimited addition and multiplication operations. This scheme employs a bootstrapping strategy to manage and decrease noise, enabling continuous and accurate decryption.

This study utilizes an SHE scheme, which allows both multiplication and addition. Such operations are essential for aggregating the weights of DL models.

### 3.2 CHEON-KIM-KIM-SONG (CKKS) SCHEME

Cheon et al. [26] introduced the CKKS scheme, which is an approximative HE scheme featuring a tunable level of approximation error for secure data computation over real or complex numbers directly. This scheme operates on polynomials within a ring, and its security is based on the assumed hardness of the Ring Learning With Errors (Ring-LWE) problem. In machine or deep learning technologies, where the approximation of arithmetic computations is adequate for privacy preservation, CKKS is highly desirable. The following section provides a brief explanation of the scheme under consideration.

Given initialized parameters, where $n$ denotes a ring dimension, $q$ represents a prime modulus, and $\Delta$ signifies a scaling factor, the secret key **sk** is randomly selected from $R_2$, and the public key $\mathbf{pk} = (\mathbf{pk_1}, \mathbf{pk_2})$ is computed. The encoding process involves converting a vector of complex numbers, denoted as $\vec{\mathbf{v}} \in \mathbb{C}^{\frac{n}{2}}$, and a value $\Delta > 1$ into a single object, denoted as **a**, in the plaintext domain. This is achieved by applying the encoding function $\text{Encode}(\vec{\mathbf{v}}, \Delta) = \lfloor \Delta . \pi^{-1}(\vec{\mathbf{v}}) \rceil$. Encryption employs **pk** and an error distribution $\chi$ to produce ciphertext $\mathbf{CT} = (\mathbf{CT_1}, \mathbf{CT_2})$ Homomorphic evaluation, denoted as $\text{EvalAdd}(\mathbf{CT}^{(1)}, \mathbf{CT}^{(2)}) = (\mathbf{CT}_1^{(3)}, \mathbf{CT}_2^{(3)})$, adds polynomial components, whereas multiplication $\text{EvalMult}(\mathbf{CT}^{(1)}, \mathbf{CT}^{(2)})$ results in $\mathbf{CT}^{(3)} = (\mathbf{CT}_1^{(3)}, \mathbf{CT}_2^{(3)}, \mathbf{CT}_3^{(3)})$, which requires

relinearization to reduce dimensionality. Following, the rescaling process is performed after multiplication to reduce noise and provide a ciphertext that closely matches the input, using $\text{Rescale}(\mathbf{CT}, \Delta) = \frac{1}{\Lambda}.[\mathbf{CT_1}, \mathbf{CT_2}]_q$. $\mathbf{sk}$ is employed on the ciphertext to decrypt and get an approximate encoded plaintext $\acute{\mathbf{m}} = [\mathbf{CT_1} + \mathbf{CT_2}.\mathbf{sk}]_q$. Decoding is an inverse process to encoding, at which $\text{Decode}(\mathbf{a}, \Delta) = \pi\left(\frac{1}{\Delta}.\ \mathbf{a}\right)$ returns a vector of complex numbers.

PYFHEL stands as a user-friendly API that comprehensively supports CKKS and facilitates efficient management of security parameters to attain the necessary level of security. Moreover, when dealing with larger parameter sizes, PYFHEL demonstrates robust optimization for such implementations, striking a balance between security requirements and computational costs that render it suitable for encrypted computations. It also facilitates the development of intricate cryptographic operations and integrates seamlessly into the technology stack utilized to implement the suggested framework. Section 4.2 elaborates on the employed security parameters within the context and key generation processes.

### 3.3 REGULATORY ASPECTS OF PRIVACY IN HEALTHCARE DOMAIN

Trust and privacy represent critical components of modern healthcare systems that must remain uncompromised. Establishing trust among stakeholders, encompassing patients, healthcare workers, legislators, and vendors of technological solutions, poses a multifaceted and challenging task.

Given the personal nature of healthcare records, maintaining privacy and security is crucial. Various types of information must be protected due to the sensitive nature of healthcare records. This includes patients' names, social security numbers, addresses, birth dates, and bank account information. It also contains other details, such as information about a patient's physical and mental condition that provides insight into their current state of health. Data concerning healthcare services such as visits, diagnoses, prescribed medications, and utilized medical equipment are also sensitive. Lastly, information about healthcare facilities and healthcare professionals who administer medical treatment must be safeguarded [19].

GDPR and HIPAA are significant legislative measures that regulate data privacy within the European Union and the USA, respectively. They have a profound impact on healthcare data privacy regulations in both the public and private sectors, emphasizing individuals' rights to control the usage of their private data. The combination of FTL with the CKKS encryption scheme provides healthcare systems with exceptional data privacy and secure sharing capabilities. This integrated strategy enhances the security of healthcare systems and aligns them with GDPR and HIPAA standards, ensuring a patient-driven approach [27].

### 3.4 FEDERATED LEARNING

Federated learning is a decentralized framework for machine learning that facilitates collaboration among stakeholders, including healthcare and other data owners, to develop a shared model while maintaining local data privacy. Entities with stringent privacy policies favor this approach as it only transmits the model's parameters or gradients. In FL, participants can train a unified model on their respective local datasets, even if these datasets exhibit vastly different distributions of common features and occupy various feature spaces. The trained models can then be aggregated on a centralized server or collaboratively among participants to create a global model. FL is adaptable to various technical platforms and applicable in centralized, decentralized, or heterogeneous environments. It serves as an effective strategy for ensuring data privacy and security, particularly in healthcare, where data heterogeneity and privacy are crucial [19]. This efficacy stems from the localized nature of the data and the collaborative training approach inherent in FL.

Collecting medical data on a large scale poses challenges due to privacy, ethics, and security concerns, resulting in medical institutes gathering ECG datasets on a smaller scale. Additionally, obtaining a large, well-annotated dataset necessitates considerable professional expertise and a time-consuming process. TL emerges as a viable solution to address the aforementioned issues faced by DL, utilizing well-trained public domain models to transfer knowledge from source to destination domains [28].

### 3.5 TRANSFER LEARNING

TL is an ML technique that leverages prior knowledge acquired from one domain to enhance learning in another, albeit interconnected, domain. This strategy allows the utilization of patterns and insights from a broader source domain when the target domain lacks labeled data [29]. An effective method involves employing a pre-trained deep CNN for automatic feature extraction. The convolutional layers within these networks retain feature maps learned during training and possess knowledge of patterns present in the original dataset. The intermediate layers of deep neural networks are capable of generating pre-extracted features more effectively than hand-crafted features for feature extraction [30].

