FedARCH: Enhancing Privacy-Preserving Brain Tumor Classification with Adaptive Reputation-aware Federated Learning and CKKS Homomorphic Encryption

Swetha Ghanta[1], Prasanthi Boyapati[2], Sujit Biswas[3], Ashok K Pradhan[4], and Saraju P Mohanty[5]

[1,2,4]**Department of Computer Science and Engineering, School of Engineering and Sciences, SRM University, AP, Guntur, Andhra Pradesh, India**
[3]**Computer Science Department, University of Northumbria at Newcastle, Newcastle, United Kingdom**
[3]**Computer Science, City, University of London, LONDON, United Kingdom**
[5]**University of North Texas, Denton, United States**

Corresponding author:
Sujit Biswas, Ashok Kumar Pradhan[3,4]

Email address: sujit.biswas@northumbria.ac.uk, ashokkumar.p@srmap.edu.in

ABSTRACT

Brain tumor diagnosis using MRI scans is critical for improving patient survival rates. However, automating the analysis of these scans faces significant challenges, including data privacy concerns and the scarcity of large, diverse datasets. A potential solution is Federated Learning (FL), which permits cooperative model training among multiple organizations without requiring the sharing of raw data, but it faces various challenges. To address these, in this work, we proposed FedARCH (Federated Adaptive Reputation-aware aggregation with CKKS Homomorphic encryption), a novel FL framework designed for a cross-silo scenario, where client weights are aggregated based on reputation scores derived from performance evaluations. Our framework incorporates a weighted aggregation method using these reputation scores to enhance the robustness of the global model. To address sudden changes in client performance, a smoothing factor is introduced, while a decay factor ensures that recent updates have a greater influence. These factors work together for dynamic performance management. Additionally, we address potential privacy risks from model inversion attacks by implementing a simplified and computationally efficient CKKS homomorphic encryption, which allows secure operations on encrypted data. With FedARCH, encrypted model weights of each client are multiplied by a plaintext reputation score for weighted aggregation. Since we are multiplying ciphertexts by plaintexts, instead of ciphertexts, the need for relinearization is eliminated, efficiently reducing the computational overhead. FedARCH achieved an accuracy of 99.39%, highlighting its potential in distinguishing between brain tumor classes. Several experiments were conducted by adding noise to the clients' data and varying the number of noisy clients. An accuracy of 94% was maintained even with 50% of noisy clients at a high noise level, while the standard FedAvg accuracy dropped to 33%. Our results and the security analysis demonstrate the effectiveness of FedARCH in improving model accuracy, its robustness to noisy data, and its ability to ensure data privacy, making it a viable approach for medical image analysis in federated settings. The FedARCH GitHub repository link is https://github.com/gswetha697/FedARCH

Keywords: Federated Learning, Brain tumor classification, Reputation, CKKS, Homomorphic Encryption

INTRODUCTION

Brain tumors are a very critical condition, and immediate identification and treatment are required to improve patient survival rates. Diagnosis of brain tumors is often done using MRI and CT scans. MRI scans are usually preferred over CT scans because they do not cause radiation exposure. Tumors can be

of two types: benign and malignant. Malignant tumors are cancerous and require immediate treatment, while benign tumors are non-cancerous but necessitate frequent tests and patient monitoring.

Analyzing MRI scans is crucial in this context, but it is often time-consuming and requires expertise. Automating brain MRI image analysis presents several challenges. The major challenge is the availability of datasets; medical institutions often do not share their patient data to protect privacy. To address this, a new paradigm called Federated Learning (FL) (McMahan et al. (2017)) has emerged as a solution, where only model weights are shared instead of raw data, thereby preserving patient data privacy. In an FL framework, there is a central server, often represented by a cloud environment, that holds a global Deep Learning (DL) model. Multiple clients, each representing a medical institution, have their local data and a copy of the global model, referred as local model. Each client trains the local model with their local data and only shares the model weight updates with the central server, preserving patient data privacy. The central server aggregates the weights received from each client using the Federated Averaging (FedAvg) algorithm and updates the global model, which is then sent back to all clients. This process is repeated for a certain number of rounds or until convergence.

Although this approach seems to offer a solution, there are several issues associated with its real-time application. For example, these frameworks require a large amount of data, which is not always possible in the medical domain, as some medical conditions can be extremely rare and underrepresented. To address this problem, we utilize transfer learning (TL). By employing pre-trained models, we can leverage existing knowledge and adapt it to our specific problem with limited data. This approach helps mitigate the challenge of data scarcity by fine-tuning models on small, specialized datasets, thereby improving performance even when the amount of local data is limited (Khan et al. (2022b)).

In FL, the global model is trained using the weights received from the clients. However, if a client sends malicious or erroneous data to the central server, which treats all clients equally, the global model will use these erroneous updates for aggregation. This can eventually corrupt the global model and affect all clients. Several works (Fan et al. (2023), Kang and Ahn (2023)) have been proposed to address this problem, but most are based on a cross-device scenario rather than a cross-silo scenario. A cross-device scenario involves IoT devices, where the number of devices is large but their computational ability is limited. In contrast, a cross-silo scenario involves organizations, where the number of entities is smaller but their computational ability is higher. For our use case, we consider a cross-silo scenario where multiple medical institutions collaborate for federated learning. In a cross-device scenario, existing solutions often reject the weights from underperforming clients and only consider the weights from the best-performing clients. This approach is feasible in cross-device scenario because the server can choose from a large pool of clients. However, in a cross-silo scenario, where the number of clients is already limited, completely rejecting a client's update can increase bias towards certain clients, ignoring others.

We propose FedARCH, a novel framework where reputations are assigned to each client based on their performance evaluation. Instead of using a simple FedAvg approach, where all the model weights are aggregated using a simple average, a reputation-based weighted aggregation is employed. This process is iterated after each round of training, as client performance and, therefore, reputations can change after any round. To prevent sudden changes in client performance from unduly affecting the assigned reputations, we have implemented a smoothing factor. This factor stabilizes the reputation adjustments, preventing abrupt increases or decreases from causing significant fluctuations. Additionally, as the training progresses across multiple rounds, it is important that more recent performance updates have a greater influence on the reputation, while older updates should gradually diminish in impact. To achieve this, we incorporate a decay factor that reduces the weight of older reputations, allowing the system to adapt to the recent client performances. We will discuss these details in the upcoming sections.

Another potential issue in FL is the model inversion attack (Fredrikson et al. (2015)), where a malicious actor can reconstruct the original data from the shared weights, thus compromising privacy. To address this problem, researchers developed homomorphic encryption (HE), which allows aggregation to be performed on encrypted data without decrypting it. In FedARCH, we used CKKS HE (Cheon et al. (2017)), a somewhat homomorphic encryption scheme (SHE). We have specifically chosen CKKS over other HE schemes like RSA and Paillier, because:

- CKKS is based on the hardness of Ring Learning With Errors (RLWE) problem, which is considered to be quantum-resistant offering security against potential quantum attacks while enabling efficient encrypted computations.

