TrustFedHealth: Federated Learning with Homomorphic Encryption and Blockchain for Heart Disease Prediction in the Smart Healthcare

Bui Tan Hai Dang[*†], Phan Huu Luan[*†], Vuong Dinh Thanh Ngan [*†], Nghia To Trong [*†],
Phan The Duy [*†], and Van-Hau Pham[*†]

[*]Information Security Laboratory, University of Information Technology, Ho Chi Minh City, Vietnam
[†]Vietnam National University, Ho Chi Minh city, Vietnam
{20520173, 20521585, 20521649}@gm.uit.edu.vn, {nghiatt, duypt, haupv}@uit.edu.vn

***Abstract*—In recent years, the advancements in the Internet of Medical Things (IoMT) or smart devices have enabled the automatic monitoring of human health. Using smart healthcare devices can not only reduce the burden on hospitals but also save costs, travel time, and provide a way to diagnose diseases at home, such as stroke rates. The IoMT generates a large amount of data, which can be used to train machine learning (ML) models for accurate disease diagnosis. However, the data used to train ML models is usually private, making it challenging to share and requiring high security. To overcome this challenge, this paper proposes a collaborative framework called TrustFedHealth for training ML-based heart disease prediction models. TrustFedHealth uses Federated Learning (FL) to allow training with decentralized data stored separately on multiple machines. Moreover, Mobile Edge Computing (MEC) is incorporated into the model as a solution to optimize communication time between the different elements of the system and reduce congestion. Homomorphic Encryption (HE) is also combined with FL to protect the confidentiality of model updates between clients and aggregation servers. Additionally, Blockchain (BC) is leveraged to tackle the traceability of contributions and guarantee transparency of model updates. Through evaluation results on the Physionet's MIT-BIH Arrhythmia Dataset, TrustFedHealth provides a promising approach for training ML-based heart disease prediction models while maintaining privacy and security.**

***Index Terms*—Federated Learning, Homomorphic Encryption, Blockchain, IoMT, Smart Healthcare, Mobile Edge Computing.**

I. Introduction

The Internet of Medical Things (IoMT) has expanded its potential applications in recent years, ranging from remote patient monitoring to emergency response and drug distribution systems [1]–[3]. Real-time data from previous studies can significantly enhance medical management, reduce hospitalizations, and increase patient satisfaction. However, traditional machine learning for IoMT typically involves collecting and storing data in a centralized place, such as a hospital or medical research facility. This approach may not be practical or feasible in circumstances where the data is generated and collected from distributed edge locations with different ownership.

As a result, concerns around data privacy, security, and ownership have led researchers to explore alternative approaches, Federated Learning (FL), for the future of Artificial Intelligence (AI) in health protection [4]. FL has the potential solution to overcome the limitations of centralized machine learning in IoMT and enable distributed edge devices to collaboratively learn and improve models without the need to share sensitive data [5], [6]. For instance, Kumar et al. [7] developed a FL-based system to detect COVID-19 using CT scans from multiple hospitals without sharing their data. However, FL still faces many challenges in communication between devices. As mentioned in the survey of Xuefei Yin et al. [8], central aggregation servers can be vulnerable to attacks and may impact the privacy of sensitive medical data from inferring the global models.

To resolve the privacy concerns in the process of FL model exchange, Homomorphic encryption (HE) techniques have been explored to mitigate this flaw and safeguard the shared model updates from suspicious parties. One such framework is PFMLP [9], which combines HE and FL to enable ML without compromising data privacy. The scheme of HE allows aggregation server computing on the ciphertext without decryption, making the FL more secure from unreliable servers. More significantly, Jing Ma et al. [10] came up with xMK-CKKS, as an improved version of the MK-CKKS scheme. It is a privacy-preserving FL scheme that provides strong data privacy protection in the FL scenario, but it is still maintaining model accuracy. Nevertheless, it should be noted that the computational overhead of the framework can be substantial due to the use of HE, without any optimizations such as batching, resulting in slower training times. Besides that, there are limitations of FL in the computing and storage capabilities of medical devices, inconsistent data formats, and communication delays between devices and central servers in practice.