TL enables healthcare personnel to utilize pre-trained models on public medical data, such as X-ray scans, and fine-tune them for specific tasks, such as ECG analysis, for early disease detection or personalized treatment planning. The application of TL in cardiac healthcare demonstrates potential for improving the accuracy and efficiency of ECG

interpretation, as ECG signals can be complex to analyze and may exhibit significant variation among patients. This obviates the requirement for extensive labeled ECG data and expedites the development of cardiac diagnostic tools, thereby fostering healthcare innovation and efficiency.

## 4. SYSTEM MODEL

This section describes the proposed PPFTL approach for classifying 2-D grayscale images of ECG scans. The method utilizes a CKKS-based HE scheme and consists of two primary stages. In the initial stage, the client-side conducts local training on a Graphical Processing Unit (GPU), employing deep CNN architecture to extract feature maps from the ECG images in its local dataset. The second stage occurs on the server-side, where encrypted model weight matrices are aggregated.

The proposed PPFTL system overview, as depicted in Figure 1, ensures data privacy through essential steps and describes the algorithms that follow:

1. **Global Model Generation and Distribution:** The aggregator or central server initializes a global CNN model for the target task and distributes it to all participating clients in an FL environment.
2. **Cryptographic Context and Keys' Generations:** Public and private keys are generated using the PYFHEL library and shared among participating clients for encrypting and decrypting a model's parameters.
3. **Local Model Initialization, Training and Validation:** Each client trains the model on their own private dataset, where the data stays locally and is protected. After successful training and validation, the best model parameters are saved for each client.
4. **Encryption of Model Updates:** After the model is trained locally in a successful manner, its parameters, usually weights of the fine-tuned layers (not the entire model weights), are encrypted utilizing the CKKS encryption scheme.
5. **Secure Aggregation:** The encrypted parameters are sent via a secure channel to the central aggregator rather than raw data, where a secure federated averaging algorithm over encrypted parameters is performed to improve the global model's generalization and performance accuracy.
6. **Decryption and Model Updates:** The encrypted result of the secure averaging procedure is sent back to each participating client. Using the generated private key of the CKKS scheme, the model parameters regarding the global model are decrypted and reintegrated into the corresponding layers of the local CNN model.
7. **Iterative Training Rounds:** The model at each client is re-trained based on the re-embedded global updates, then encrypted again for further aggregation by the aggregator. This procedure is repeated for several rounds until the model performs adequately.
8. **Model Evaluation and Deployment:** The global model is evaluated and deployed for its intended use once it has achieved satisfactory performance and passed validation.

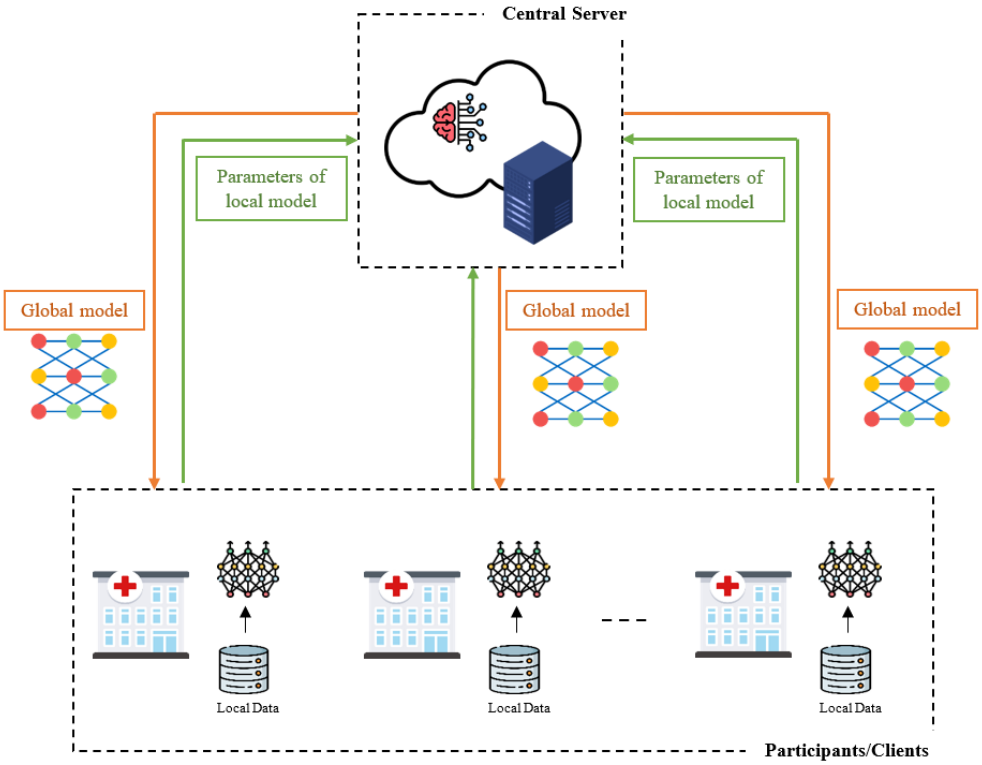


**FIGURE 1. - System overview**

### 4.1 NOTATIONS

$D_i$: *local private dataset at client i.*
$G_{architecture}$: *golbal model architecture.*
$h_{architecture}$: *local model architecture.*
$G_{weights}$: *global model weights.*
$h_{weights}$: *local model weights.*
*c* : *number of clients.*
*layer* : *layer within the model.*
*scheme*: *ckks.*
*n*: *polynomial modulud degree.*
*Scale*: *Encodings will use it for converting float to fixed point.*
*qi_size*: *Number of bits of each prime in the chain.*
$K_{pub}$: *public_key.*
$K_{priv}$: *private_key.*
$[\![Enc_{weights}]\!]$: *encrypted weights' matrix.*
$[\![Enc_{weights}]\!]_{agg}$ : *Aggregate Encrypted Weights Matrix*
$[\![Enc_{weights}]\!]_{Avg\_agg}$ :*Average encrypted weights matrix.*

### 4.2 CRYPTOGRAPHIC CONTEXT AND KEYS' GENERATIONS

This is a crucial step in safeguarding and ensuring privacy for the local model parameters (updates). PYFHEL first establishes the environment for cryptographic homomorphic operations by creating context. The context defines fundamental parameters for encryption, decryption, and homomorphic operations, such as scheme_type = CKKS in our case, polynomial modulus degree n = $2^{13}$, which increases the security level and computation complexity. The scale value, which affects precision and noise budget (value) in calculations (scale = $2^{30}$), and the coefficient modulus chain qi_sizes = [60, 30, 30, 30, 60], which is essential for the accuracy of the computations performed under encryption, is also defined. Next, a public key is generated to encrypt the model parameters from plaintext to ciphertext, and a private key is also generated and kept securely on the client's side, which helps to revert the ciphertext back to its original plaintext form. Algorithm 1 illustrates generating cryptographic context and keys (public and private) utilizing PYFHEL library.

