AP2FL: Auditable Privacy-Preserving Federated Learning Framework for Electronics in Healthcare

Abbas Yazdinejad, Ali Dehghantanha, *Senior Member, IEEE*, and Gautam Srivastava, *Senior Member, IEEE*

***Abstract*—The growing application of machine learning (ML) techniques in healthcare has led to increased interest in federated learning (FL), which enables the secure and private training of robust ML models. However, conventional FL methods often fall short of providing adequate privacy protection and face challenges in handling non-independent and identically distributed (Non-IID) training data. These shortcomings are of significant concern when employing FL in electronic devices in healthcare. To address these issues, we propose an Auditable Privacy-Preserving Federated Learning (AP2FL) model tailored for electronics in healthcare settings. By leveraging Trusted Execution Environments (TEE), AP2FL ensures secure training and aggregation processes on both client and server sides, effectively mitigating data leakage risks. To manage Non-IID data within the proposed framework, we incorporate the Active Personalized Federated Learning (ActPerFL) model and Batch Normalization (BN) techniques to consolidate user updates and identify data similarities. Additionally, we introduce an auditing mechanism in AP2FL that reveals the contribution of each client to the FL process, facilitating the updating of the global model following diverse data types and distributions. In other words, it ensures the FL process's integrity, transparency, fairness, and robustness. Our results demonstrate that the proposed AP2FL model outperforms existing methods in accuracy and effectively eliminates privacy leakage.**

***Index Terms*—Privacy, FL, auditing, non-IID, healthcare.**

I. Introduction

HEALTHCARE research often involves collecting data from various sources, such as healthcare providers, pharmacies, health insurers, and academic institutions. As health information is highly sensitive, legal and social implications arising from its disclosure render privacy a major concern in the healthcare sector. Consequently, increasing numbers of governments have enacted regulations to safeguard personal information, including electronic devices in healthcare [1]. Applying machine learning (ML) techniques in healthcare has fueled interest in federated learning (FL) to ensure data privacy in electronic healthcare. FL enables privacy preservation in ML by sharing model parameters between clients and servers rather than raw data [2], [3]. Although FL is universally acknowledged as a method that preserves privacy (PP), it isn't exempt from privacy compromises and security risks, including Inversion Attacks and Inference Attacks. These susceptibilities pose potential threats to the inherent privacy attributes of FL [4]. Recent studies have revealed that adversaries can extract sensitive information through model parameter-based attacks [5]. Data reconstruction and inference attacks [6], [7] exemplify such threats, which can occur when ML models inadvertently embed irrelevant information from training data. Both clients and servers can initiate such attacks in FL settings.

A further challenge in FL lies in the varying data distributions among clients. Non-independent and identically distributed (non-IID) data is a particular concern in healthcare, given the diverse demographics, lifestyles, patients, and countries represented in healthcare data sets [8]. This issue can adversely impact healthcare applications, especially those involving FL. Furthermore, it is crucial to understand the contributions of individual participants in a federated setup, which allows for assessing each client's performance and developing a reliable global model. An audit mechanism can also help identify malicious clients who intentionally tamper with training data (poisoning attacks). Thus, incorporating auditability into the FL model design is highly desirable. To successfully implement ML models in healthcare, addressing these challenges in FL is essential. However, existing FL models often lack privacy guarantees and audit capabilities, may leak privacy information, and struggle to handle non-IID data among clients effectively. In response, we propose an Auditable Privacy-Preserving Federated Learning (AP2FL) model for healthcare applications. This model employs Trusted Execution Environments (TEE) on both the client and server sides to prevent privacy leakage in FL settings. The selection of TEE is based on their ability to facilitate trustworthy interactions between domain and data models, which is crucial for maintaining the confidentiality of medical data collaborations.

To address the non-IID data distribution across various clients, we incorporate the Active Personalized Federated Learning (ActPerFL) [9] and Batch Normalization (BN) in our DL model training within an FL setup. The ActPerFL model strikes a productive balance between local and global training phases while also indirectly aiding other clients' training. This model signifies an active learning strategy in FL designed to counter the challenges posed by non-IID data. The approach picks the most significant data points for training, aiming to

Manuscript received 25 April 2023; revised 26 June 2023; accepted 13 September 2023. Date of publication 22 September 2023; date of current version 26 April 2024. *(Corresponding author: Gautam Srivastava.)*

Abbas Yazdinejad and Ali Dehghantanha are with the Cyber Science Lab, Canada Cyber Foundry, University of Guelph, Guelph, ON N1G 4S7, Canada (e-mail: ayazdine@uoguelph.ca; adehghan@uoguelph.ca).

Gautam Srivastava is with the Department of Mathematics and Computer Science, Brandon University, Brandon, MB R7A 6A9, Canada, also with the Research Centre for Interneural Computing, China Medical University, Taichung City 404, Taiwan, and also with the Department of Computer Science and Mathematics, Lebanese American University, Beirut 1102-2801, Lebanon (e-mail: srivastavag@brandonu.ca).

Digital Object Identifier 10.1109/TCE.2023.3318509

lower the data requirements and boost overall performance. Meanwhile, BN plays a crucial role in optimizing model performance by mitigating the variations in input distribution across different batches of data. Additionally, we introduce an auditing mechanism in AP2FL to track the impact of each client in FL, enabling updates to the global model based on diverse data types. Our contributions include:

- Design and implement the privacy-preserving federated learning (PPFL) model for electronics in healthcare.
- In FL-based healthcare environments, we adopt TEE for local training on both client and server sides while the auditor component communicates with them.
- Provide support for non-IID clients in healthcare environments using the ActPerFL model and BN in a DL model while preserving each client's specificity.
- Propose an audit method to not only validate participants' private training data but also ensure the FL process's integrity, transparency, fairness, and robustness.