To alleviate the computational cost and latency on central servers, MEC has emerged as a promising paradigm for improving the efficiency and responsiveness of mobile and IoMT devices. According to Lim Wy et al. [11], the power of this technology can combine with FL to reduce delay and communication costs as well as large computational of aggregation servers by moving computation and data storage closer to the edge. This approach can dramatically cut down the computational overhead, leading to faster and more responsive while maintaining data privacy.

Nonetheless, in the context of numerous organizations or training devices, it is difficult to handle and trace their contribution to collaborative training tasks with trustworthiness and transparency. To this end, the blockchain with outstanding features, such as decentralization, immutability, and traceability can be utilized to improve the security and scalability, and transparency of the contribution of joining parties in FL [12]. For instance, the integration of FL and BC creates a collaborative framework called FLchain [13], to revolutionize intelligent edge networks by incorporating them with a decentralized platform. Each device in FLchain is endowed with equal client rights to update and aggregate the learning model in a decentralized manner. It helps to create a secure and efficient data-sharing environment between health service providers, victims, and IoMT devices.

The above-mentioned studies have been developed to tackle some specific aspects of training ML models with the scheme of FL in IoMT. Nevertheless, they do not investigate the fusion of securing FL and transparency of model contribution and overhead and latency reduction in the context of smart healthcare. Hence, we introduce the TrustFedHealth scheme, a secure and privacy-preserving collaborative framework to overcome existing challenges in the collaborative training smart healthcare models. By combining FL, HE, BC, and MEC, our paradigm creates multiple layers of protection for data owners while providing an efficient performance in predicting heart diseases. Therein, BC is leveraged to monitor and record all activities in the collaborative learning process, ensuring transparency and traceability, and encouraging all parties involved to contribute in the local training phase.

The remainder of this work is organized as follows. **Section II** describes the architecture of our proposed TrustFedHealth in the context of smart healthcare. The experimental results are given in **Section III**. Finally, in **Section IV**, we conclude the paper.

II. Methodology

A. Overview of Architecture

The architecture of the TrustFedHealth framework, as shown in **Fig. 1** aims to address the challenge of building ML models in the context of smart healthcare, specifically in the domain of heart disease detection. Our framework comprises multiple layers, with user data being securely held in the lowest layer, i.e., at the data owner or end device level, through the use of a FL scheme. The user data is generated through the operation of smart health monitoring devices such as smartwatches and wristbands.

To further enhance the privacy and security of the FL scheme, we incorporate HE at the local training devices to make sure that the data remains encrypted even during model aggregation. It helps remove the privacy concerns of accessing a global model on an unreliable party like an aggregation server where local organizations do not control. Additionally, MEC is integrated into the FL scheme to allow FL to make a more distributed architecture, reducing the potential pressure on the network infrastructure and the computation requirements at the central aggregation server. The integration of HE and MEC in the FL scheme ensures that the privacy of the user data is protected throughout the entire training process, from local training to model aggregation, while maintaining the accuracy and effectiveness of the ML model and reducing the load on the central server.

Furthermore, we integrate Blockchain with our FL scheme to ensure the transparency and traceability of the system. The incentive mechanisms of BC ensure that data owners receive appropriate rewards for their contributions, which encourages more data owners to contribute data. The BC system records all operations within the system, including the training and aggregation of ML models, and controls access to the model file, which can be used to analyze possible attacks on their infrastructure.

B. Secure Federated Learning with HE and MEC

The TrustFedHealth architecture uses the HE-CKKS scheme [14] to achieve end-to-end security and privacy in FL. HE is a cryptographic technique that allows computation on encrypted data, without requiring decryption of the data. This means that sensitive data can be kept encrypted during the training process, while still allowing the ML model to be trained on the data. However, the computational overhead associated with HE is a limitation, but that can be resolved by integrating MEC to enable distributed data processing.