**Algorithm 1:** Generate CKKS Encryption Context and Keys
**Input:**
Scheme (Encryption algorithm CKKS); n (polynomial degree); scale (scale factor); qi_sizes (coefficient modulus sizes).
**Output:**
HE (a Pyfhel object with initialized context and generated keys)
**Begin**
**Step 1:** HE ← Pyfhel() *// Initialize a new Pyfhel object, HE*
**Step 2:** Set the encryption scheme for HE with specified parameters:
**Step 2.1:** scheme: "CKKS"
**Step 2.2:** n ← 2**13
**Step 2.3:** scale: scale ← 2**30
**Step 2.4:** qi_sizes: [60, 30, 30, 30, 60]
**Step 3:** HE.contextGen (parameters) *// Generate context for HE with the specified scheme and parameters*
**Step 4:** HE.KeyGen () *// Generate encryption and decryption keys for HE*
**Step 5:** Capacity ← (n / 2 = 4,096)
**Step 6:** Return (HE) object with the context and keys configured
**End**

### 4.3 CLIENT INITIALIZATION

The steps outlined in Algorithm 2 illustrate the entire sequence of actions occurring during the initialization stage. Each client inherits both the global architecture and the global model weights, then independently retrains their model's higher layers using their own private dataset with a batch size of 128 and 40 rounds, while keeping the generic feature extraction layers unchanged and freezing their weights. The model is compiled with a categorical cross-entropy loss function and the Adam optimizer. Subsequently, the weight matrix of the trained model is encrypted and transmitted to the server using the HE-CKKS encryption scheme. For successful encryption, the public and private keys were generated beforehand (prior to the training step) based on the CKKS homomorphic crypto scheme by utilizing the PYFHEL library. After training, the algorithm iterates over each layer within the trained model, utilizing the generated

public key to encrypt its parameters, and then additively appends them to an encrypted weights matrix. This process continues until all weights are encrypted and properly structured before transmission.

**Algorithm 2** Client Model Training

**Input:**

Local dataset $D_i = \{(\boldsymbol{x}, \boldsymbol{y}) | \, \boldsymbol{x} \in \mathbb{R}^m, \boldsymbol{y} \in \mathbb{R}\}_{i=0}^{m}$, Public encryption key $K_{pub}$,

Initialized Global model $G_{architecture}$ with weight $G_{weights}$.

**Output:**

Encrypted weights matrix $[\![Enc_{weights}]\!]$

**Begin:**

**Step 1:** $(X_{train}^i, Y_{train}^i), (X_{test}^i, Y_{test}^i) \leftarrow SplitForTrainTest(D_i)$

**Step 2:** $h_{architecture} \leftarrow G_{architecture}$ *// Inherit Global Architecture*

**Step 3:** $h_{weights} \leftarrow G_{weights}$ *// Initialize Weights from Global Model*

**Step 4:** $h.compile(loss = \text{"categorical\_crossentropy"}, optimizer = \text{"Adam"})$

**Step 5:** $h.fit(X_{train}^i, Y_{train}^i)$

**Step 6:** $[\![Enc_{weights}]\!] \leftarrow \emptyset$

**Step 7:** For each *layer* ∈ *h* do:

**Step 7.1:** $Enc_{layer} \leftarrow Encrypt\ (layer_{weights}, K_{pub})$ *// Encrypt Layer Weights*

**Step 7.2:** $[\![Enc_{weights}]\!] \leftarrow [\![Enc_{weights}]\!] \cup Enc_{layer}$ *// Aggregate Encrypted Weights*

**Step 8:** End For

**Step 9:** Return $[\![Enc_{weights}]\!]$ *// weights' matrix in encrypted form*

**End**

### 4.4 MODEL AGGREGATION

The central server aggregation process starts by initializing a storage structure for the averaged encrypted weights and computes an encrypted version of the denominator, which represents the reciprocal of the total number of clients. The encrypted weights matrix of each client is then retrieved and homomorphically added to the aggregation process element-wise, ensuring that every corresponding weight belonging to each client equally contributes to the final aggregated sum. Subsequently, the average weight of neurons in the encrypted domain is computed using the Federated Averaging algorithm (FedAvg). Algorithm 3 provides a detailed description of the aggregation process.

**Algorithm 3:** Secure Model Aggregation

**Input:**

Encrypted weight matrices from all participating clients $\{[\![\text{Enc}_{\text{weights}}]\!]_1, [\![\text{Enc}_{\text{weights}}]\!]_2, \ldots, [\![\text{Enc}_{\text{weights}}]\!]_c\}$;

c : Number of clients;

$K_{pub}$: Public_key;

$[\![\text{Enc}_{\text{weights}}]\!]_{agg}$: Aggregate Encrypted Weights Matrix;

**Output:**

Average encrypted weights matrix $[\![Enc_{weights}]\!]_{Avg\_agg}$

**Begin:**

**Step 1:** $[\![Enc_{weights}]\!]_{agg} \leftarrow \emptyset$ *// Initialize Aggregate Encrypted Weights Matrix*

// Aggregation of Encrypted Weights Matrices:

**Step 2:** for i =1 to *c* do: *// In parallel*

**Step 2.1:** If $[\![\text{Enc}_{\text{weights}}]\!]_{agg} = \emptyset$ then

*// Set Initial Aggregate Encrypted Weights*

**Step 2.1.1:** $[\![Enc_{weights}]\!]_{agg} \leftarrow [\![Enc_{weights}]\!]_i$

**Step 2.2:** Else

*// Update Aggregate Encrypted Weights*

**Step 2.2.1:** $[\![Enc_{weights}]\!]_{agg} \leftarrow [\![Enc_{weights}]\!]_{agg} \cup [\![Enc_{weights}]\!]_i$

**Step 3:** End for

*// Secure Federated Averaging*

**Step 4:** $[\![\text{Enc}_{\text{weights}}]\!]_{\text{Avg\_agg}} \leftarrow \text{Enc}\left(\frac{1}{c}\right) \otimes \left(\sum_{i=1}^{c} \text{Enc}_{weights_i, layer}\right)$

**Step 5:** Return $[\![Enc_{weights}]\!]_{Avg\_agg}$

**End**

### 4.5 MODEL DECRYPTION

Once the weights have been encrypted and securely aggregated, clients are able to decrypt them. Each local client receives the updated global model and iterates through each layer to embed the updated weight values, then decrypts them with the private key through the PYFHEL library. It is essential to note that PYFHEL loses reference to the decrypted weights (floating point values) and does not have a direct link to its encrypted version; hence, re-referencing is needed. This is because the decryption procedure is unidirectional and treats the decrypted weights as a new entity. So, for the decryption to be entirely successful, we need to re-embed the decrypted floating point weight values to the model. The decryption process is essential for updating the local models, as demonstrated in Algorithm 4.