The rest of the paper can be broken down as follows. Discussion of PP FL works presented in Section II. Section III provides an overview of the AP2FL framework and Section IV gives the security analysis. We assess the proposed model in Section V. Finally, in Section VI, we conclude and outline future directions for the paper.

II. Related Works

Due to the critical nature of PP in FL [10], particularly in healthcare, we provide a concise review of studies focused on mitigating privacy leakage and non-IID data in federated environments. FL research has explored membership inference to determine if a specific data sample is part of the training set [11], [12]. Moreover, investigators have found that exchanged data can be exploited to extract unintended private information, such as the presence of eyeglasses [13]. Li et al. [14] demonstrated that the most significant privacy threat arises when determining the optimal input-label pairs corresponding to exchanged gradients. Building on this approach, [15] proposed an analytical method to extract label information, although its applicability is limited to shallow networks trained on low-resolution images. Geiping et al. [6] extended this to restore ImageNet-level high-resolution data from deeper networks using a magnitude-invariant loss design.

In more recent work, Yin et al. [16] employed BN statistics to encode a strong prior, enabling image batch reconstruction. Furthermore, the study by [17] introduced multi-agent federated reinforcement learning, focusing on a mobile and fog agents-based paradigm. This approach aims to create a consumer-centric, cyborg-efficient training and testing system within the Internet of Medical Things (IoMT). In non-FL, Sun et al. [18] provided PP in medical record searching for IoT Healthcare. To achieve PP in FL, [19] proposed a PPFL method via TEE. Greedy layer-wise training was utilized for local training within the trusted area on clients; however, this approach entails synchronization and communication overheads. The non-IID assumption in this work was oversimplified, with each client selecting samples from only two random classes. Another study [20] presented a personalized

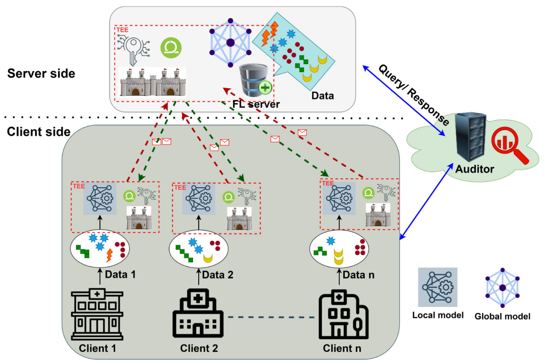

Fig. 1. A schematic diagram of the AP2FL framework.

FL approach with BN for healthcare, which supports non-IID scenarios, addresses domain shifts, and generates personalized models for local clients. However, this method neglects PPFL concerns. Although comprehensive experiments demonstrated satisfactory accuracy, the popular FL algorithm FedAvg [21] often outperforms other methods [22]. FedAvg [23] is ill-suited for handling non-IID data from multiple clients, as it directly averages parameter values from all participants. Several algorithms have been developed to address non-IID situations, such as FedProx [24], which is tailored for non-identical data. However, FedProx cannot generate personalized models for clients, as it learns a global model for all participants. Accessing large public datasets like FedHealth [25] is often infeasible in real-world applications. FedBN [26] addresses the non-IID issue by learning local BN layers for each client but fails to consider cross-client similarities that could enhance the personalization. Although previous research has aimed to achieve PP in FL or address non-IID issues in FL environments, no comprehensive work has tackled these challenges in healthcare applications. Furthermore, no additional PP measures or defences, such as auditable features, have been incorporated in FL environments within healthcare. This paper highlights these concerns as our primary contributions and discusses them in detail throughout the work.

III. The AP2FL Framework

This section introduces a novel auditable PPFL model for healthcare that aims to minimize privacy leakage and non-IID issues in federated healthcare environments. Fig. 1 provides a comprehensive view of the proposed model, which includes multiple hospitals (clients A, B, etc.) with distinct data distribution characteristics (e.g., clients A and B have different activities and lifestyles), an FL server, and an Auditor component. In our proposed model, we employ TEE to prevent data leakage at both client and server levels, support non-IID situations by implementing the ActPerFL model and BN method, and incorporate an auditing feature into our model design. The Auditor component serves as a security policy manager, facilitating client and server communication. In the subsequent sections, we elaborate on TEE, the ActPerFL model, BN, and auditing.

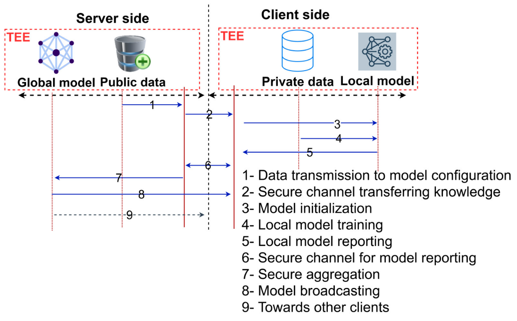

Fig. 2. Workflow of the AP2FL framework.

A. *Remove Privacy Leakage*

To mitigate privacy leakage in federated healthcare environments, we adopt a TEE-based approach, as TEE can effectively prevent data leakage on both server and client sides within a federated context. Executing ML and DL models within TEE can conceal model parameters from Rich Operating System Execution Environment (REE) adversaries and maintain privacy, as already demonstrated during model training and lightweight data analytics. TEE are emerging security technologies that offer promising solutions for mitigating system attacks [27]. They facilitate data processing with fine-grained access control and memory protection through a hardware root of trust [27]. TEE have various use cases, including cloud computing, the Internet of Things (IoT), multi-party computations (MPC), and ML.