- CKKS allows a limited number of both addition and multiplication operations on encrypted data, which is necessary for our weighted aggregation, unlike other HE schemes that support only one of the two operations.

Some of the other popular RLWE-based HE schemes include BGV and BFV (Brakerski (2012), Brakerski et al. (2014)). However, CKKS HE was selected because it operates on approximations, significantly enhancing computational efficiency. CKKS can handle real numbers, enabling it to support the complex arithmetic required for our model. This approximate arithmetic capability makes CKKS faster compared to other schemes that operate on exact arithmetic, providing a good balance between security and performance for our proposed FedARCH framework.

CONTRIBUTIONS OF THE CURRENT PAPER

Motivation

Most existing FL research focuses on cross-device scenarios, which involve numerous simple IoT devices or mobile phones with limited computational capabilities and intermittent connectivity. These studies typically assume high dropout rates, ignore underperforming clients, and don't provide feedback on client contributions. While these assumptions may suit cross-device FL, they are not applicable to cross-silo FL, where multiple organizations, such as medical institutions, collaborate with substantial, valuable data.

In contrast to cross-device FL, the stakes are notably higher for cross-silo FL. Here, each client represents an organization, contributing critical and sensitive data, especially relevant in domains like healthcare. Ignoring any client, even an underperforming one, risks losing essential data. Organizations in this setting are generally more reliable and experience lower dropout rates than individual devices, making it essential to devise sophisticated approaches to handling client contributions effectively. Furthermore, providing performance feedback to clients in cross-silo FL can help organizations understand their contribution's impact on the global model. Such feedback enables institutions to improve their local models and strengthen future contributions to the global model.

Problem Addressed

FL applications in medical image analysis face multiple challenges that limit their potential. Key issues include untrusted third-party servers, inadequate client data validation, calculating accurate client reputations, and managing dynamic performance variations. Many existing solutions only address one or a few of these challenges, often at the cost of overall system performance and increased computational overhead. For FL to be fully effective in sensitive fields like healthcare, these challenges must be addressed in a unified manner without sacrificing model performance.

Solution Proposed

We propose FedARCH, a novel FL framework that evaluates each client's contribution before incorporating it into the global model, using an adaptive reputation mechanism with smoothing and decay factors to maintain dynamic, reliable reputations. This adaptive reputation mechanism factors in both recent and historical performance, ensuring that contributions remain meaningful over time while mitigating the influence of sudden performance changes.

To address the challenge of the untrusted server, we employ the CKKS HE technique, which enables secure operations on encrypted weights, thereby protecting the data from model inversion attacks. CKKS is particularly well-suited as it supports both addition and multiplication operations on real numbers, a feature that other HE schemes lack. This setup allows the server to work exclusively with encrypted data without needing decryption, maintaining data privacy. The computational overhead associated by using CKKS HE is reduced by using the plaintext-ciphertext multiplications instead of ciphertext-ciphertext multiplications. This greatly reduces the ciphertext growth and noise accumulation.

Novelty and Significance of the Solution

FL holds tremendous potential to automate medical image analysis, yet its benefits in critical fields are hindered by ongoing security and performance challenges. FedARCH addresses these issues comprehensively without compromising model accuracy.

The primary contributions of this work include:

1. FedARCH, an innovative cross-silo FL framework – Featuring adaptive reputation-based weighted aggregation for real-time performance management, particularly useful in classifying brain tumors from MRI scans.

2. Client performance evaluation – Using validation reports from neighboring clients, the system provides feedback to underperforming clients, encouraging continuous improvement.

3. Incorporation of optimized CKKS HE – This approach effectively guards against model inversion attacks from an untrusted server without compromising on computational efficiency.

4. Extensive simulations with variable client performance – Compared with the standard FL algorithm, FedARCH demonstrates superior performance, especially in scenarios with multiple underperforming clients.

The proposed framework advances the field by enhancing both security and model performance, particularly in high-stakes applications like medical imaging.

RELATED PRIOR RESEARCH

Table 1. Comparison of Features Across Different References

Reference	Dataset	Approach	Accuracy
Khan et al. (2022b)	Figshare	23-layer CNN	97.8%
Mathivanan et al. (2024)	Kaggle	MobileNetV3	99.75%
Rasool et al. (2022)	Kaggle	GoogleNet-SVM	98.01%
Senan et al. (2022)	SARTAJ	AlexNet-SVM	95.10%
Khan et al. (2022a)	Kaggle	Hierarchical Deep Learning-Based Brain Tumor (HDL2BT) classification	92.13%
Lamrani et al. (2022)	Kaggle	CNN model for binary classification	96.33%
Gaur et al. (2022)	SARTAJ	CNN and Explainable AI	94.64%
Vidyarthi et al. (2022)	Kaggle	NN classifier with Cumulative Variance method (CVM) for feature selecion	95.86%
Albalawi et al. (2024)	Kaggle	VGG with FL	98%
Islam et al. (2023)	Kaggle	Voting Ensemble of 6 TL models	With FL 91.05% Without FL 96.68%
Viet et al. (2023)	Figshare	VGG with FL	98.69%
Ay et al. (2024)	-	FedAvg	85.55%
Zhou et al. (2024)	SARTAJ	FL with EfficientNetB0 and ResNet50	80.17% 65.32%

With the advent of DL and Convolutional Neural Networks (CNNs), there are several research papers published to address the problem of brain tumor classification using DL techniques. A 23-layer CNN model was proposed for brain tumor classification on the Figshare dataset, while TL was also applied to address a binary classification task on a smaller Harvard dataset (Khan et al. (2022b)). To further leverage TL, an ensemble approach was employed for feature extraction across multiple TL models, combining the top three models using a Multi-layer Perceptron (MLP) (Remzan et al. (2024)). For the same classification problem, YOLOv7 was utilized, incorporating a Convolutional Block Attention Module (CBAM) to enhance feature extraction (Abdusalomov et al. (2023)).

Although these approaches generate high-performing accuracies, they are based on simple local learning models trained on smaller datasets, which may lack generalizability when applied to different datasets. Centralized learning, where all data is collected and processed at a single location, poses additional challenges, including the risk of a single point of failure and reluctance from organizations to participate due to concerns about patient data privacy. To address these issues, researchers introduced FL,

a collaborative learning technique that preserves patient privacy by working with model updates rather than raw data.

FL has gained significant attention as an approach to train models across decentralized devices or institutions while preserving data privacy. Initially, the FedAvg algorithm was introduced, enabling local models to be trained independently on each client and subsequently aggregated using a simple average to form a global model that synthesizes knowledge from all clients (McMahan et al. (2017)). Building on this foundation, FL was first applied to medical image analysis, demonstrating its potential in sensitive domains (Sheller et al. (2020)). To further enhance FL's performance, ensemble and voting techniques were integrated to improve classification accuracy in complex datasets (Islam et al. (2023)). Additionally, TL techniques were combined with FL specifically for brain tumor classification, allowing the model to be evaluated across various client contribution levels and performance metrics, thus highlighting the adaptability of FL in handling diverse data distributions (Viet et al. (2023)).