**Algorithm 4:** Decryption and Local Model Update

**Input:**

Private encryption key $K_{priv}$; Initialized global model $G_{architecture}$;
Averaged encrypted weights matrix $[\![Enc_{weights}]\!]_{Avg\_agg}$.

**Output:**

Updated local model $h$.

**Begin:**

**Step 1:** $h \leftarrow G_{architecture}$
**Step 2:** for each layer $\in h$ *do*
*// Retrieve encrypted weights for layer l*
**Step 2.1:** $Enc_{layer} \leftarrow [\![Enc_{weights}]\!]_{Avg\_agg}(l)$
*// Decrypt and update layer weights*
**Step 2.2:** *layer* $\leftarrow Decrypt\ (Enc_{layer}, K_{priv})$
**Step 3:** end for
**Step 4:** $h$.save_Model *// Save the updated local model*

**End**

## 5. EXPERIMENTAL EVALUATION

This section evaluates the performance and efficacy of our proposed methods. We outline the experimental setup, including the datasets used, architecture, and selected performance metrics. Subsequently, we present the results in a comparative manner and provide insights into the implications for privacy and computational efficiency.

### 5.1 DATASET AND PREPROCESSING

This study utilized the MIT-BIH arrhythmia database [31], containing properly labeled ECG records of six distinct types of cardiac arrhythmia (NOR, PVC, PAB, LBB, RBB, and APC). The ECG signals were converted into 2-D grayscale images with dimensions of 96x96. These images were then utilized as input for the ResNet50V2-based feature extractor. FTL was employed to facilitate decentralized training among multiple clients while ensuring data privacy through the use of HE-CKKS. The 2-D ECG beat images were fed into ResNet50V2, serving as a feature extractor, enabling the extraction of key features from the data. From the original dataset, 107,620 samples were partitioned into 80% training (86,092) and 20% testing (21,528). Both oversampling of minority classes and undersampling of dominant classes were implemented as strategies to address the issue of class imbalance. Since CNN served as a classifier, the training dataset was augmented with rotation, zoom, shifting, and flipping to mitigate overfitting and balance class distribution among federated participants. This preprocessing technique facilitated the effective utilization of spatial features of ECG data in a federated and privacy-preserving manner, establishing a robust foundation for subsequent analyses.

### 5.2 IMPLEMENTATION AND EXPERMENTAL SETUP

Table 1 presents a summary of the CNN architecture utilized in the ECG study, which was derived from ResNet50V2.

**Table 1. - Model summary**

| Layer (Type) | Output Shape | No. of Parameters |
|---|---|---|
| input_1 (InputLayer) | [(None, 96, 96, 3)] | 0 |
| conv1_pad (ZeroPadding2D) | (None, 102, 102, 3) | 0 |
| conv1_conv (Conv2D) | (None, 48, 48, 64) | 9472 |
| pool1_pad (ZeroPadding2D) | (None, 50, 50, 64) | 0 |
| pool1_pool (MaxPooling2D) | (None, 24, 24, 64) | 0 |
| conv2_block1_preact_bn (BatchNormalization) | (None, 24, 24, 64) | 256 |
| …… | | |
| post_relu (Activation) | (None, 3, 3, 2048) | 0 |
| avg_pool (GlobalAveragePooling2D) | (None, 2048) | 0 |
| flatten (Flatten) | (None, 2048) | 0 |
| dropout (Dropout) | (None, 2048) | 0 |
| d1 (Dense) | (None, 1024) | 2098176 |
| dropout_1 (Dropout) | (None, 1024) | 0 |
| d2 (Dense) | (None, 256) | 262400 |
| dropout_2 (Dropout) | (None, 256) | 0 |
| classifier (Dense) | (None, 6) | 1542 |

Total params: 25,926,918. Trainable params: 2,362,118. Non-trainable params: 23,564,800

The implementation code was written in Python 3.8.16 and relies on pre-existing third-party libraries. Keras and TensorFlow, two widely used ML libraries, were utilized. Data structures and weight arrays were manipulated using NumPy. Furthermore, weight export serialization was achieved using Pickle. PYFHEL [12], a Python wrapper for Microsoft SEAL, was employed for HE, offering similar functionalities to SEAL [32].

In 2015, Microsoft released the SEAL library for HE, which incorporates both BFV [33] and CKKS [26] schemes. The framework provides SHE throughout the entire process, including key generation, evaluation, and operations such as addition, multiplication, and relinearization.

In this study, we use the standard parameters for HE context generation to implement the CKKS scheme within the PYFHEL library. Security parameter $n$ = 8192, *scale factor* = $2^{30}$, and $q_{i\,sizes} = [60, 30, 30, 30, 60]$. The dataset is randomly distributed across clients C ∈ (2, 3, and 4). The predictive results of performance in the encrypted domain are then compared to those in the plaintext domain.

### 5.3 EXPERIMENTAL RESULTS

Initially, we conducted an experiment using the MIT-BIH dataset for ECG-based arrhythmia classification without employing the FTL and CKKS encryption scheme. Only one client participated in this baseline examination. Table 2 displays the performance metrics for this non-federated, non-encrypted model, with a total execution time of 2761.7636 seconds.

**Table 2. - Performance metrics for this non-federated, non-encrypted model**

| Accuracy | Precision | Recall | F1-score |
|---|---|---|---|
| 0.90537 | 0.87112 | 0.82637 | 0.84175 |

Figure 2 shows the loss and accuracy of the analyzed model's training set, while the confusion matrix depicts the results of the heartbeat classification conducted on the test set in Figure 3.