In the AP2FL model, privacy risk is dependent on the aggregation level, as gradients remain inaccessible to adversaries while being updated within TEE. This approach also defends against privacy attacks such as data reconstruction attacks (DRAs) and parameter inference attacks (PIAs) [19]. As illustrated in Fig. 2, the AP2FL model is schematically depicted, showcasing the elimination of privacy leaks through the use of TEE. The server uses random weights or publicly available information to assign weights to the global model. Utilizing random or public data for global model initialization in AP2FL promotes privacy and fairness, despite potential slower convergence or initial performance issues. In this way, the model m is prepared for broadcast after it has been initialized (m). By using TEE, the server constructs secure communication with participating clients. During steps 1 and 2, data will be transmitted to the model configuration through a secure channel. Once the model is initialized, clients do local training and model initialization (steps 3 and 4). A client loads the received model parameters from the server and then decrypts and loads the target model in their TEE. In step 5, the model parameters will be reported, and in step 6, the server will receive the model reporting. After receiving client updates, servers decrypt weights and parameters and perform secure aggregation and averaging inside their TEE. The server side does secure aggregation and broadcasts the model; steps 8 and 9 update, other participants. Steps 4-9 must be repeated until the model m converges or a certain amount of training rounds have been completed. Also, the model training and aggregation procedure is detailed in Algorithm 1 for both the client and server sides.

Algorithm 1 Workflow of TEE in AP2FL

```
Input: Global model ($m$), Number of clients: $N$, Data ($D$), Communication rounds ($R$), Local data $[X]$ and labels $[Y]$, Number of local training epochs: $E$
Output: Updated parameters of local model
  Aggregated final parameters:$\{P_0, P_1, \ldots, P_m\}$
Initialization step
  Server side:
  Initialize participating client list $T = []$
  Load D $\rightarrow$ m
for $n$=1 to $N$ do
    Build secure channel
    Data Transmission in the list $T$
    Initialize model m $\rightarrow$ TEE
end
Client-side:
for $e$=1 to $E$ do
    for $t$=1 to $T$ do
        Model configuration
        Local model initialization
    end
    model running $\rightarrow$ TEE
end
Training step in FL
for $r = 1$ to $R$ do
    for $\{x, y\}$ to $X$ do
        Local training
        local updating
    end
    Model reporting $\rightarrow$ TEE
end
Secure aggregate parameters:$\{P_0, P_1, \ldots, P_m\}$ $\rightarrow$ TEE
Model Broadcasting $\rightarrow$ all clients
Return Aggregated final parameters
```

The server initializes the global model using random weights or publicly available information. In this manner, the model m is prepared for broadcasting after initialization. Utilizing TEE, the server establishes secure communication with participating clients. During steps 1 and 2, data is transmitted to the model configuration through a secure channel. Following the model initialization, clients perform local training and model initialization (steps 3 and 4). A client loads the received model parameters from the server, decrypts them, and loads the target model into their TEE. In step 5, the model parameters are reported, and in step 6, the server receives the model reporting. Upon receiving client updates, servers decrypt weights and parameters, then carry out secure aggregation and averaging within their TEE. The server performs secure aggregation, broadcasts the model, and updates other participants in steps 8 and 9. Steps 4-9 must be repeated until the model m converges or a predefined number of training rounds are completed. The model training and aggregation procedure is also outlined in Algorithm 1 for both the client and server sides. It begins with initializing the global model using public data, establishing secure channels, and local training on the client side. It concludes with the server securely aggregating local updates and broadcasting the updated global model to all clients.

B. Addressing Non-IID Data in AP2FL

Conventional FL methods, such as FedAvg [21] and other approaches [28], often struggle to effectively handle non-IID data, particularly when clients possess limited data to train a model in FL settings. A significant contribution of AP2FL in healthcare environments is its ability to handle non-IID situations by incorporating ActPerFL [9] and BN in our DL models within the FL framework. Specifically, the data quality is evaluated for each client based on the statistical information obtained from multiple clients, which subsequently informs the update of the model's hyperparameters and BN layer.

ActPerFL represents a self-aware, personalized FL approach that intelligently balances the training of individual local models with the global model, indirectly contributing to the training of other clients' models [9]. BN integration in DL is crucial to improving the model's performance and managing domain shifts effectively. BN layers provide sufficient statistics, including mean and standard deviation, which are essential in the process [29]. To represent the data distributions of clients, BN is primarily utilized. Consequently, clients' feature distributions are maintained through local BN. Additionally, BN-related statistics are used to determine client similarity, enhancing support for non-IID scenarios through weighted aggregation.

1) Problem Formulation in AP2FL: In FL, there are N different clients (organizations or users), indicated as $\{H_1, H_2, \ldots, H_N\}$ and each client has its own dataset, i.e.,$\{D_1, D_2, \ldots, D_N\}$. Each dataset D_i includes two parts, i.e., a train dataset Di_{train} and Di_{test}. Therefore, the entire number of samples is: $n_i = ni_{train} + ni_{test}$ and $D_i = Di_{train} \cup Di_{test}$. The distributions of the datasets are all different, $P_{Di} \neq P_{Dj}$. Clients have their own models defined as $\{F_i\}_{i=1}^N$. To learn a good model, we aggregate information from all clients in order to learn f_i for the local dataset of each client D_i without private data leakage:

$$\min_{\{F_i\}_{i=1}^N} = \frac{1}{N}\sum_{1}^{N}\frac{1}{n_{\frac{te}{i}}}\sum_{j=1}^{n_{\frac{te}{i}}}\eta(Di) \tag{1}$$

where η is a loss function in Eq. (1).