The workflow of TrustFedHealth is outlined in **Algorithm1**. As observed, the data processing in the entire process of model training is carried out through the HE technique to improve the privacy of data, particularly during the FedAvg computation using ciphertext data. Within the context of encrypted FL, the procedure comprises three fundamental steps: $encrypt, ciphertext - aggregate$ and $decrypt$. Each learner encrypts their locally trained model with the public key in the encryption step using a scheme of HE. Then, it transmits the resulting ciphertext to the server. Without ever decrypting any of the individual models, the server computes a new encrypted community model after receiving the encrypted local models from clients. The finalized new model is then distributed to each client, who uses their private keys to decrypt it. The clients train the new decrypted model on their own local data set after it has been decrypted. The entire process is then repeated until the model is converged.

In the FL-HE scheme, the central server would have to perform computationally intensive tasks, leading to high latency and a large amount of data transmission. The MEC framework provides a viable option for partitioning the aggregation process into two primary stages: the edge server and the central server. Initially, a global model is created on the central server, which is subsequently disseminated to all edge servers. At the edge servers, the model parameter (community weight) is encrypted and then disseminated to collaborative

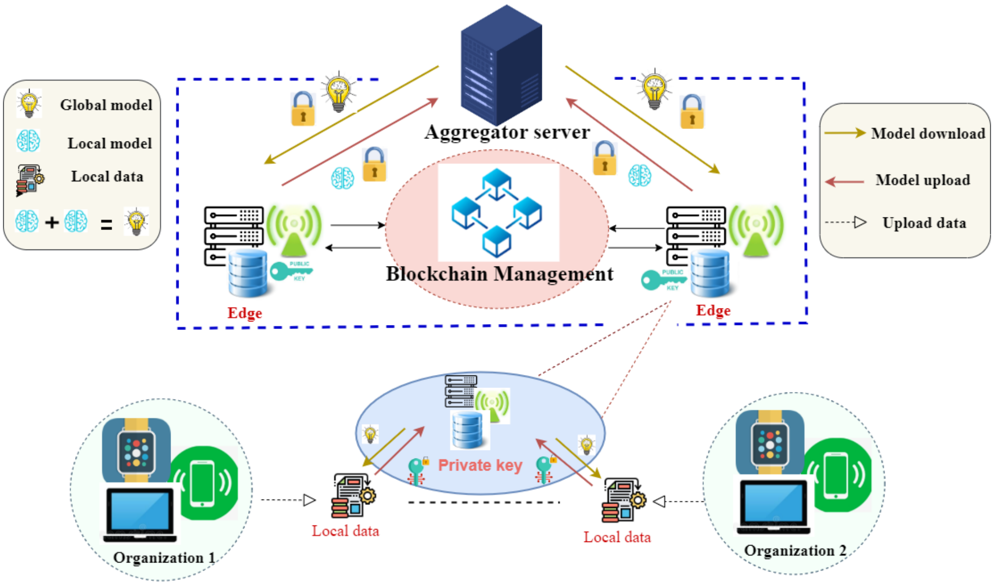

Fig. 1: Overview of TrustFedHealth architecture in Smart Healthcare

devices for facilitating training with the dataset of data owners. After training, the local parameters (local weights) are sent back to the edge server, where FedAvg algorithm is used to aggregate the parameters. The new aggregated parameter is then sent back to the clients. On each edge server cluster, these procedures are repeated several times before a model is created. The outcomes of the model production process on the edge servers are transferred to the central server for aggregation together with the encrypted parameters. The central server aggregates all received parameters to generate a new community parameter. Upon completion of the model synthesis process, the updated parameter is conveyed back to the edge servers. Notably, computations are performed solely using ciphertext throughout the entire process. This iterative process is repeated consistently to refine the model parameters until convergence is achieved.

Furthermore, the integration of MEC in the TrustFedHealth architecture can alleviate the computational load on the central server, leading to enhanced efficiency and collaboration in scenarios with limited resources.