a b

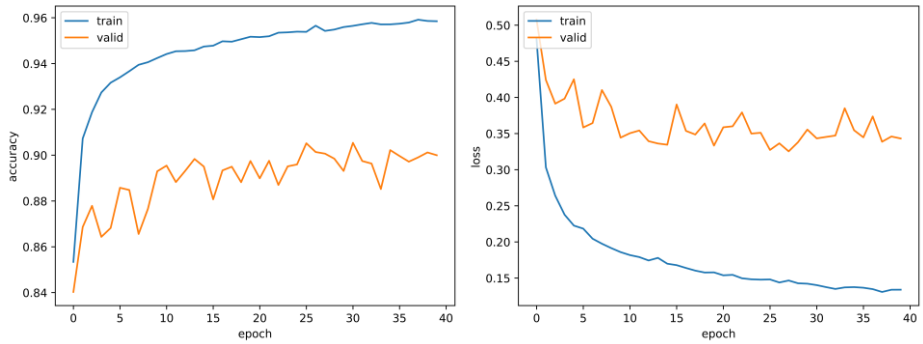


**FIGURE 2. – Model perfomance evaluation:(a) Model accuracy; (b) Model loss**

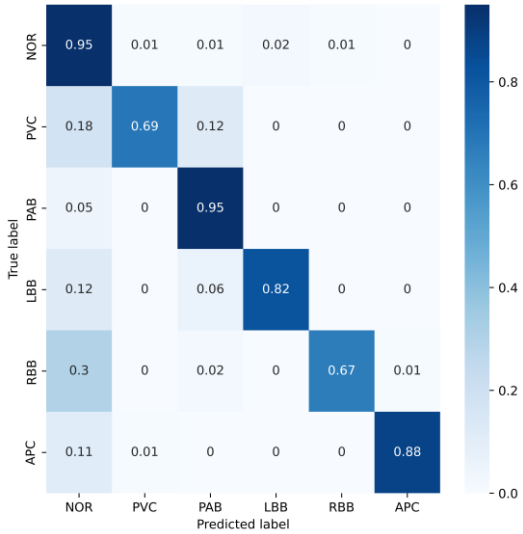


**FIGURE 3. – Confusion matrix for heartbeat classification on a test set**

Subsequently, we employed FTL in the framework and monitored the model's performance based on evaluation metrics. Table 3 presents a side-by-side comparison, assessing the performance metrics of our proposed model in both unencrypted and CKKS-encrypted scenarios.

**Table 3. - Performance metrics of FTL under different data configurations: Plain vs. Encrypted.**

| Balance-Augment configuration | Clients | Accuracy | | Precision | | Recall | | F1-score | |
|---|---|---|---|---|---|---|---|---|---|
| | | Plain | HE | Plain | HE | Plain | HE | Plain | HE |
| NoBalance, NoAugment | 2 | 0.84367 | 0.84367 | 0.92283 | 0.92283 | 0.53172 | 0.53172 | 0.59115 | 0.59115 |
| | 3 | 0.86763 | 0.82962 | 0.94197 | 0.93125 | 0.61931 | 0.50743 | 0.71147 | 0.60058 |
| | 4 | 0.85815 | 0.84493 | 0.93718 | 0.72842 | 0.56775 | 0.51881 | 0.64459 | 0.55127 |
| Balance, NoAugment | 2 | 0.82364 | 0.83849 | 0.67416 | 0.70171 | 0.83185 | 0.82468 | 0.71953 | 0.73535 |
| | 3 | 0.80897 | 0.81971 | 0.64816 | 0.68371 | 0.79393 | 0.80550 | 0.68094 | 0.70115 |
| | 4 | 0.77959 | 0.76012 | 0.60654 | 0.64030 | 0.80540 | 0.82636 | 0.66172 | 0.67549 |
| Balance, Augment | 2 | 0.71276 | 0.71822 | 0.54876 | 0.55980 | 0.75552 | 0.74910 | 0.58745 | 0.58625 |
| | 3 | 0.73564 | 0.67988 | 0.60486 | 0.59664 | 0.75553 | 0.78417 | 0.61769 | 0.61613 |
| | 4 | 0.72313 | 0.64602 | 0.56782 | 0.57961 | 0.75755 | 0.73265 | 0.60172 | 0.56798 |

Figures 4, 5, and 6 provide a comprehensive visualization of evaluation metrics in federated environments, including accuracy, precision, recall, and F1-score, across various data configurations grouped by the number of clients (2, 3, and 4). The initial models were trained without encryption, followed by the application of the HE-CKKS encryption scheme.