2) Incorporating ActPerFL and Batch Normalization: To effectively apply ActPerFL, we need three critical components: 1- Appropriate initialization for local clients at each round, 2- Automatic determination of local training steps, and 3- Discrepancy-aware aggregation rules for the global model. Algorithm 2 outlines the overall process of ActPerFL and the BN approach. At round t, the parameters from client i are represented as $\omega_i^t = \alpha_i^t + \beta_i^t$. After updating ω_i^t with local data from the i-th client, the parameters become $\omega_i^{*t} = \alpha_i^{*t} + \beta_i^{*t}$. The $*$ notation indicates updated parameters. The server uses the updating strategy to aggregate data via Eq. (2):

$$\omega_i^{t+1} = \omega^*i^t \Rightarrow \alpha i^{t+1} + \beta_i^{t+1} = \alpha^*i^t + \beta^*i^t \tag{2}$$

To address this issue, the prediction layer aggregates mean and variance values over the entire training set. As training becomes distributed, each device has its local data, and

Algorithm 2 The ActPerFL and BN

```
1  Input: Global model $m$, client activity rate $C$, learning rate $\eta$,
   data of $N$ clients $D_1, D_2, \ldots, D_N$
2  Output: Client models $F_i$ for $i = 1, 2, \ldots, N$ using ActPerFL
   and BN
3  Initialize: Calculate empirical variance [9], set convergence
   criteria and maximum rounds Update global model $m$ while not
   converged or maximum rounds not reached do
4      for $n$=1 to $N$ do
5          Distribute global model $m$ to client $\rightarrow C$
6          Each client computes $D_i = D_{i,train} \cup D_{i,test}$
7          Compute initial parameters $\omega_i^{(t)} = \alpha_i^{(t)} + \beta_i^{(t)}$ while
           client i not converged do
8              Perform local training using SGD with BN
               Automatically determine local training steps
               Update local parameters to $\omega_i^{*(t)} = \alpha_i^{*(t)} + \beta_i^{*(t)}$
               Aggregate local parameters on the server using
               discrepancy-aware aggregation rules DUpdate
               global model $\rightarrow m$
9          end
10     end
11 end
```

the batch is also distributed. Each device trains on a local batch before global aggregation. During training, BN performs normalization on local batch statistics. This is crucial when handling non-identical data distributions, as each device's batch statistics do not represent global statistics, leading to prediction discrepancies.

The weights are evaluated by ω. We calculate only the statistics of BN layers' inputs. The BN statistics of the i^{th} client are acquired by Eq. (3):

$$(\varphi^i, \mu^j) = \left[\left(\varphi^{i1}, \mu^{j1}\right), \left(\varphi^{i2}, \mu^{j2}\right), \ldots, \left(\varphi^{in}, \mu^{jn}\right)\right] \tag{3}$$

The distance between two clients i and j is computed using the Kantorovich-Rubinstein metric. This metric, which measures the distance between clients' probability distributions, is particularly suited for FL due to its sensitivity to displacements in data distribution, robustness to noise, and adherence to the principles of a mathematical distance (symmetry and triangle inequality), providing a reliable measure of distribution discrepancies. The computation is based on Eq. (4):

$$d_{ij} = \sum_{l=1}^{L}\left(\left|\mu^j - \mu^i\right|^2 + \left|\varphi^j - \varphi^i\right|^2\right)^{\frac{1}{2}} \tag{4}$$

A large d_{ij} value indicates a significant distribution distance between the i-th and j-th clients. Therefore, the larger d_{ij} is, the less similar the two clients are, resulting in a smaller ω_{ij}.

C. Auditing Method in AP2FL

In AP2FL, auditing is essential to verify all clients' contributions of model gradients and weights to the FL server. This process, while complex, is necessary to ensure the global model accurately reflects diverse data types and distributions.

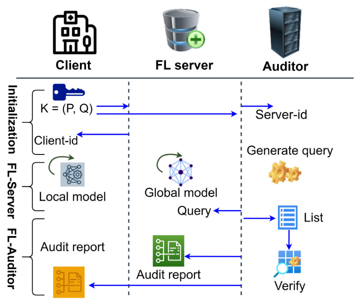

Fig. 3. Workflow of the AP2FL framework.

To streamline this process, we've established a comprehensive auditing protocol that integrates smoothly into the AP2FL framework.

1) Auditing Protocol: The protocol has been divided into three major phases that work harmoniously to provide reliable auditing:

- *Phase 1: Initialization*
  1) Each client, hospital, during initialization of its local models in TEE generates a public-private key-pair $K = (sk; pk)$ and public parameters based on the security parameter λ.
  2) Each client should register with public key (pk) in both FL server and auditor and get $Client - ID$.
  3) Auditor detects all FL servers in the FL setup and gets the list of FL server and their $server - ID$.
- *Phase 2: FL-Server*
  1) Run the FL environment, aggregating model parameters in the FL server and updating the global model's parameters.
  2) The FL server generates authentication tags, $SigVerif$, for each client that received its update, records this issue and counts the number of participants while updating the global model.
- *Phase 3: FL-Auditor*
  1) Auditor sends the query to the FL server. If the FL server had updated the global model, the server would like a response.
  2) The auditor gets the number of participants from the FL server after updating the global model.
  3) The auditor verifies the aggregation outcome in the FL server, recognizing participants via their public keys. This mechanism addresses any client's contention about the data included in the server's aggregation, and the auditor can validate or refute the claim.
  4) Lastly, the auditor generates an audit report summarizing the model update events and sends the response to the FL server and the participants, clarifying their participation in the global model update.

The auditing mechanism ensures transparency and validity in the FL process, confirming that all contributions are recognized and the global model update is accurately performed. Fig. 3 presents the auditing process in the AP2FL framework, displaying the interactions among clients, the FL server, and the auditor.

IV. Security Analysis

A. Threat Model and Defense Strategies

Incorporating TEE in our FL model equips us to address potential threats on both the client and server sides.

Client-side: These involve adversaries compromising client devices to access sensitive data during local model training or manipulate local model updates. *Defense Strategy:* TEE secure the model training and data processing within an isolated environment. Therefore, even in case of device compromise, sensitive information remains protected inside the TEE.

Server-side: These pertain to unauthorized attempts to access the global model or tamper with the server's aggregation process. *Defense Strategy:* TEE fortify security by performing client update aggregations in a secure server-side environment, preventing unauthorized manipulations or access to sensitive client updates. With the auditing protocol, we effectively combat these threats and preserve privacy in our FL model. We enhance our FL model's PP through TEE usage by ensuring secure local training, encrypted model updates, and secure aggregation. However, we do not explore TEE and their SDKs concerning side-channel and physical attacks.