C. *Blockchain-enabled Monitoring for Collaboration*

This study introduces a decentralized platform that combines the transparency and immutability of Hyperledger Fabric (Consortium Blockchain) with the collaborative learning capabilities of Federated Learning (FL) to facilitate the recording and analysis of member behaviors within the system. The selection of Hyperledger Fabric as the underlying blockchain framework is justified based on its compatibility with system requirements, encompassing access control, scalability, and transaction cost considerations. The integration of Hyperledger Fabric with FL enhances privacy protection, data ownership, and fosters collaborative learning among participants. Empirical results demonstrate the effectiveness and efficiency of the proposed decentralized platform in accurately recording and analyzing member behaviors. Additionally, the decentralized nature of blockchain, where all nodes possess equal rights to verify and manage data, ensures resilience against data modifications or attacks.

Our TrustFedHealth framework is designed to automate the process of serving as a connection between data owners and those who wish to develop ML models using the available data. Furthermore, a reward mechanism will be accurately and equitably implemented to encourage organizations to contribute their data. To this end, we have developed a smart contract that contains Transaction's Type and Transaction's Attributes, as described in **Table I**. This smart contract ensures the security and privacy of data, as it is only accessible by authorized parties with the necessary permissions. Throughout the collaborative learning process between the data owner and server, all activities will be recorded on the BC, thereby enabling the evaluation of member contributions and the rapid detection of abnormal behaviors. Should an attack occur, the transparency of the BC will enable us to trace and remove the attacker from the system with relative ease.

III. Experiment and Evaluation

A. *Experimental settings*

We have built a Convolution Neuron Network (CNN) for training an ML-based heart disease prediction model in TrustFedHealth. It contains five layers; the first three are namely Convolutional, Batch Normalization, Max Pooling, and Flatten layers. Each of these has 32 units, whereas the left ones are Fully Connected. The ReLU activation is applied to the output of every convolutional layer. The input will pass through these layers before feeding to Softmax function. Subsequently, our CNN model is trained using a learning rate of 0.01, a batch size of 100, and 5 epochs in one round.

Algorithm 1: The FL algorithm with HE-MEC

```
Input: The Aggregation Server $AggS$; The Edge
       Server $ES$; Clients $\mathcal{C}$ = { ($\mathcal{C}_k$); = 1,2,...,$n$} ;
       The batch size $B$; The local epochs $E$; The
       number of aggregation times $\mathcal{R}$; An
       authenticated server initializes $(SK, PK)$.
Output: Aggregated model for heart disease prediction
Init:
  1) $AggS$ generates a community model and initial
     parameter $W_c$.
  2) The $ESs$ receives the generated model.
  3) The $ESs$ encrypts $W_c$ with $PK$:
     $W_c^e = Encrypt(W_c, PK)$.
  4) The model is sent to the clients the ES handles.
  5) Clients $\mathcal{C}_k \in (\mathcal{C})$ receive the model M along with
     encrypted $W_c$ and associated parameters from $ES$.
for $r \leq R$ do
    (I). For the parties involved:
    for $\forall k \in (1, 2, 3, ..., n)$ do
        Firstly, Client $\mathcal{C}_k$ decrypts $W_c$
        $W_c = Decrypt$(W$_c^e$, SK)
        B $\leftarrow$ Split training data $D_k$ into batches of size
        $B$
        for $i \in E$ do
            for $b \in B$ do
                Client $\mathcal{C}_k$ train model using its datasets.
        Client $\mathcal{C}_k$ computes the weight $W_k^r$
        Client $\mathcal{C}_k$ uses $(PK)$ to encrypt.
        $E(W_k^r) = Encrypt(W_k^r, PK)$
        Client $\mathcal{C}_k$ sends the encrypted weights $E(W_k^r)$
        to the $ES$.
    (II). Aggregation Edge Server:
    for $\forall k \in (1, 2, 3, ..., n)$ do
        All parameter $E(W_k^r)$ from all clients are sent
        to the $ES$.
        Using the FedAvg method, the $ES$ creates a
        new parameter and then sends it back to its
        clients.
    Then, $ES$ sends that parameters to $AggS$
    (III). Aggregation Server:
    All parameters from all $ES$ are sent to the $AggS$.
    Using the FedAvg method, the $FedAvg$ creates a
    new parameter $W_c^e$ and then sends it back to the
    $Es$
    $r \leftarrow r + 1$
```