While effective, model inversion attacks (Fredrikson et al. (2015)), pose a significant threat to FL systems by reconstructing sensitive data from model updates. Various defense mechanisms have been considered, including differential privacy (Dwork et al. (2014)) and secure multi-party computation (Zhao et al. (2019)), but these often come with trade-offs in terms of computational overhead and model accuracy. To address these challenges and preserve data privacy, secure aggregation techniques were explored to ensure that the central server cannot access individual model updates (Bonawitz et al. (2017)). Recent advancements, such as the use of HE (Cheon et al. (2017)), enable computation on encrypted data, eliminating the need for decryption in a zero-trust architecture and further enhancing privacy in FL systems. The SHE approach was employed for cancer image analysis, incorporating an additive secret sharing technique (Truhn et al. (2024)). But since all clients are treated equally and their updates are aggregated to update the global model, ignoring the issue of underperforming clients can affect the performance of the final model.

To address client contribution disparity, weighted aggregation was utilized based on a data quality factor, along with the EL-Gamal HE technique (Zhang et al. (2022)). Since EL-Gamal supports multiplicative homomorphism, the encryption scheme was modified to enable additive homomorphism, thereby reducing communication overhead. The FedRaHa framework was proposed, incorporating reputations for client selection based on cosine similarity scores and employing hierarchical aggregation to reduce communication overhead (Panigrahi et al. (2023)). A Lightweight Privacy-preserving Federated Learning (LPBFL) scheme was introduced to calculate the reputation of each client prior to aggregating their updates into the global model, thereby preventing malicious updates from poisoning the final model. This scheme utilized Paillier HE to maintain the privacy of local model updates (Fan et al. (2023)). Paillier is a partial HE scheme, which supports only either of addition or multiplication operations, and it is considerably slow. A Genetic Algorithm approach was proposed to optimize client selection for FL, with communication cost minimization as the objective function (Kang and Ahn (2023)). The use of GA can significantly increase the training time and is not suitable for larger datasets and huge number of clients. A private blockchain-based framework was considered for storing model weights in chunks rather than directly, with miners tasked with evaluating the quality of local updates (Bhatia and Samet (2023)). The major limitation of this work is the scalability, if the number of the clients increase, then the number of weights will increase predominantly, thus making the idea of storing the weights in blockchain inefficient. Table 1 summarizes various existing work in the field of brain tumor classification task. In summary, while significant progress has been made in addressing data privacy, model robustness, and client heterogeneity in FL, challenges remain, particularly in cross-silo scenarios. FedARCH builds on these foundations by introducing reputation-based weighted aggregation, smoothing and decay factors for dynamic performance management, and the integration of CKKS HE to enhance privacy and security. CKKS HE, in particular, is notable for its efficient handling of approximate arithmetic, making it especially suitable for FL applications.

PRELIMINARIES

Federated Learning

FL is a latest trending paradigm in the machine learning community, offering solutions to several problems such as data scarcity, data privacy preservation, and real-time collaborative learning. FL has gained significant accolades for its capability to allow multiple parties to collaborate and train a global model without sharing their raw data, instead sharing the weight updates. This replaced the centralized learning

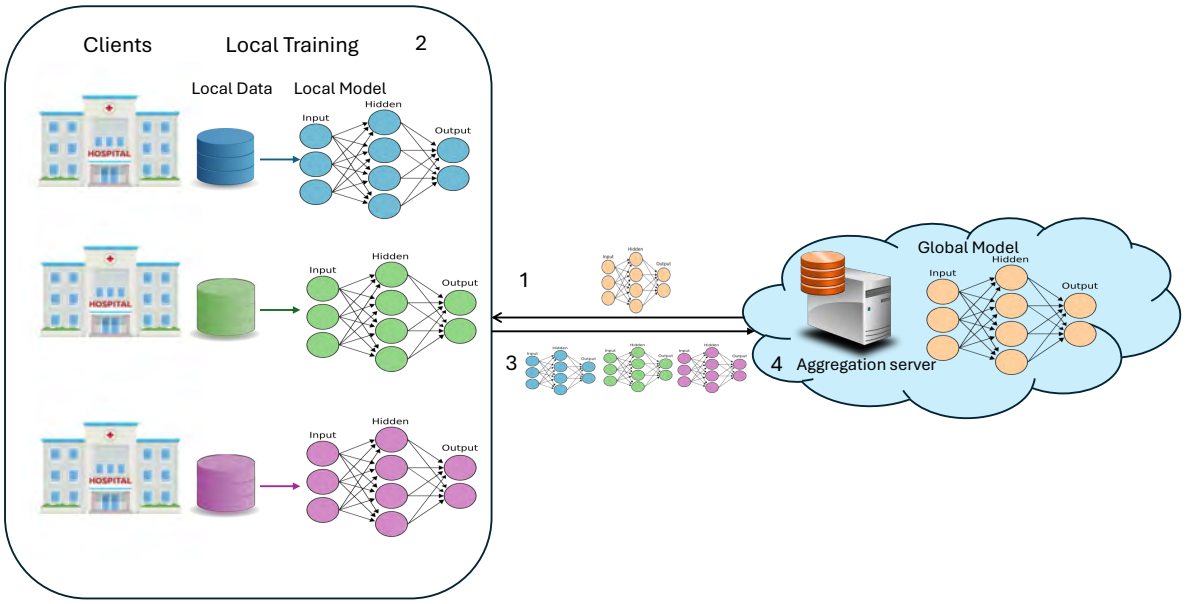

Figure 1. FL architecture

scenario, where data from multiple clients is collected in a cloud server and is used to train a global model, which also resides in the same cloud. FL implementation requires the following components and is illustrated in Figure 1:

- Server: A server is a cloud environment that holds the global model and acts as an aggregation server, aggregating the weights from clients.
- Client: A client can be any organization or medical institution in a cross-silo scenario, while in a cross-device scenario it can be any device like mobiles, IoT devices, etc.
- Global Model: In FL, multiple clients collaboratively train a global model, which can be any ML or DL model.
- Local Model: Each client receives a replica of the global model to train on its local data and at the client side it is referred as local model.
- Model Weights: When training the local model with local data, model weights are obtained. These weights represent the learned parameters of the neural network, determining the importance of input features, controlling the strength of neural connections, and encapsulating the model's knowledge gained from the training data.

The typical workflow of FL involves the server distributing the initial global model to all clients. Each client trains the model on its local data, updates the model weights, and sends these updates back to the server. The server aggregates the updates to form a new global model, which is then redistributed to the clients. This process repeats for a predefined number of rounds or until convergence.

CKKS Homomorphic Encryption

To further enhance data privacy and security in FL, especially when dealing with a curious server that might attempt to infer sensitive information from model updates, we employ CKKS HE (Cheon et al. (2017)). CKKS (Cheon-Kim-Kim-Song) is a type of somewhat homomorphic encryption scheme that supports arithmetic operations on encrypted data without needing to decrypt it, ensuring that data remains secure during the aggregation process. Key components of CKKS HE within FL include:

- CKKS Context: This holds parameters such as the polynomial modulus degree, scaling factor, security parameters, and the public-secret key pair. It defines the encryption scheme's environment, setting up the structure for encryption, decryption, and homomorphic operations.