B. Privacy-Preserving Proof

To formally prove privacy preservation for the threat models, we will define the security properties of the FL model and provide mathematical proof for the defensive strategies using TEE. Let D_i denote the sensitive data held by client i and M_i represent the local model updates computed by client i. Let A_{agg} be the aggregation function performed on the server, and M_{global} be the global model after aggregation.

Security Property 1 (Confidentiality): For any adversary $\mathcal{A}$ and client i, the probability that $\mathcal{A}$ can access sensitive data D_i or local model updates M_i should be negligible. Mathematically, we can define this property as Eq. (5):

$$Pr[\mathcal{A} \text{ can access } D_i \text{ or } M_i] \leq negl(\lambda) \tag{5}$$

where λ is the security parameter, and $negl(\lambda)$ represents a negligible function in λ.

Security Property 2 (Integrity): For any adversary, $\mathcal{A}$, the probability that $\mathcal{A}$ can manipulate the aggregation process A_{agg} or tamper with the global model M_{global} should be negligible.

Mathematically, we can define this property as Eq. (6):

$$Pr\left[\mathcal{A} \text{ can manipulate } A_{agg} \text{ or tamper with } M_{global}\right] \leq negl(\lambda) \tag{6}$$

Proof of Confidentiality and Integrity Using TEE: Since TEE provide a secure and isolated execution environment, we can assume that the probability of adversaries compromising a TEE is negligible. We denote this as Eq. (7):

$$Pr\left[\mathcal{A} \text{ can compromise TEE}\right], \leq negl(\lambda) \tag{7}$$

Now, it is feasible to demonstrate that the defensive tactics involving TEE meet the stringent security requirements of confidentiality and integrity.

Confidentiality: For client-side threats, TEE protect the sensitive data D_i and local model updates M_i. Thus, the probability that an adversary can access D_i or M_i is bounded by the probability of compromising the TEE, which is negligible.

Integrity: For server-side threats, TEE ensure that the aggregation process A_{agg} and global model M_{global} remain secure. Thus, the probability that an adversary can manipulate A_{agg} or tamper with M_{global} is also bounded by the probability of compromising the TEE, which is negligible.

Based on these proofs, we can conclude that the use of TEE in the FL model effectively addresses the identified threat models and ensures privacy preservation by satisfying the confidentiality and integrity security properties.

C. Effectiveness of Auditing Mechanism

Our proposed auditing mechanism fortifies the FL process with integral characteristics of integrity, transparency, fairness, and robustness while simultaneously ensuring privacy and security. This balance, fundamental to the mechanism's functionality, is supported by corresponding theorems and lemmas, affirming its effectiveness.

Theorem 1: Integrity and Transparency.

Proof: To prove the integrity and transparency of the auditing mechanism in the AP2FL framework, we need to show that the contributions of each client are accurately incorporated into the global model and that the FL server provides accurate information about clients' participation in the process. ■

Lemma 1 (Authentication Tags): The authentication tags, *SigVerif*, generated by the server can be used to verify the contributions of each client to the aggregation process, ensuring that their local model updates are properly accounted for. This guarantees the integrity of the FL process.

Lemma 2 (Query Response): The FL server's ability to provide accurate information about clients' participation in the FL process in response to a query from the auditor ensures transparency in the FL process.

Theorem 2: Fairness.

Proof: To prove the fairness of the auditing mechanism in the AP2FL framework, we need to show that the weights assigned to each client's contribution in the aggregation process are determined fairly based on their contributions. ■

Lemma 3: Weight Calculation: The auditor accurately calculates the weights W_i based on the distances δ_i, ensuring a fair representation of each client's contribution in the aggregation process. This guarantees the fairness of the FL process.

Theorem 3: Robustness.

Proof: To prove the robustness of the auditing mechanism in the AP2FL framework, we need to show that the auditing mechanism can identify and mitigate the impact of adversarial clients in the FL process. ■

Lemma 4: Adversarial Client Detection: The auditing mechanism can detect clients in $\mathcal{A}$ by identifying discrepancies in their contributions, such as malicious or unregistered clients. This contributes to the robustness of the FL process.

Lemma 5: Mitigating Adversarial Impact: By identifying and excluding clients' contributions in $\mathcal{A}$, the auditing mechanism can reduce the impact of adversarial clients on the global model, ensuring the robustness of the FL process.

V. Performance Evaluation

To evaluate the AP2FL framework, we have equipped our testing system with 16 GB DDR4 memory, RTX 2070 GPU, and Intel (R) Core(TM) i7 CPU-10700KF@ 3.80 GHz, which support Intel Soft Guard Extensions (SGX). To build our setup, we used libraries such as *PyTorch/torch* and integrated *PySyft* (Python library for secure and private Deep Learning) with Intel SGX as PySyft + Intel SGX [30]. The privacy risks associated with DL have been extensively studied in Convolutional Neural Networks (CNN). As a result, the proposed framework uses the LetNet5 [21] model, one of the most well-known CNN models in modern DL-based computer vision.

A. Datasets

The proposed framework was assessed using the MedMNIST collection [31], a standardized collection of biomedical images similar to MNIST that includes 12 2D datasets and 6 3D datasets. Each image in MedMNIST is preprocessed to 28×28 (2D) or $28 \times 28 \times 28$ (3D), making it accessible to users without background knowledge. From the 12 2D datasets in MedMNIST, we selected the three with the most classes: OrganMNIST Sagittal, OrganMNIST Coronal, and OrganMNIST Axial [31].