All the training tasks are implemented on Google Colab with Tensorflow framework. Besides that, the TenSEAL library [15] is utilized for Homomorphic Encryption. We use the HE-CKKS scheme with a poly modulus degree of 8096 and batching encoding to reduce costs and computation time. For our blockchain experiments, we deploy Hyperledger Fabric version 2.4.7 on Docker with Ubuntu 18.04, 32 GB of RAM, and a 60 GB hard drive on an Intel(R) Core(TM) i7-10750H CPU @ 2.60GHz machine. Hyperledger Caliper [16] version 0.5.0 is employed for comprehensive benchmarking of the BC system, evaluating metrics such as Successful Request, Latency, Throughput, and Resource Consumption of transactions on the blockchain platform.

TABLE I: The structure of BC transactions for Monitoring and Logging System

Transaction's Type	Transaction's Attributes
Register Task	TaskID, Key
Submit Task	TaskID, Key, Model Hash, Round
Get Model	Model_ID, Round
Update Model	Model_ID, newModelHashing, newQuality
Remove Model	Model_ID, Round
Create Model	Model_ID, modelHashing, Quality.
Claim Reward	TaskID, Key

B. Dataset

In this article, we use the famous Physionet's MIT-BIH Arrhythmia Dataset which comes from ECG Heartbeat Categorization [17] to validate the effectiveness of our designed system. The dataset was gathered from a total of 452 patients, comprising both those with various arrhythmias and healthy individuals. The authors preprocessed this dataset; thereby, no further processing was required. The original dataset with 5 labels is transformed into two labels, Normal and Abnormal. In different circumstances during the experiment, the data set is distributed evenly among all members joining the training process and each member has an equal ratio of labels.

C. Evaluation metrics and scenarios

We employ evaluation standards metrics including accuracy, precision, recall, and F1-score to assess the performance of models. These metrics are computed relying on the rates of True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN)

In addition, this study has proposed five scenarios aimed at evaluating the effectiveness of our designed architecture.

- *Scenario 1:* Training the model using FL.
- *Scenario 2:* Training the model using FL with the scheme of HE.
- *Scenario 3:* Training the model using FL with MEC.
- *Scenario 4:* Training the model using FL-MEC-HE.
- *Scenario 5:* Measure the performance of the BC system.

D. Experimental result

The outcomes derived from the experimental contexts are presented as follows:

1) Training the model using FL: In this scenario, we use the whole preprocessed and balanced data sets to train our FL model. Individuals receive an equal share of the data for each training run, consisting of 5 epochs with a batch size of 64. The model is trained with varying numbers of clients (2, 4, 6, and 8) and training rounds (10 and 20) to comprehensively evaluate performance. Additionally, the traditional learning

TABLE II: Traditional Learning method performance of heart disease prediction model

Accuracy	Recall	Precision	F1-Score
0.9892	0.9741	0.9923	0.9802

TABLE III: FL performance of heart disease prediction model

Round	Client	Accuracy	Recall	Precision	F1-score
10	2	0.9688	0.8468	0.9682	0.9033
	4	0.9793	0.8967	0.9814	0.9371
	6	0.9815	0.9187	0.9728	0.9449
	8	0.9837	0.9306	0.9739	0.9518
20	2	0.9784	0.9189	0.9543	0.9363
	4	0.9810	0.9295	0.9590	0.9440
	6	**0.9829**	**0.9780**	**0.9752**	**0.9696**
	8	0.9792	0.9010	0.9764	0.9764

TABLE IV: Data size after encryption: Batch Encryption vs. no Batch Encryption.