- Message Encoding and Decoding:
Encoding: In CKKS, real numbers are encoded into a polynomial ring to enable encrypted operations (Huynh (2020)). The message m is transformed into a plaintext polynomial $\Delta m(x)$, where Δ is a scaling factor used to maintain precision during homomorphic operations by converting floating-point numbers to large integers. This is done by multiplying the floating-point numbers by the scaling factor before encryption, enabling accurate representation within the encryption scheme. This encoding maps the message into the ring

$$R = Z[x]/(x^N + 1) \tag{1}$$

where Z represents integers, $(x^N + 1)$ is a cyclotomic polynomial, and N is the polynomial degree, usually represented as powers of 2:

$$m \rightarrow \Delta m(x) \in Z[x]/(x^N + 1) \tag{2}$$

This polynomial representation allows CKKS to perform homomorphic operations like addition and multiplication on encrypted data, with the operations corresponding to similar operations on plaintext polynomials.
Decoding: The reverse process that maps the polynomial back to real numbers.

- Key Generation: Generate public and private keys: (pk, sk), where pk is used for encryption and sk for decryption. Each plaintext polynomial is encrypted using a public key, resulting in ciphertexts of the form: $c_1 = (c_{1,0}, c_{1,1})$ and $c_2 = (c_{2,0}, c_{2,1})$, where $c_{i,j}$ is a polynomial in $Z_q[x]/(x^N + 1)$

- Encryption: Given a plaintext polynomial $m(x)$, the encryption function using public key $pk = (a, b)$ and a random noise e generates a ciphertext c, a pair of polynomials where,

$$c = Enc(m(x), pk) = (c_0, c_1) \tag{3}$$

$$c_0 = b.s + m + e_0 \tag{4}$$

$$c_1 = a + e_1 \tag{5}$$

- Homomorphic operations:
Both addition and multiplication operations are performed on the ciphertexts, producing encrypted results that approximate the arithmetic operations on the underlying plaintexts.
Addition:

$$Enc(m_1(x), pk) + Enc(m_2(x), pk) = Enc(m_1(x) + m_2(x), pk) \tag{6}$$

$$Enc(m_1(x) + m_2(x), pk) = (c_{1,0} + c_{2,0}, c_{1,1} + c_{1,2}) \tag{7}$$

where $(c_{1,0}, c_{1,1})$ and $(c_{2,0}, c_{2,1})$ are ciphertexts for $m_1(x)$ and $m_2(x)$ respectively.
Multiplication:
When two ciphertexts are multiplied, it is not as straightforward as addition, the polynomial representations of ciphertexts expand:

$$c_{mul} = c_1 * c_2 \tag{8}$$

since each ciphertext is a tuple (c_0, c_1), the multiplication expands as follows:

$$c_{mul} = (c_{1,0}, c_{1,1}) * (c_{2,0}, c_{2,1}) \tag{9}$$

$$c_{mul} = (c_{1,0}c_{2,0}, c_{1,0}c_{2,1} + c_{1,1}c_{2,0}, c_{1,1}c_{2,1}) \tag{10}$$

This results in a new third-term ciphertext, i.e.,

$$c_{mul} = (c'_0, c'_1, c'_2) \tag{11}$$

The ciphertext is expanded in degree, like here in this example it is 3, further it will increase to 5, 9 and so on. To prevent this, relinearization is required to bring it back to the standard 2-term format and maintain the size of the ciphertext. But, it further increases the computational complexity and overhead.

- Decryption: Given a ciphertext c, the decryption function returns the plaintext polynomial

$$m(x) = Dec(c, sk). \tag{12}$$

In FedARCH framework, encrypted model weights (E_i^t) of the client i at round t are multiplied by a plaintext normalized reputation score ($\bar{R}_i^t$) for weighted aggregation. Since we multiply ciphertexts by plaintexts, rather than by other ciphertexts, relinearization is not required. Relinearization, typically used in HE schemes, manages the growth of ciphertext size and complexity after multiplying ciphertexts together. By avoiding the need for relinearization, we simplify our computational process and reduce overhead. These weights from different clients are further added together using ciphertext addition, which is a straightforward operation in CKKS.

Integrating CKKS HE into our FL framework provides a robust solution to protect sensitive client data from potential privacy breaches by the central server. This approach ensures that even if the server is compromised or curious, it cannot access or infer the original data, thus maintaining the confidentiality and privacy of each client's data throughout the training process.

FEDARCH FRAMEWORK

Adaptive Reputation-aware weighted aggregation

We propose FedARCH, a novel FL framework for collaborative learning in a cross-silo scenario. In this framework, we created a simulated environment with 10 clients, where each client represents a medical institution. A central server, referred to as the aggregation server, holds the global model used for the FL process. The server performs the aggregation of client weights after each FL round, and this process is repeated until the specified number of rounds is reached.

In this scenario, we assume the server is not trustworthy and it could perform a model inversion attack to obtain the original data, thus being termed as "curious" server. We also assume that clients are trustworthy, meaning they do not perform a model inversion attack or intentionally send malicious or erroneous updates. However, clients can still underperform due to several reasons:

Data Heterogeneity: Clients have different data distributions. For example, medical institutions may have varying case severities, leading to differences in model performance.

Resource Constraints: Some clients might have limited computational resources, resulting in less effective training.

Model Training Issues: Suboptimal hyperparameter settings, insufficient training epochs, or software bugs can cause variations in local model performance.

Environmental Factors: Factors like network latency or power outages could impact the training process for some clients.

Data Quality: Variations in data quality across clients, such as noisier or less representative data, can lead to poorer model performance.

Table 2. Notations Used in Federated Learning with Reputation and CKKS Encryption

Notation	Description
$\mathscr{D}_{train}$	Training dataset
$\mathscr{D}_{val}$	Validation dataset
$\mathscr{D}_{test}$	Testing dataset
N	No. of clients
R	No. of rounds
α	Smoothing factor for reputation update
β	Decay factor for reputation update
W^0	Initial global model weights
W^t	Global model weights at round t
W_i^t	Local model weights of client i at round t
E_i^t	Encrypted local model weights of client i at round t
R_i^t	Reputation of client i at round t
$\bar{R}_i^t$	Normalized reputation of client i at round t
P_i^t	Validation score of client i at round t
P_{prev}^t	Validation score of the previous client at round t
E_{prev}^t	Encrypted local model weights of the previous client at round t
W_{prev}^t	Local model weights of the previous client at round t
$context$	CKKS encryption context

Algorithm 1 Federated Learning with Reputation and CKKS Encryption

Require: Training dataset $\mathscr{D}_{train}$, Validation dataset $\mathscr{D}_{val}$, Testing dataset $\mathscr{D}_{test}$, Number of clients N, Number of rounds R, Smoothing factor α, Decay factor β, CKKS context $context$