B. Experimental Analysis

As shown in Fig. 4, the distribution of samples across 20 clients is visualized for OrganMNIST Sagittal, Coronal, and Axial datasets. In MedMNIST, training data is used for training as well as validation data is used for fine-tuning hyperparameters, and test data is used for testing. Therefore, the LeNet5 [21] model uses these datasets for training and prediction. Assuming a learning rate of 10^{-2}, the models are trained using cross-entropy loss and SGD optimization. If there is no local update epoch setting, our default value is $E = 3$, where E means training epochs in one round. Our method's θ value is 0.5 since θ has little influence on accuracy and only affects convergence speed. The AP2FL model is compared with common FL methods and FL methods designed for non-ID data: *FedAvg* [21], all client models are aggregated without non-IID data. *FedProx* [24], Update FedAvg to include a proximal term and allow for partial information aggregation. *FedPer* [32], ayers are preserved locally by each client. *FedBN* [26], each client preserves the local BN.

Utilizing three distinct MedMNIST datasets, Table I, Table II, and Table III provide a detailed breakdown of classification results per client, respectively. The inclusion of an auditing mechanism in our proposed framework enables us to monitor each participant's contributions during the federated learning process. Analyzing these results, it is evident that our approach demonstrates superior performance in terms of accuracy, consistently outperforming other tested methods. In particular, while FedBN struggles to achieve satisfactory results due to its focus on handling feature shifts, our model successfully navigates label shifts via the ActPerFL, boosting its accuracy. The presented results in Table I, Table II, and Table III represent each user's optimal performance during runtime, respectively. The term 'Ave', used in our tables,

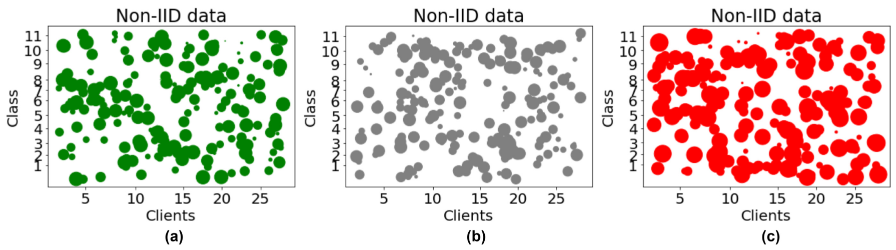

Fig. 4. Allocated samples to each client per class.

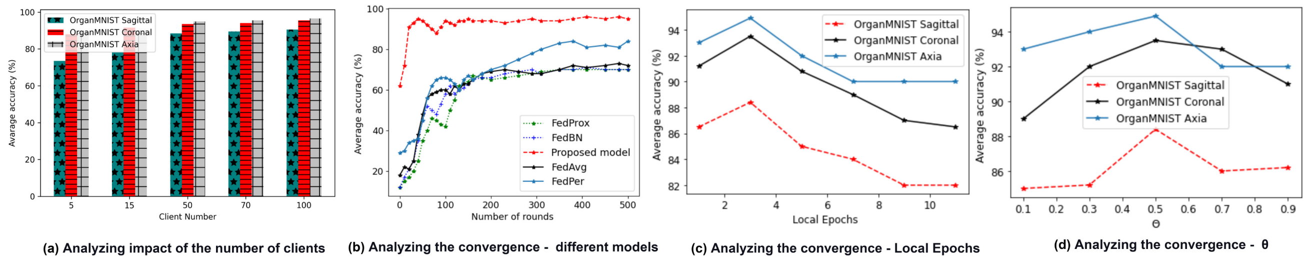

Fig. 5. Parameter sensitivity analysis.

TABLE I
ACCURACY OF USERS ON ORGANMNIST SAGITTAL

User	FedAvg	FedBN	FedProx	FedPer	Proposed model
5	47.39	71.87	66.3	85.74	93.51
10	68.45	78.8	52.3	85.74	90.69
15	78.2	91.78	71.25	64.56	88.75
17	36.55	45.04	41.93	58	98.5
20	41.3	53.48	83.57	92.42	89.1
Ave	64.8	80.44	64.9	74.55	88.4

TABLE II
ACCURACY OF USERS ON ORGANMNIST CORONAL

User	FedAvg	FedBN	FedProx	FedPer	Proposed model
5	79.22	88.7	79.73	77.87	92.8
10	85.83	96.46	80.61	80.07	98.82
15	81.23	61.38	90.39	51.52	91.89
17	67.45	83.81	73.99	65.37	99
20	54.56	90.71	88.18	70.61	89.5
Ave	78.36	88.18	78.3	71.14	93.5

TABLE III
ACCURACY OF USERS ON ORGANMNIST AXIA

User	FedAvg	FedBN	FedProx	FedPer	Proposed model
5	80.03	93.14	80.37	83.22	94.5
10	92.32	84.1	77.58	41.71	99.1
15	74.66	87.84	82.98	53.22	91.79
17	83.31	62.77	74.39	62.06	88.1
20	81.32	93.2	90.42	93.95	96.6
Ave	84.06	89.28	84.11	70.02	94.9

stands for 'Average', referring to the average value based on all participants.

1) Analyzing Client Number, Convergence, and Parameter Sensitivity: Fig. 5(a) underscores how variations in client distributions contribute to the differential difficulty levels experienced among clients. This underscores the value of using specific clients' local data to achieve particular distributions, further enhancing our method's accuracy. As the number of clients increases, accuracy increases because we have more parameters for aggregation (Fig. 5(a)). Moreover, we examine our method's convergence and parameter sensitivity. As shown in Fig. 5(b), our method almost converged after the 20th round. In contrast, other methods require more than 350 rounds. On three MedMNIST benchmarks, we tested AP2FL's parameter sensitivity by local epochs, and value of θ. In our presentation of parameter sensitivity, one parameter was changed, and the other parameters were fixed. Our method achieved acceptable results that according to the graph in Fig. 5(c), the best value for local epochs is 3, and our method is the best with this value. Insufficient communication between the clients and keeping the total number of epochs constant has caused it to decrease with more local epochs. Based on Fig. 5(d), our method's average accuracy and convergence rate are affected by θ. As $\theta = 0.5$, AP2FL consistently performs better.