Using Batch Encryption	Plaintext size	Ciphertext size	Time to encrypt
No	1.7 MB	Out of memory	
Yes	1.7 MB	689.6 MB	6.5s

TABLE V: Statistics of total data processed by the server in 20 Rounds.

Number of Client	10	100	1000
FL	680 MB	6800 MB	68000 MB
FL + 10 Edge Server	680 MB	680 MB	680 MB
FL+ 5 Edge Server	340 MB	340 MB	340 MB
FL+ 2 Edge Server	136 MB	136 MB	136MB

method is employed, where the model is trained on the entire dataset for 5 epochs with a batch size of 64. The corresponding results are presented in **Table II**. Results in **Table III** show that increasing the number of clients from 2 to 6 in round 10 leads to improved accuracy, recall, precision, and f1-score. Similarly, in round 20, participants achieve higher metrics. These findings suggest minimal performance differences among clients in federated learning compared to traditional methods, with instances where federated learning outperforms in terms of recall rate. This underscores the efficacy of federated learning in preserving data privacy while maintaining model performance. Additionally, we utilize the outcomes obtained from training the traditional FL model, as illustrated in **Fig. 2**, as a fundamental reference point to compare and contrast with the models we propose. For clarity, we selected 6 clients and 20 rounds in our FL model for evaluation due to their superior metrics compared with others in **Table III**.

2) Training the model using FL with HE: Our paper employs HE with an effective technique known as Batch Encryption. It is a useful optimization technique that speeds up HE schemes by allowing parallel processing of multiple plaintexts during encryption/decryption and homomorphic evaluation. The technique involves concatenating several plaintexts into a single, so it significantly decreases the computational complexity and encoding time of the model, as well as data size. When Batch Encryption is not applied, the data increases exponentially, leading to a model that cannot be computed. In contrast, while applying batch processing, the data size is markedly decreased, which is shown in **Table** IV. Despite a significant reduction in size using the batch technique, the ciphertext still remains larger than the plaintext, which is one of the limitations of our study.

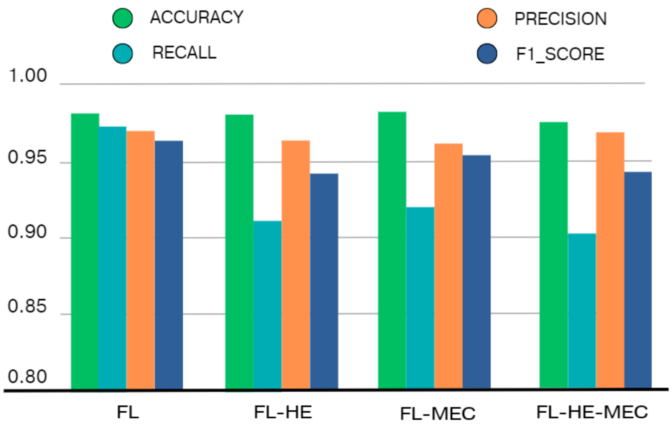

Fig. 2: The performance metrics of our proposed models

After that, we conduct training on our model FL with the scheme of HE. The result is summarized in the **Fig. 2** yields that their accuracy is as high as traditional FL. Despite the increase in data size and training time, the performance of the model remained good as well as the security of the model was significantly improved. This indicates a promising combination of the two technologies.

3) Training the model using FL with MEC: Our advanced CNN model with a size of 1.7 MB assumed that we will train that model in 20 Rounds (R=20). If 10 clients (K=10) participate in the data contribution process, the total data exchanged between the server and clients in the traditional FL strategy is 680 MB. As opposed to this, when MEC is used with 2 Edge Servers, the total amount of data that the aggregation server receives is 136 MB, which drops by around 5 times. **Table V** demonstrates the statistical findings after raising the client count from 100 to 1000 and the number of edge servers from 5 to 10 with 20 rounds and a 1.7 MB model size. Additionally, we also train that model to examine the performance of the multi-layer deep learning model FL-MEC. In this case, we use two Edge Sever to aggregate models from clients before sending them to the main server. The result in **Fig. 2** indicates that the FL-MEC model exhibits a slight down in recall and F1-score metrics compared to the FL scheme, although accuracy and precision remain high, which are both greater than 97%. Besides, edge servers alleviate the central server's computational burden and enhance data transmission, mitigating data loss risk.