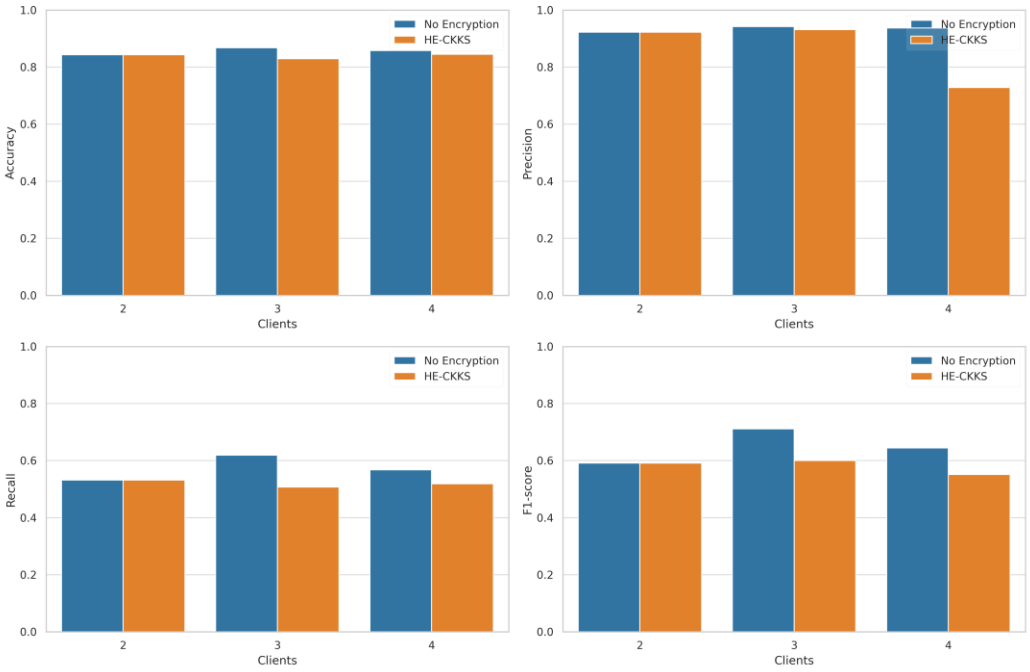


**FIGURE 4. – Performance metrics under 'NoBalance, NoAugment' data configuration**

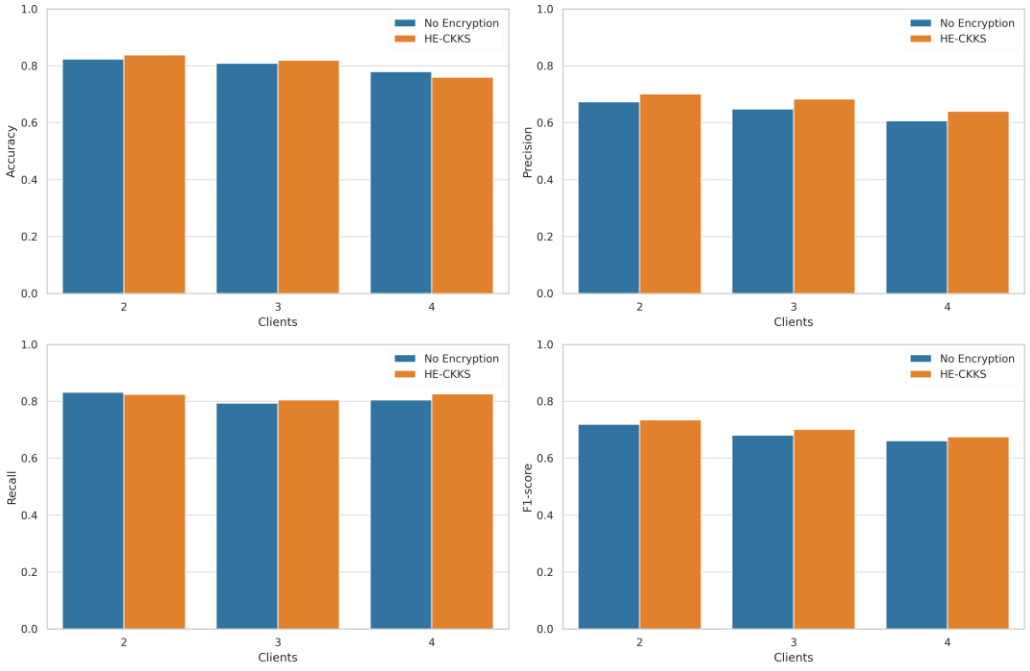


**FIGURE 5. – Performance metrics under 'Balance, NoAugment' data configuration**

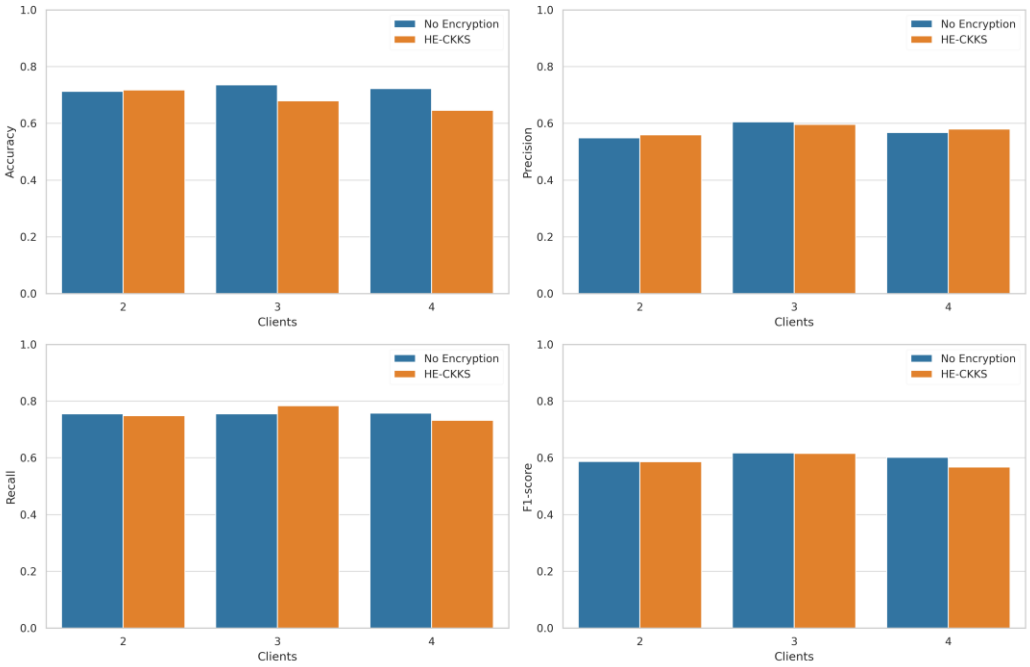


**FIGURE 6. – Performance metrics under 'Balance, Augment' data configuration**

In addition to the evaluation metrics computed above, the entire procedure proves to be computationally intensive, encompassing client GPU training, collaborative training, and the classification results using a secure aggregated model. Tables 4 and 5 illustrate the impact of encryption and the computational complexity associated with the collaborative process across varying numbers of clients.

**Table 4. - Encryption Overhead (in seconds)**

| | Number of Clients | | |
|---|---|---|---|
| | **2** | **3** | **4** |
| **Key Generation** | 0.0459 | 0.0403 | 0.044 |
| **Weight Encryption (Client_1)** | 15.6733 | 15.2855 | 15.3499 |
| **Weight Encryption (Client_2)** | 15.2039 | 15.1663 | 15.2111 |
| **Weight Encryption (Client_3)** | - | 15.2039 | 15.2303 |
| **Weight Encryption (Client_4)** | - | - | 15.2917 |
| **Aggregating Encrypted Weights** | 5.1959 | 5.3739 | 5.4311 |
| **Decrypt Encrypted Aggregated Weights** | 6.1114 | 6.2281 | 6.1971 |

**Table 5. - Computation Cost (Run Time in seconds)**

| No_Of_Clients | No Encryption | HE-CKKS |
|---|---|---|
| 2 | 3281.2124 | 3322.1771 |
| 3 | 3785.0586 | 3841.0353 |
| 4 | 4322.2305 | 4393.4354 |

Lastly, we utilized a histogram to visualize the growth in running time against HE. Figure 7 illustrates that implementing HE-CKKS led to a slight increase in running time attributed to TL. The low overhead can be attributed to the reuse of a pre-trained model for training the classifier layers.

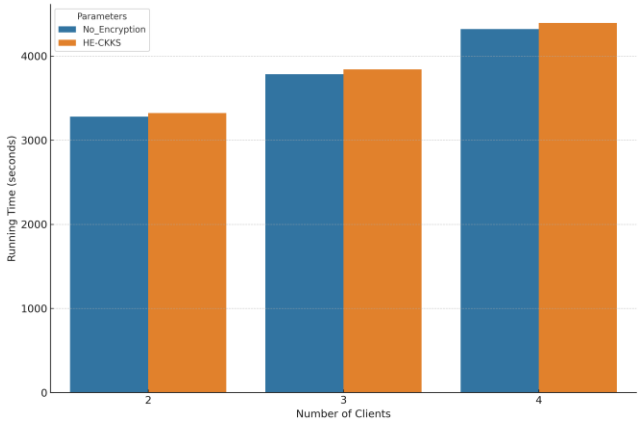


**FIGURE 7. – Running time in seconds.**

## 6. DISCUSSION

The experimental results presented in Table 3 provide valuable insights into encrypted and unencrypted FTL approaches, offering performance metrics across various preprocessing configurations and scalability. In FTL environments with different numbers of participants, our analysis of performance metrics demonstrates remarkable consistency, irrespective of the implementation of HE-CKKS encryption. This consistency is observed across different data preprocessing configurations, including "NoBalance, NoAugment," "Balance, NoAugment," and "Balance, Augment." Particularly, under the 'NoBalance, NoAugment' configuration, the accuracy metrics vary approximately from 0.844 to 0.868 with no encryption and from 0.844 to 0.830 with encryption. Similarly, the range for 'Balance, NoAugment' is 0.780 to 0.824 with no encryption and 0.760 to 0.838 with encryption. Lastly, in the 'Balance, Augment' configuration, the accuracy ranges from 0.713 to 0.736 with no encryption and from 0.646 to 0.718 with encryption. Regardless of whether HE-CKKS encryption is used, the minor variations in these performance metrics highlight the system's robustness and scalability. The findings indicate that HE-CKKS is a practical choice for preserving privacy in FTL, regardless of the number of clients or data preprocessing techniques utilized. This makes it well-suited for real-world applications.