TABLE IV
AUDITING SUMMARY BASED ON PARTICIPANTS

Query	Info	Valid	CF	Tag ID	RND
1	Get clients' IDs & Tags	Yes	0.9	—	1
4	Audit client# 5	Yes	0.05	7dc023b5	3
5	Audit client# 8	Yes	0.1	df73e6c7	4
9	Get clients' IDs & Tags	Yes	0.65	—	6
15	Audit client# 20	No	0.2	2b9100c6	5

C. Auditing Functionality

Table IV summarizes the information that can be inferred based on one of the datasets (OrganMNIST Axia) for analyzing the functionality of the audit protocol. The *Query* field refers to the queries sent from the Auditor to the FL server, and the *Info* field indicates the query context like #*Participants*. The *Valid* field indicates whether clients have registered on both the server and Auditor sides. The *Tag ID* fields indicate the total number of clients and their unique IDs during runtime. The *CF* field represents the ratio of clients updated to the server. The Auditor uses Euclidean distance to measure the difference between the parameter models for different clients and the global model. The *RND* field indicates the round in which the global model was updated. 'Audit client' represents individual clients or nodes participating in the FL process, each with a unique model contributing to the update of the global model. For example, if 25 clients participated in updating the global model during several rounds in a federated setup, the *Query* $= 1$ query would ask the FL server to provide all client IDs and tags, and the *Valid* field would be "Yes" since all clients were valid for updating the global model in *RND* 1. In this round, the *CF* field indicates that 90% of clients updated the server, indicating their significant impact on the global model. Similarly, in *Query* $= 15$, the Auditor audits *client* 20 and discovers that it has not registered with the Auditor but has participated in updating the global model on the FL server. The Auditor reports any malicious or unregistered clients and suspends that round of updating the global model.

VI. Conclusion and Future work

We proposed the Auditable Privacy-Preserving Federated Learning (AP2FL) framework for electronics in healthcare. that preserves privacy via Trusted Execution Environments (TEE) for secure aggregation and local training on the client and server sides. Also, to address the non-IID issue, we used the combination of ActPerFL and Batch Normalization (BN) to learn similarities between clients, automated tune local model parameters, and model aggregation. We also devised an auditing method to monitor the impact of each client in Federated Learning (FL) for averaging and updating the global model based on different data distributions. In future research, we will enhance the AP2FL with blockchain for training Machine Learning (ML) models and auditing, thereby increasing transparency and traceability. Furthermore, the potential single-point-of-failure issue in AP2FL with auditors could be mitigated through multiple independent auditors or a consensus-based approach.

References

[1] M. Wazid, A. K. Das, and S. Shetty, "BSFR-SH: Blockchain-enabled security framework against ransomware attacks for smart healthcare," *IEEE Trans. Consum. Electron.*, vol. 69, no. 1, pp. 18–28, Feb. 2023.

[2] B. Jayaraman and D. Evans, "Evaluating differentially private machine learning in practice," in *Proc. 28th USENIX Security Symp. (USENIX Security)*, 2019, pp. 1895–1912.

[3] Y.-T. Lee, W.-H. Hsiao, Y.-S. Lin, and S.-C. T. Chou, "Privacy-preserving data analytics in cloud-based smart home with community hierarchy," *IEEE Trans. Consum. Electron.*, vol. 63, no. 2, pp. 200–207, May 2017.

[4] X. Jiang, X. Zhou, and J. Grossklags, "Comprehensive analysis of privacy leakage in vertical federated learning during prediction," in *Proc. Privacy Enhanc. Technol.*, vol. 2022, no. 2, 2022, pp. 263–281.

[5] L. Melis, C. Song, E. De Cristofaro, and V. Shmatikov, "Exploiting unintended feature leakage in collaborative learning," in *Proc. IEEE Symp. Security Privacy (SP)*, 2019, pp. 691–706.

[6] J. Geiping, H. Bauermeister, H. Dröge, and M. Moeller, "Inverting gradients-how easy is it to break privacy in federated learning?" in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 16937–16947.

[7] D. Saraswat et al., "Blockchain-based federated learning in UAVs beyond 5G networks: A solution taxonomy and future directions," *IEEE Access*, vol. 10, pp. 33154–33182, 2022.

[8] J. Xu, B. S. Glicksberg, C. Su, P. Walker, J. Bian, and F. Wang, "Federated learning for healthcare informatics," *J. Healthcare Informat. Res.*, vol. 5, no. 1, pp. 1–19, 2021.

[9] H. Chen et al., "ActPerFL: Active personalized federated learning," in *Proc. Workshop Feder. Learn. Nat. Lang. Process. (ACL)*, 2022, p. 7.

[10] A. Yazdinejad, R. M. Parizi, A. Dehghantanha, and H. Karimipour, "Federated learning for drone authentication," *Ad Hoc Netw.*, vol. 120, Sep. 2021, Art. no. 102574.

[11] M. Nasr, R. Shokri, and A. Houmansadr, "Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning," in *Proc. IEEE Symp. Security Privacy (SP)*, 2019, pp. 739–753.

[12] A. Yazdinejad, A. Dehghantanha, R. M. Parizi, M. Hammoudeh, H. Karimipour, and G. Srivastava, "Block hunter: Federated learning for cyber threat hunting in blockchain-based IIoT networks," *IEEE Trans. Ind. Informat.*, early access, Feb. 7, 2023, doi: 10.1109/TCE.2023.3242375.

[13] K. Ganju, Q. Wang, W. Yang, C. A. Gunter, and N. Borisov, "Property inference attacks on fully connected neural networks using permutation invariant representations," in *Proc. ACM SIGSAC Conf. Comput. Commun. Security*, 2018, pp. 619–633.

[14] Z. Li, J. Zhang, L. Liu, and J. Liu, "Auditing privacy defenses in federated learning via generative gradient leakage," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 10132–10142.