4) Training the model using FL-MEC-HE: In this scenario, we integrate HE with the prior FE-MEC model to strengthen

TABLE VI: Result in training model using FL-HE-MEC

Round	Client	Accuracy	Recall	Precision	F1-Score
10	**2**	0.9748	0.9009	0.9503	0.9249
	4	0.9757	0.8834	0.9734	0.9262
	6	0.9787	0.9030	0.9712	0.9358
	8	0.9777	0.9038	0.9603	0.9312
20	**2**	0.9793	0.9054	0.9727	0.9378
	4	0.9740	0.9348	0.9163	0.9255
	6	**0.9803**	**0.9067**	**0.9774**	**0.9407**
	8	0.9779	0.9010	0.9521	0.9385

TABLE VII: The performance of Get Model transaction in smart contract

Successful request	Min latency(s)	Max latency (s)	Avg latency (s)	Throughput (TPS)
1000	0.001	0.04	0.00	17.18
5000	0.003	0.17	0.01	42.34
10000	0.003	0.25	0.01	56.10
50000	0.012	0.42	0.05	209.94
100000	0.018	0.64	0.07	335.52

the confidentiality of data during the transmission process. As depicted in **Fig.** 2 and **Table** VI, despite a modest decline in the recall, and f1-score metrics, the accuracy and precision metrics are still as great as the FL model. Every indicator in the FL-MEC-HE model is likewise above 91%. The accuracy metric takes the highest percentage at 98% while the recall metric displays the lowest percentage at 90%. Consequently, the integration of the three technologies not only enhances the security of our paradigm, but also guarantees the quality of the training model. Alternatively, the incorporation of the CKKS-HE scheme is confronted with the challenge of handling a considerable surge in data size as well as the extended time required for model training, amounting to 689.6 MB and 6.5s for Ciphertext size and Time to encrypt, respectively, with a Plaintext size of 1.7 MB.

5) Measure the performance of the BC system combined with Collaborative Learning: One of the main tasks of a smart contract when interacting with a system is to retrieve the model, so this method will be used for evaluation. To have a more comprehensive overview, we increased the number of transactions from 1000 to 100000 to obtain a table of data as shown in **Table VII**. The more requests, the more latency, and the highest latency is 0.64s when the request number reaches 100000. These findings regarding resource consumption and system performance demonstrate that the system adequately meets the requirements of data monitoring and logging.

IV. Conclusion

This paper introduces a secure collaborative framework that utilizes FL and HE techniques for users and organizations to share private data for training purposes in a confidential and secure manner, without compromising privacy. Furthermore, our TrustFedHealth system leverages highlight technologies such as BC, and MEC to optimize resource consumption and operational costs. Based on the in-depth experimental results of the Physionet's MIT-BIH Arrhythmia Dataset, proves that our system effectively employs community data to attain a model with high accuracy. It also offers a high level of flexibility and can be adapted to diverse applications. Despite bringing advantages from our proposed architecture, there are certain shortcomings that require further improvement. Since there is a notable expansion in the volume of data and increasing training time when applying HE. In the future, we intend to strengthen HE effectiveness for our paradigm.

References

[1] T. McGhin, K.-K. R. Choo, C. Z. Liu, and D. He, "Blockchain in healthcare applications: Research challenges and opportunities," *Journal of Network and Computer Applications*, vol. 135, pp. 62–75, 2019.

[2] G. J. Joyia, R. M. Liaqat, A. Farooq, and S. Rehman, "Internet of medical things (iomt): Applications, benefits and future challenges in healthcare domain." *J. Commun.*, vol. 12, no. 4, pp. 240–247, 2017.