Further investigation into computational overhead, as shown in Table 4, indicates that the runtime of both encrypted and non-encrypted configurations experiences a slight increase as the number of clients rises. TL implementation positively impacts computation time across various client numbers in different scenarios. The runtime for two clients increases by approximately 1.25%, for three clients it's around 1.48%, and for four clients, it's about 1.65%. When comparing execution times, CKKS encryption incurs slightly greater computational costs across all clients. However, the variance is negligible, suggesting that the CKKS scheme is reasonably efficient. Leveraging pre-trained ResNet50V2 as a feature extractor diminishes computational costs as encryption and fine-tuning become unnecessary in the FTL environment. Additionally, since encryption is solely applied to the classification part, computational overhead is confined to a smaller portion of the model compared to encrypting the entire architecture. In contrast, the work of Wibawa in the literature, which did not utilize TL, exhibited a substantial discrepancy in computation time between non-encrypted and encrypted versions, underscoring the effectiveness of our approach. Utilizing TL reduces these discrepancies, resulting in a more efficient computational process that significantly enhances overall runtime performance. Furthermore, the file size of their encrypted weights is approximately 7 GB, whereas in our case, it is roughly 2.44 GB regardless of execution time.

## 7. CONCLUSION

Data privacy is increasingly crucial, particularly in the healthcare sector, with regulations like GDPR in Europe and HIPAA in the US heightening the importance of secure data handling. FTL reduces data exposure by distributing data training across institutions. When paired with HE, it enables private and secure computations of sensitive data. While these technologies enhance performance, they also introduce computational complexity. As the number of clients grows, this trade-off becomes more pronounced, yet it remains acceptable in the healthcare sector, where privacy preservation is paramount.

In our FTL tests, the best outcomes were observed with three clients in a non-encrypted setting, where the accuracy reached 86.8%. Encrypted HE-CKKS mode marginally altered peak performance to four clients, achieving 84.5% accuracy. In both scenarios, data balance and augmentation led to a reduction in accuracy. When balancing and augmentation were applied, accuracy dropped to 64.6% in the encrypted option, compared to 71.2% in the non-encrypted setting. The results indicate that utilizing HE-CKKS encryption has a limited impact on the model's

accuracy. However, when encryption is combined with data balancing and augmentation, it presents challenges that necessitate further research to achieve optimal performance.

Privacy attacks pose a significant challenge to the development of data-driven healthcare models, as they compromise both data security and patient privacy. Therefore, enhancing data security and refining processing strategies are imperative. This study advocates for FTL and HE to bolster healthcare application security and efficacy by minimizing the sharing of sensitive data. Future research could explore the utilization of Multi-Key CKKS (MK-CKKS) HE schemes to add an extra layer of security. This feature ensures that aggregated data remains inaccessible to any participant, thereby enhancing data privacy and ensuring the confidentiality of model updates. Moreover, accommodating various healthcare domains will enhance the scalability and efficiency of the framework, ensuring its broader applicability across diverse settings.

## FUNDING

None

## ACKNOWLEDGEMENT

None

## CONFLICTS OF INTEREST

The author declares no conflict of interest.

## REFERENCES

[1] R. Miotto, F. Wang, S. Wang, X. Jiang, and J. T. Dudley, “Deep learning for healthcare: review, opportunities and challenges,” Brief. Bioinform., vol. 19, no. 6, pp. 1236–1246, Nov. 2018, doi: 10.1093/bib/bbx044.

[2] Alaa Hamza Omran, Sahar Yousif Mohammed, and M. Aljanabi, “Detecting Data Poisoning Attacks in Federated Learning for Healthcare Applications Using Deep Learning,” Iraqi J. Comput. Sci. Math., vol. 4, no. 4, pp. 225–237, Nov. 2023, doi: 10.52866/ijcsm.2023.04.04.018.

[3] P. Regulation, “Regulation (EU) 2016/679 of the European Parliament and of the Council,” Regul., vol. 679, no. April 2016, 2016, [Online]. Available: http://data.europa.eu/eli/reg/2016/679/oj

[4] L. O. Gostin, “National Health Information Privacy,” JAMA, vol. 285, no. 23, p. 3015, Jun. 2001, doi: 10.1001/jama.285.23.3015.

[5] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” Nature, vol. 521, no. 7553, pp. 436–444, May 2015, doi: 10.1038/nature14539.

[6] W. Li et al., “Privacy-Preserving Federated Brain Tumour Segmentation,” in In Machine Learning in Medical Imaging: 10th International Workshop, MLMI 2019, Held in Conjunction with MICCAI 2019, Shenzhen, China, vol. 11861, C. Suk, HI., Liu, M., Yan, P., Lian, Ed. Cham: Springer International Publishing, 2019, pp. 133–141. doi: 10.1007/978-3-030-32692-0_16.

[7] K. Aggarwal et al., “Has the Future Started? The Current Growth of Artificial Intelligence, Machine Learning, and Deep Learning,” Iraqi J. Comput. Sci. Math., vol. 3, no. 1, pp. 115–123, Jan. 2022, doi: 10.52866/ijcsm.2022.01.01.013.

[8] Y. Liu, Y. Kang, C. Xing, T. Chen, and Q. Yang, “A Secure Federated Transfer Learning Framework,” IEEE Intell. Syst., vol. 35, no. 4, pp. 70–82, Jul. 2020, doi: 10.1109/MIS.2020.2988525.

[9] Hameed, R. T., & Mohamad, O. A. (2023). Federated Learning in IoT: A Survey on Distributed Decision Making . Babylonian Journal of Internet of Things, 2023, 1–7. https://doi.org/10.58496/BJIoT/2023/001

[10] K. T. Chui et al., “Enhancing Electrocardiogram Classification with Multiple Datasets and Distant Transfer Learning,” Bioengineering, vol. 9, no. 11, pp. 1–21, 2022, doi: 10.3390/bioengineering9110683.

[11] M. Alloghani et al., “A systematic review on the status and progress of homomorphic encryption technologies,” J. Inf. Secur. Appl., vol. 48, p. 102362, Oct. 2019, doi: 10.1016/j.jisa.2019.102362.

[12] A. Ibarrondo and A. Viand, “Pyfhel,” in Proceedings of the 9th on Workshop on Encrypted Computing & Applied Homomorphic Cryptography, Nov. 2021, pp. 11–16. doi: 10.1145/3474366.3486923.