Ensure: Final global model W^R

Split $\mathscr{D}_{train}$ among N clients
Initialize global model W^0
Initialize reputations $R_i^0 \leftarrow 1$ for all clients i
Distribute $\mathscr{D}_{val}$ to all N clients
for $t = 0$ to $R-1$ **do**
  for each client i **do**
    Train local model and obtain W_i^t
    $E_i^t \leftarrow \text{CKKSEncryption}(W_i^t, context)$
    Send E_i^t to client $(i+1) \mod N$
  end for
  for each client i **do**
    Call VALIDATION($E^t_{(i-1+N) \mod N}$, context)
  end for
  Collect all E_i^t and validation scores P_i^t at the global server
  Call UPDATEREPUTATION($P_i^t, R_i^t, \alpha, \beta$)
  Update global model weights:
  $E^{t+1} = \sum_{i=1}^{N} \bar{R}_i^{t+1} \cdot E_i^t$ (Aggregate weights)
  for each client i **do**
    $W^{t+1} \leftarrow \text{CKKSDecryption}(E^{t+1}, context)$
  end for
end for
$W^R \leftarrow W^{t+1}$
Evaluate final global model W^R on $\mathscr{D}_{test}$

Algorithm 2 Validation

Require: Encrypted weights E^t_{prev}, CKKS context $context$
Ensure: Validation score P^t_{prev}
1: $W^t_{prev} \leftarrow \text{CKKSDecryption}(E^t_{prev}, context)$
2: Validate the model using $\mathscr{D}_{val}$
3: Store validation score P^t_{prev}

Algorithm 3 UpdateReputation

Require: Validation scores P^t_i, Reputations R^t_i, Smoothing factor α, Decay factor β
Ensure: Updated and normalized reputations $\bar{R}^{t+1}_i$
1: **for** each client i **do**
2: $\quad R^{t+1}_i = (\alpha \cdot R^t_i + (1-\alpha) \cdot P^t_i) \cdot \beta$
3: **end for**
4: Normalize reputations $\bar{R}^{t+1}_i = \frac{R^{t+1}_i}{\sum_{j=1}^{N} R^{t+1}_j}$

Algorithm 4 CKKS Encryption

Require: Local model weights W_i, CKKS context $context$
Ensure: Encrypted local model weights E_i
1: $E_i \leftarrow \{\}$
2: **for** each layer k in W_i **do**
3: $\quad vector \leftarrow \text{Flatten}(W_i[k])$
4: $\quad E_i[k] \leftarrow \text{CKKSEncrypt}(vector, context)$
5: **end for**
6: **return** E_i

Algorithm 5 CKKS Decryption

Require: Encrypted local model weights E_i, CKKS context $context$
Ensure: Decrypted local model weights W_i
1: $W_i \leftarrow \{\}$
2: **for** each layer k in E_i **do**
3: $\quad decrypted\_vector \leftarrow \text{CKKSDecrypt}(E_i[k], context)$
4: $\quad W_i[k] \leftarrow \text{Reshape}(decrypted\_vector)$
5: **end for**
6: **return** W_i

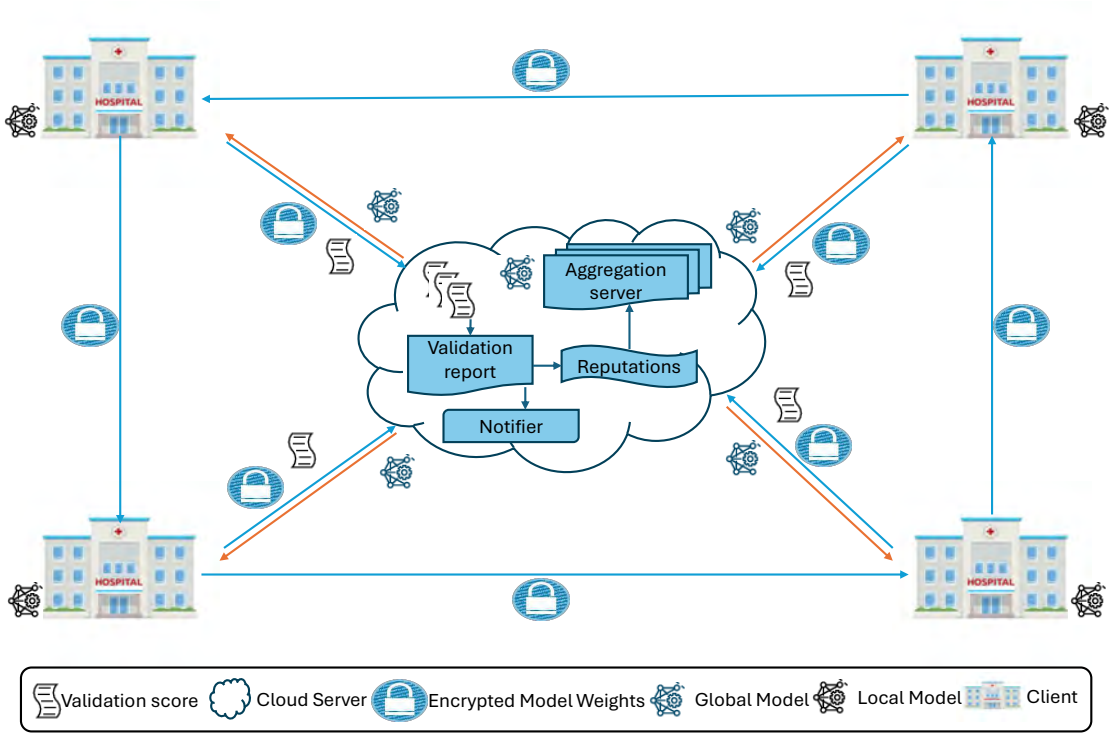

Figure 2. FedARCH architecture

By considering these factors, our proposed framework aims to accommodate and adjust for underperforming clients through reputation-based weighted aggregation, smoothing, and decay factors, ensuring that the global model remains robust and accurate despite these variations.

Each client trains the global model, enhancing decision-making by participating in the FL process with local data while ensuring data privacy by sharing only the model weights. In FedARCH framework, we use the pre-trained ResNet18 model (He et al. (2016)) and fine-tune it for our specific use case. A replica of the global model W^0 is shared with all clients. Upon receiving the model, each client C_i trains it with their local data D_{train}. The local model weights W_i^t are generated at each client, and these weights are encrypted using CKKS HE to preserve privacy from a curious server. Figure 2 provides an overview and Figure 3 a detailed illustration of the proposed FedARCH framework.

Each client C_i shares its encrypted local model weights E_i^t with the server for aggregation and with the next client C_{i+1} for validation. In this framework, each client C_i also acts as a validator for its previous client C_{i-1}. Specifically, client C_i validates the local weights E_{i-1}^t of the previous client C_{i-1} using the validation data D_{val} and generates a validation score (val_score) P_{prev}^t for that previous client, which is then sent to the server. The next client, upon receiving the previous client's encrypted local model weights E_{prev}^t, decrypts them using CKKS decryption to obtain the local model weights W_{prev}^t. To facilitate this, we assume that all clients share a common encryption scheme with a public-private key pair managed by a trusted authority. This ensures that each client can securely decrypt the weights from the previous client using the shared private key. This validation mechanism provides an additional layer of accountability and accuracy, reducing potential biases and ensuring a more comprehensive evaluation of the model's performance across various datasets.