[3] S. Vishnu, S. J. Ramson, and R. Jegan, "Internet of medical things (iomt)-an overview," in *2020 5th international conference on devices, circuits and systems (ICDCS)*. IEEE, 2020, pp. 101–104.

[4] N. Rieke, J. Hancox, W. Li, F. Milletari, H. R. Roth, S. Albarqouni, S. Bakas, M. N. Galtier, B. A. Landman, K. Maier-Hein *et al.*, "The future of digital health with federated learning," *NPJ digital medicine*, vol. 3, no. 1, p. 119, 2020.

[5] L. Ngan Van, A. Hoang Tuan, D. Phan The, T.-K. Vo, and V.-H. Pham, "A privacy-preserving approach for building learning models in smart healthcare using blockchain and federated learning," in *Proceedings of the 11th International Symposium on Information and Communication Technology*, 2022, pp. 435–441.

[6] S. Singh, S. Rathore, O. Alfarraj, A. Tolba, and B. Yoon, "A framework for privacy-preservation of iot healthcare data using federated learning and blockchain technology," *Future Generation Computer Systems*, vol. 129, pp. 380–388, 2022.

[7] R. Kumar, A. A. Khan, J. Kumar, N. A. Golilarz, S. Zhang, Y. Ting, C. Zheng, W. Wang *et al.*, "Blockchain-federated-learning and deep learning models for covid-19 detection using ct imaging," *IEEE Sensors Journal*, vol. 21, no. 14, pp. 16 301–16 314, 2021.

[8] X. Yin, Y. Zhu, and J. Hu, "A comprehensive survey of privacy-preserving federated learning: A taxonomy, review, and future directions," *ACM Computing Surveys (CSUR)*, vol. 54, no. 6, pp. 1–36, 2021.

[9] H. Fang and Q. Qian, "Privacy preserving machine learning with homomorphic encryption and federated learning," *Future Internet*, vol. 13, no. 4, p. 94, 2021.

[10] J. Ma, S.-A. Naas, S. Sigg, and X. Lyu, "Privacy-preserving federated learning based on multi-key homomorphic encryption," *International Journal of Intelligent Systems*, vol. 37, no. 9, pp. 5880–5901, 2022.

[11] W. Y. B. Lim, N. C. Luong, D. T. Hoang, Y. Jiao, Y.-C. Liang, Q. Yang, D. Niyato, and C. Miao, "Federated learning in mobile edge networks: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 3, pp. 2031–2063, 2020.

[12] R. Myrzashova, S. H. Alsamhi, A. V. Shvetsov, A. Hawbani, and X. Wei, "Blockchain meets federated learning in healthcare: A systematic review with challenges and opportunities," *IEEE Internet of Things Journal*, pp. 1–1, 2023.

[13] D. C. Nguyen, M. Ding, Q.-V. Pham, P. N. Pathirana, L. B. Le, A. Seneviratne, J. Li, D. Niyato, and H. V. Poor, "Federated learning meets blockchain in edge computing: Opportunities and challenges," *IEEE Internet of Things Journal*, vol. 8, no. 16, pp. 12 806–12 825, 2021.

[14] K. A. K. M. . S. Y. Cheon, J. H., "Homomorphic encryption for arithmetic of approximate numbers," *In Advances in Cryptology – ASIACRYPT 2017*, vol. 1, no. 23, pp. 409–437, 2017.

[15] A. Benaissa, B. Retiat, B. Cebere, and A. E. Belfedhal, "Tenseal: A library for encrypted tensor operations using homomorphic encryption," *arXiv preprint arXiv:2104.03152*, 2021.

[16] Q. Nasir, I. A. Qasse, M. Abu Talib, and A. B. Nassif, "Performance analysis of hyperledger fabric platforms," *Security and Communication Networks*, vol. 2018, 2018.

[17] M. Kachuee, S. Fazeli, and M. Sarrafzadeh, "Ecg heartbeat classification: A deep transferable representation," in *2018 IEEE international conference on healthcare informatics (ICHI)*. IEEE, 2018, pp. 443–444.