[13] J. Xu, B. S. Glicksberg, C. Su, P. Walker, J. Bian, and F. Wang, “Federated Learning for Healthcare Informatics,” J. Healthc. Informatics Res., vol. 5, no. 1, pp. 1–19, Mar. 2021, doi: 10.1007/s41666-020-00082-4.

[14] N. Rieke et al., “The future of digital health with federated learning,” npj Digit. Med., vol. 3, no. 1, p. 119, Sep. 2020, doi: 10.1038/s41746-020-00323-1.

[15] R. S. Antunes, C. André da Costa, A. Küderle, I. A. Yari, and B. Eskofier, “Federated Learning for Healthcare: Systematic Review and Architecture Proposal,” ACM Trans. Intell. Syst. Technol., vol. 13, no. 4, pp. 1–23, Aug. 2022, doi: 10.1145/3501813.

[16] M. J. Sheller, G. A. Reina, B. Edwards, J. Martin, and S. Bakas, “Multi-institutional Deep Learning Modeling

Without Sharing Patient Data: A Feasibility Study on Brain Tumor Segmentation," in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol. 11383 LNCS, Springer International Publishing, 2019, pp. 92–104. doi: 10.1007/978-3-030-11723-8_9.

[17] A. Vijaya Kumar, M. S. Sujith, K. T. Sai, G. Rajesh, and D. J. S. Yashwanth, "Secure Multiparty computation enabled E-Healthcare system with Homomorphic encryption," IOP Conf. Ser. Mater. Sci. Eng., vol. 981, no. 2, p. 022079, Dec. 2020, doi: 10.1088/1757-899X/981/2/022079.

[18] F. Wibawa, F. O. Catak, S. Sarp, and M. Kuzlu, "BFV-Based Homomorphic Encryption for Privacy-Preserving CNN Models," Cryptography, vol. 6, no. 3, p. 34, Jul. 2022, doi: 10.3390/cryptography6030034.

[19] F. Wibawa, F. O. Catak, M. Kuzlu, S. Sarp, and U. Cali, "Homomorphic Encryption and Federated Learning based Privacy-Preserving CNN Training: COVID-19 Detection Use-Case," in EICC 2022: Proccedings of the European Interdisciplinary Cybersecurity Conference, Jun. 2022, vol. 1, no. 1, pp. 85–90. doi: 10.1145/3528580.3532845.

[20] R. Bocu and C. Costache, "A homomorphic encryption-based system for securely managing personal health metrics data," IBM J. Res. Dev., vol. 62, no. 1, pp. 1:1-1:10, Jan. 2018, doi: 10.1147/JRD.2017.2755524.

[21] M. Kara et al., "A fully homomorphic encryption based on magic number fragmentation and El-Gamal encryption: Smart healthcare use case," Expert Syst., vol. 39, no. 5, pp. 1–14, Jun. 2022, doi: 10.1111/exsy.12767.

[22] A. Ali et al., "Deep Learning Based Homomorphic Secure Search-Able Encryption for Keyword Search in Blockchain Healthcare System: A Novel Approach to Cryptography," Sensors, vol. 22, no. 2, p. 528, Jan. 2022, doi: 10.3390/s22020528.

[23] Kavitha, T., Amirthayogam, G., Hephzipah, J. J., Suganthi, R., Kumar G, V. A., & Chelladurai, T. (2024). Healthcare Analysis Based on Diabetes Prediction Using a Cuckoo-Based Deep Convolutional Long-Term Memory Algorithm. Babylonian Journal of Artificial Intelligence, 2024, 64–72. https://doi.org/10.58496/BJAI/2024/009

[24] D. Gao, Y. Liu, A. Huang, C. Ju, H. Yu, and Q. Yang, "Privacy-preserving Heterogeneous Federated Transfer Learning," in 2019 IEEE International Conference on Big Data (Big Data), Dec. 2019, pp. 2552–2559. doi: 10.1109/BigData47090.2019.9005992.

[25] C. Marcolla, V. Sucasas, M. Manzano, R. Bassoli, F. H. P. Fitzek, and N. Aaraj, "Survey on Fully Homomorphic Encryption, Theory, and Applications," Proc. IEEE, vol. 110, no. 10, pp. 1572–1609, Oct. 2022, doi: 10.1109/JPROC.2022.3205665.

[26] J. H. Cheon, A. Kim, M. Kim, and Y. Song, "Homomorphic Encryption for Arithmetic of Approximate Numbers," in International Conference on the Theory and Application of Cryptology and Information Security., P. T. Takagi T., Ed. Springer, Cham, 2017, pp. 409–437. doi: 10.1007/978-3-319-70694-8_15.

[27] E.-B. van Veen, "Observational health research in Europe: understanding the General Data Protection Regulation and underlying debate," Eur. J. Cancer, vol. 104, pp. 70–80, Nov. 2018, doi: 10.1016/j.ejca.2018.09.032.

[28] P. W. Kariuki, P. K. Gikunda, and J. M. Wandeto, "Deep Transfer Learning Optimization Techniques for Medical Image Classifcation - A Survey," in 2022 International Conference on Intelligent Computing and Machine Learning (2ICML), Oct. 2023, pp. 7–15. doi: 10.1109/2ICML58251.2022.00013.

[29] M. K. Gajendran, M. Z. Khan, and M. A. K. Khattak, "ECG Classification using Deep Transfer Learning," in 2021 4th International Conference on Information and Computer Technologies (ICICT), Mar. 2021, no. July, pp. 1–5. doi: 10.1109/ICICT52872.2021.00008.

[30] M. Salem, S. Taheri, and J. Yuan, "ECG Arrhythmia Classification Using Transfer Learning from 2-Dimensional Deep CNN Features," in 2018 IEEE Biomedical Circuits and Systems Conference (BioCAS), Oct. 2018, pp. 1–4. doi: 10.1109/BIOCAS.2018.8584808.

[31] G. B. Moody and R. G. Mark, "The impact of the MIT-BIH Arrhythmia Database," IEEE Eng. Med. Biol. Mag., vol. 20, no. 3, pp. 45–50, 2001, doi: 10.1109/51.932724.

[32] H. Chen, K. Laine, and R. Player, "Simple Encrypted Arithmetic Library - SEAL v2.1," in International Conference on Financial Cryptography and Data Security, M. Brenner, K. Rohloff, J. Bonneau, A. Miller, P. Y. A. Ryan, V. Teague, A. Bracciali, M. Sala, F. Pintore, and M. Jakobsson, Eds. Cham: Springer International Publishing, 2017, pp. 3–18. doi: 10.1007/978-3-319-70278-0_1.

[33] H. Chen, K. Laine, and R. Player, "Simple Encrypted Arithmetic Library - SEAL v2.1," in Cryptology ePrint Archive, 2017, pp. 3–18. [Online]. Available: http://link.springer.com/10.1007/978-3-319-70278-0_1