Upon receiving the val_scores from all clients, the server's notifier informs underperforming clients if their validation score falls below a threshold value, defined as the average of the validation accuracies of all clients in that round. This notification helps clients take appropriate measures to improve their local data or training processes. Although clients could validate themselves, the notifier is necessary because clients do not have access to the validation accuracies of other clients to calculate this threshold. As a part of the server, the notifier ensures that clients receive the necessary feedback to enhance their performance. The server then assigns a reputation value R_i^t to each client using the val_scores. These reputations are

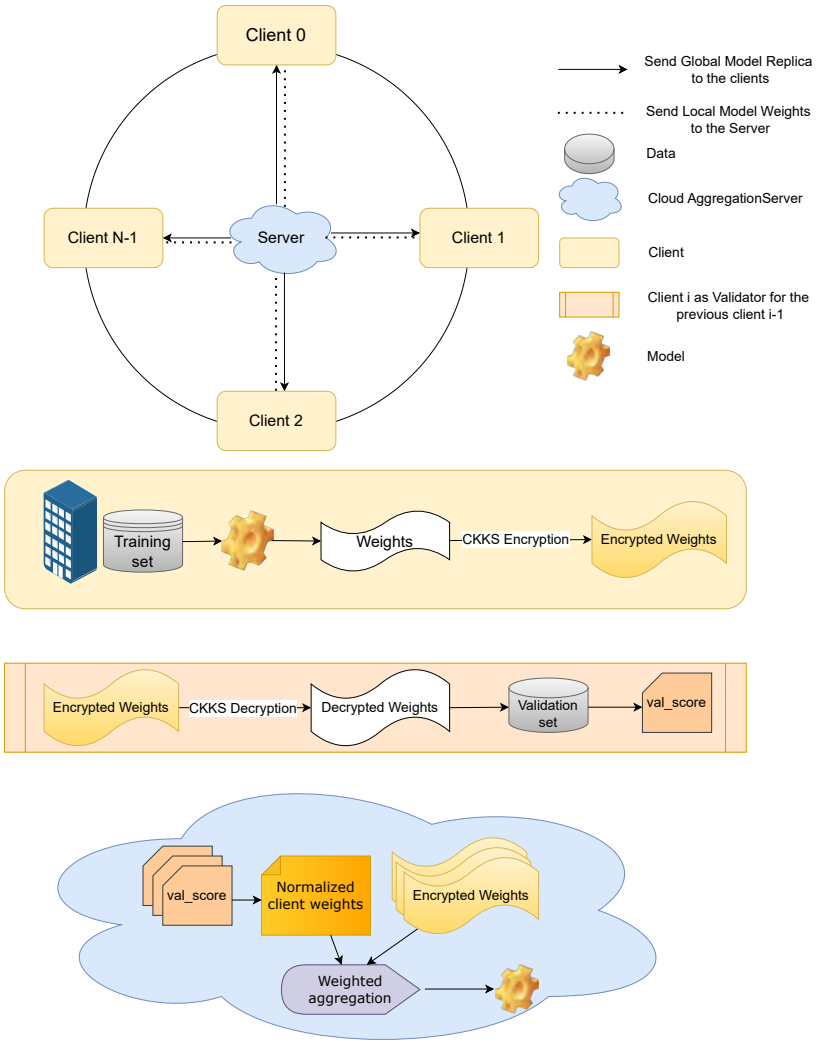

Figure 3. Working of the Proposed FedARCH framework

updated and adjusted using smoothing and decay factors. A smoothing factor α is employed to handle sudden increases or decreases in client performance and maintain stability, while a decay factor β reduces the impact of older reputations, ensuring the model adapts to the latest updates. The working of the smoothing and decay factors is given by Equation 13 and the notations are described in Table 2.

$$R_i^{t+1} = (\alpha \cdot R_i^t + (1-\alpha) \cdot P_i^t) \cdot \beta \tag{13}$$

If the smoothing factor α is high (closer to 1), the new reputation will rely more heavily on the previous reputation, reducing the influence of the current performance. This makes the system less sensitive to sudden changes or fluctuations in client performance. On the other hand, if α is low (closer to 0), the current performance will have a greater influence, making the reputation more responsive to recent client behavior. For the decay factor, if β is close to 1, the reputations will maintain their value over time, retaining a strong memory of both past and current performance. If β is closer to 0, the reputations will gradually decay, allowing newer updates to have a stronger influence while older updates lose significance. The choice of these factors can be dynamically adjusted by the server based on the validation scores

obtained from the clients.

The reputations are then normalized to obtain the normalized repuation weight score $\bar{R}_i^t$ for each client. Using these plaintext normalized reputation weights, the server performs weighted aggregation on the clients' encrypted local model weights, optimizing CKKS HE to perform addition and multiplication operations on encrypted data without increasing the computational complexity. This process is represented in Equation 14.

$$E^{t+1} = \sum_{i=1}^{N} \bar{R}_i^{t+1} \cdot E_i^t \tag{14}$$

After the weighted aggregation, the initial global model W_0 is updated with the new aggregated weights W_t, which are then sent to all clients to update their local models. These aggregated weights remain in encrypted form, so the clients decrypt them using CKKS decryption before updating their local models. This entire process is repeated for R rounds or until convergence.

EXPERIMENTAL RESULTS

Dataset

For implementing FedARCH, we have considered the Kaggle dataset (Nickparvar (2021)) containing 7,023 brain MRI images with four class labels: meningioma, glioma, pituitary, and no tumor. Three datasets—Figshare, SARTAJ, and Br35H—combined to form this dataset. A representative sample image for each class label is shown in Figure 4. The dataset is organized into two main folders: Training and Testing. Each folder contains subfolders corresponding to the four class labels: meningioma, glioma, pituitary, and no tumor. The Training folder contains 5,712 images, while the Testing folder contains

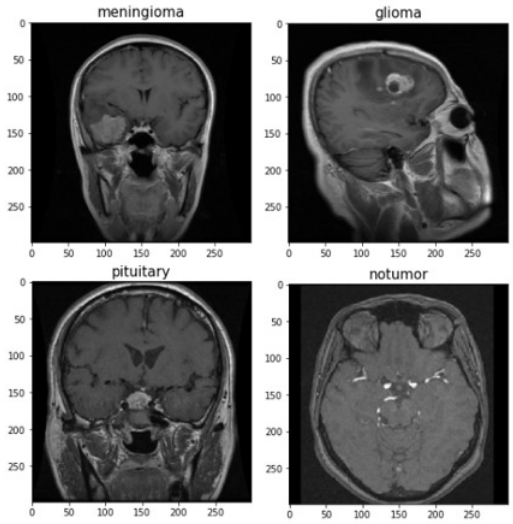

Figure 4. Sample brain MRI images

1,311 images. The class distribution in each folder is illustrated in Figure 5. We further split the images in the Testing folder into validation and testing sets, with 655 images for validation and 656 images for testing. We have created a simulation environment with 10 clients and a central server with a global model. Each client holds a replica of the global model and acts as a validator for the previous client. The training data is split among the 10 clients, and the validation data is distributed to all clients for client evaluation.