[15] B. Zhao, K. R. Mopuri, and H. Bilen, "IDLG: Improved deep leakage from gradients," 2020, *arXiv:2001.02610*.

[16] H. Yin, A. Mallya, A. Vahdat, J. M. Alvarez, J. Kautz, and P. Molchanov, "See through gradients: Image batch recovery via gradinversion," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 16337–16346.

[17] P. Tiwari, A. Lakhan, R. H. Jhaveri, and T.-M. Gronli, "Consumer-centric Internet of Medical Things for cyborg applications based on federated reinforcement learning," *IEEE Trans. Consum. Electron.*, early access, Feb. 7, 2023, doi: 10.1109/TCE.2023.3242375.

[18] Y. Sun, J. Liu, K. Yu, M. Alazab, and K. Lin, "PMRSS: Privacy-preserving medical record searching scheme for intelligent diagnosis in IoT healthcare," *IEEE Trans. Ind. Informat.*, vol. 18, no. 3, pp. 1981–1990, Mar. 2022.

[19] F. Mo, H. Haddadi, K. Katevas, E. Marin, D. Perino, and N. Kourtellis, "PPFL: Privacy-preserving federated learning with trusted execution environments," in *Proc. 19th Annu. Int. Conf. Mobile Syst. Appl. Services*, 2021, pp. 94–108.

[20] W. Lu et al., "Personalized federated learning with adaptive batchnorm for healthcare," *IEEE Trans. Big Data*, early access, May 23, 2022, doi: 10.1109/TBDATA.2022.3177197.

[21] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. Artif. Intell. Stat.*, 2017, pp. 1273–1282.

[22] T. Ching et al., "Opportunities and obstacles for deep learning in biology and medicine," *J. Roy. Soc. Interface*, vol. 15, no. 141, 2018, Art. no. 20170387.

[23] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of FedAVG on non-IID data," 2019, *arXiv:1907.02189*.

[24] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," in *Proc. Mach. Learn. Syst.*, vol. 2, 2020, pp. 429–450.

[25] Y. Chen, X. Qin, J. Wang, C. Yu, and W. Gao, "FedHealth: A federated transfer learning framework for wearable healthcare," *IEEE Intell. Syst.*, vol. 35, no. 4, pp. 83–93, Jul./Aug. 2020.

[26] X. Li, M. Jiang, X. Zhang, M. Kamp, and Q. Dou, "FedBN: Federated learning on non-IID features via local batch normalization," 2021, *arXiv:2102.07623*.

[27] M. Müller, A. Simonet-Boulogne, S. Sengupta, and O. Beige, "Process mining in trusted execution environments: Towards hardware guarantees for trust-aware inter-organizational process analysis," in *Proc. Int. Conf. Process Min.*, 2022, pp. 369–381.

[28] H. Gao, A. Xu, and H. Huang, "On the convergence of communication-efficient local SGD for federated learning," in *Proc. AAAI Conf. Artif. Intell.*, vol. 35, 2021, pp. 7510–7518.

[29] Y. Li, N. Wang, J. Shi, X. Hou, and J. Liu, "Adaptive batch normalization for practical domain adaptation," *Pattern Recognit.*, vol. 80, pp. 109–117, Aug. 2018.

[30] "PySyft + Intel SGX." Accessed: Apr. 15, 2020. [Online]. Available: https://blog.openmined.org/pysyft-pytorch-intel-sgx/

[31] J. Yang, R. Shi, and B. Ni, "Medmnist classification decathlon: A lightweight AutoML benchmark for medical image analysis," in *Proc. IEEE 18th Int. Symp. Biomed. Imag. (ISBI)*, 2021, pp. 191–195.

[32] M. G. Arivazhagan, V. Aggarwal, A. K. Singh, and S. Choudhary, "Federated learning with personalization layers," 2019, *arXiv:1912.00818*.

Abbas Yazdinejad received the B.Sc. degree in computer engineering from the Shahid Bahonar University of Kerman, Iran, and the M.Sc. degree specializing in computer system architecture from the University of Isfahan, Iran. He is currently pursuing the Ph.D. degree with the Cyber Science Lab, Canada Cyber Foundry, School of Computer Science, University of Guelph, ON, Canada. He is also associated with the Decentralized Science Lab, Kennesaw State University, Kennesaw, GA, USA, and with the Smart Cyber-Physical Lab, University of Calgary. His research interests span cybersecurity, blockchain, federated learning, SDN, and IoT/IIoT.

Ali Dehghantanha (Senior Member, IEEE) is an Academic Entrepreneur in Cybersecurity, the Canada Research Chair in Cybersecurity and Threat Intelligence, and an Associate Professor of Cybersecurity with the University of Guelph, ON, Canada. He is the Director of Cyber Science Lab—a research lab dedicated to advanced research and training in cybersecurity—and the Director and a Founder of the Master of Cybersecurity and Threat Intelligence Program, University of Guelph.

Gautam Srivastava (Senior Member, IEEE) received the B.Sc. degree from Briar Cliff University, USA, in 2004, and the M.Sc. and Ph.D. degrees from the University of Victoria, Victoria, BC, Canada, in 2006 and 2012, respectively. He is currently a Full Professor with Brandon University, Brandon, MB, Canada, where he is currently active in various professional and scholarly activities. He also holds research positions with China Medical University, Taiwan, as well as Lebanese American University, Lebanon. He is popularly known as Dr. G. In his five years as a research academic, he has published a total of 180 papers in high-impact conferences in many countries and in high-status journals (SCI and SCIE) and has also delivered invited guest lectures on big data, cloud computing, Internet of Things, and cryptography at many universities worldwide. He is active in research in the field of cryptography, data mining, security and privacy, and blockchain technology. He is an editor of several SCI/SCIE journals. He is also an associate editor of many IEEE journals.