Experimental Setup

As discussed earlier, a simulation environment is created to establish a client-server framework, consisting of a single central server and 10 clients. The entire FL process is implemented from scratch using PyTorch,

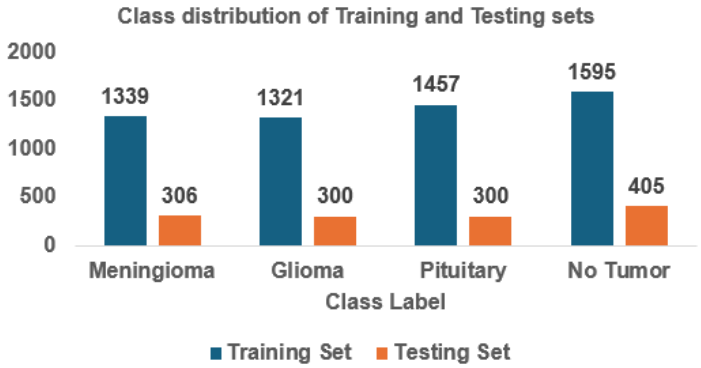

Figure 5. Class distribution of Kaggle dataset

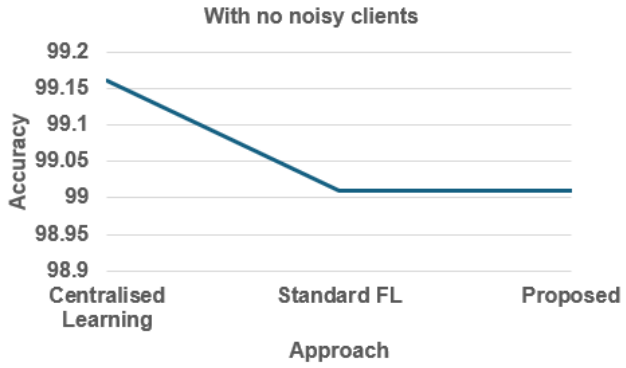

Figure 6. Comparison of CL vs Standard FL vs Proposed

without relying on any existing FL frameworks. For CKKS HE, the TenSEAL package is utilized. The implementation is carried out using Jupyter Notebook on a DGX server with the following specifications: Nvidia RTX 3060 GPUs with 12 GB GDDR6 graphics and Intel Core i9 CPUs with 8 cores and 64 (2 x 32GB) DDR4RAM.

Evaluation metrics

We rigorously evaluated the proposed framework against state-of-the-art solutions using various evaluation metrics (Singamsetty et al. (2024)). Accuracy is used to obtain the overall performance measure. Precision and recall are employed to assess the model's impact in reducing the number of false positives (FP) and false negatives (FN), respectively. The F1-score is calculated to balance both precision and recall. For brain tumor multi-class classification, it is crucial to not only reduce the number of FPs but also reduce FNs. An FP could cause unnecessary panic and lead to unnecessary treatment for patients, while an FN could overlook a potentially dangerous tumor, leading to delayed treatment and decreasing patient survival rates. These metrics ensure that the FedARCH framework is thoroughly evaluated, thereby improving decision-making and patient outcomes.

Results

We compared our proposed framework with existing solutions, and the comparison is presented in Table 3. This table highlights the key features incorporated in the proposed framework that are not addressed by the existing work. The proposed FedARCH framework is compared with centralized learning and standard FL with FedAvg, and the results are shown in Figure 6. FedARCH performs on par with FedAvg and almost similarly to centralized learning. To further evaluate its robustness, gaussian noise is added to some clients' data to observe the impact on the final model accuracy. We initially introduce noise to 10% of the clients and gradually increase this to 50% of the clients. Three different noise levels are considered: low (noise_level=0.1), medium (noise_level=0.4), and high (noise_level=0.8). FedARCH is compared with the standard FL with FedAvg, and the results are illustrated in Figures 7-9. The results

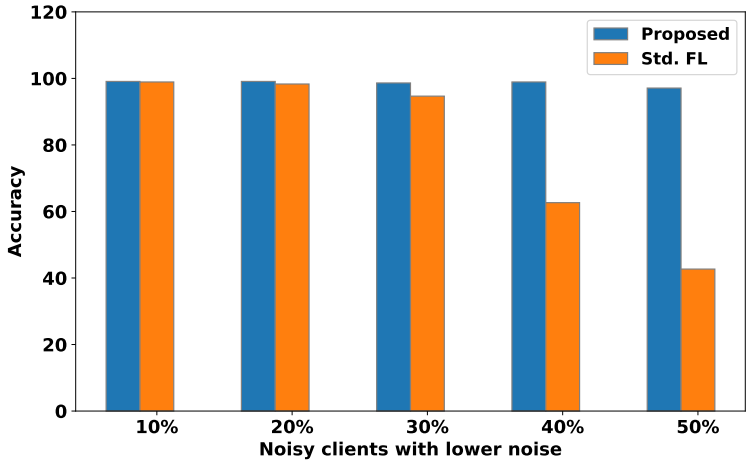

Figure 7. Comparison of Standard FL and Proposed with different percentages of noisy clients with lower noise level

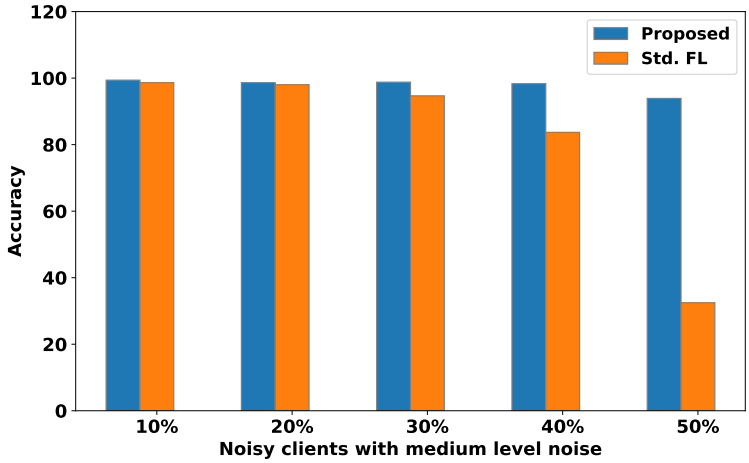

Figure 8. Comparison of Standard FL and Proposed with different percentages of noisy clients with medium noise level

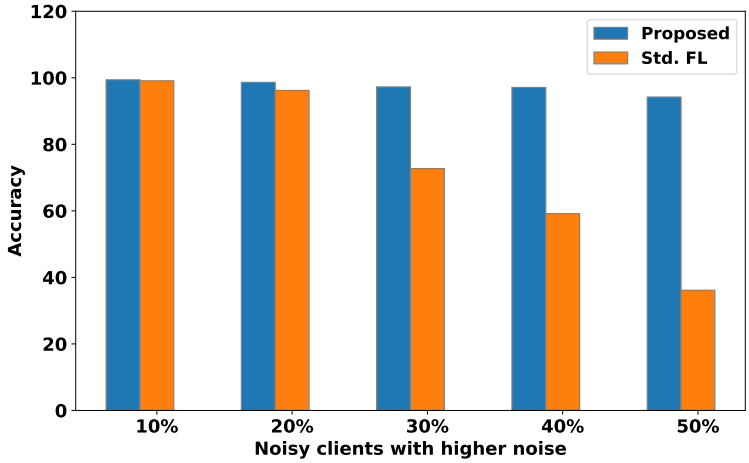

Figure 9. Comparison of Standard FL and Proposed with different percentages of noisy clients with higher noise level

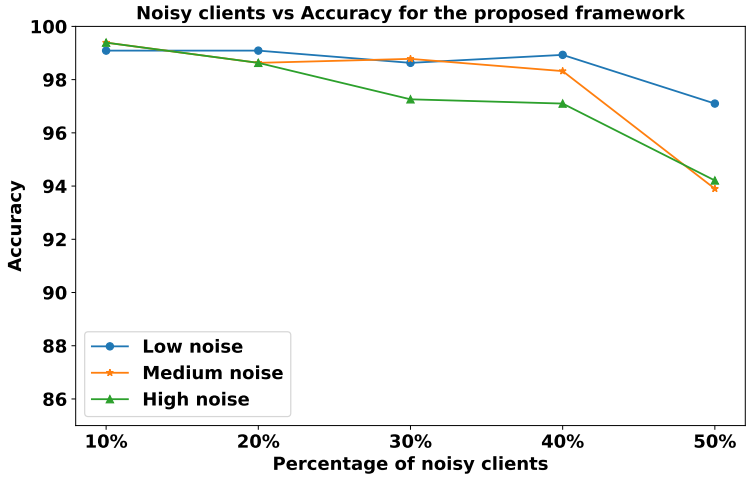

Figure 10. Comparison of Proposed approach with different percentages of noisy clients at different noise levels

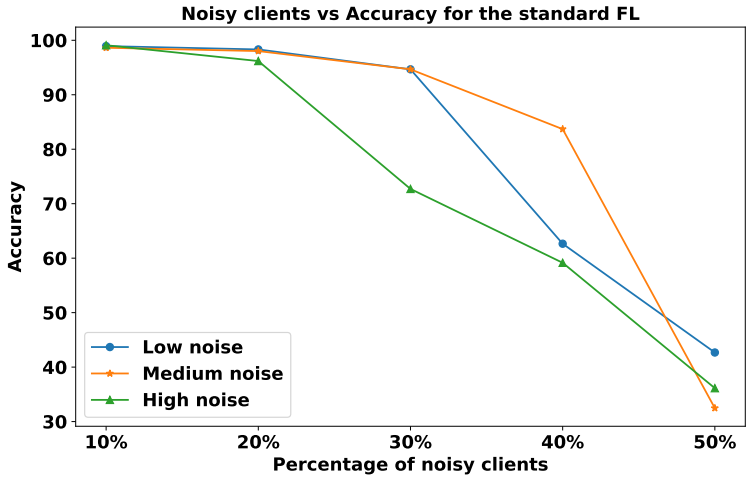

Figure 11. Comparison of Standard FL with different percentages of noisy clients at different noise levels

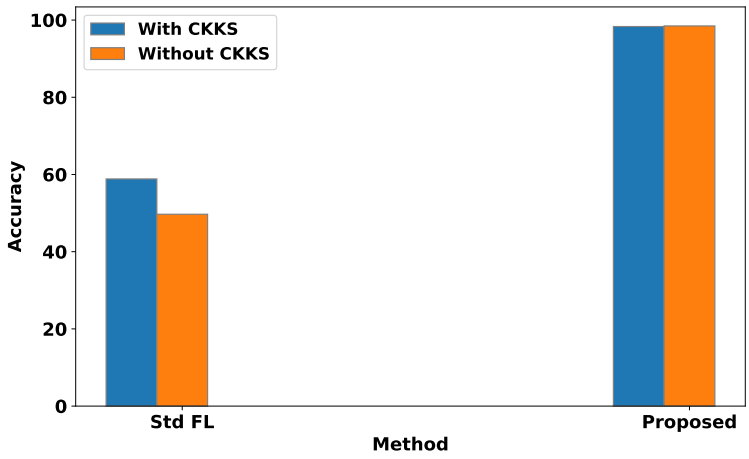

Figure 12. Comparison of with and without ckks for Standard FL and Proposed approaches with 40% noisy clients at a low noise level

Table 3. Comparison of Features Across Different References

Reference	DL	FL	Reputation	Weighted Aggregation	Dynamic Performance Management	HE	Under performing Clients	Medical Data
Thiriveedhi et al. (2025)	✓	×	×	×	×	×	×	✓
Khan et al. (2022b)	✓	×	×	×	×	×	×	✓
Mathivanan et al. (2024)	✓	×	×	×	×	×	×	✓
Albalawi et al. (2024)	✓	✓	×	×	×	×	×	✓
Islam et al. (2023)	✓	✓	×	×	×	×	×	✓
Viet et al. (2023)	✓	✓	×	×	×	×	×	✓
Ay et al. (2024)	✓	✓	×	×	×	×	×	✓
Bhatia and Samet (2023)	✓	✓	×	×	×	×	✓	✓
Lytvyn and Nguyen (2023)	✓	✓	×	×	×	×	×	✓
Fan et al. (2023)	✓	✓	✓	×	✓	✓	✓	×
Zhang et al. (2022)	✓	✓	✓	✓	×	✓	×	✓
Panigrahi et al. (2023)	✓	✓	✓	×	×	✓	×	✓
Kang and Ahn (2023)	✓	✓	×	×	×	×	×	×
Truhn et al. (2024)	✓	✓	×	×	×	✓	×	✓
Kim et al. (2024)	✓	✓	×	×	×	×	×	✓
Yang et al. (2021)	✓	✓	×	✓	×	×	×	✓
FedARCH	✓	✓	✓	✓	✓	✓	✓	✓

clearly demonstrate that as both the percentage of noisy clients and the level of noise in the clients' data increase, FedARCH efficiently resists the impact of noise, whereas the standard FedAvg approach fails.

The impact of increasing the noise level on model accuracy is also considered and is illustrated in Figures 10 and 11. With an increasing noise level and number of noisy clients, there is some impact on the proposed framework, as the accuracy slightly reduces from 99% to 94%. However, for standard FedAvg, there is a significant drop in performance, with accuracy plummeting from 99% to 32%. This highlights the level of resistance exhibited by our proposed FedARCH framework.

We also compare the influence of CKKS HE on both the standard and proposed approaches. A simulation with 40% noisy clients at a low noise level is used to evaluate the impact on both approaches, with and without CKKS. The results are shown in Figure 12. No significant difference is observed in the proposed approach, but the standard approach performs better with the inclusion of CKKS. This highlights that the addition of CKKS HE does not negatively affect the performance of our model, unlike the

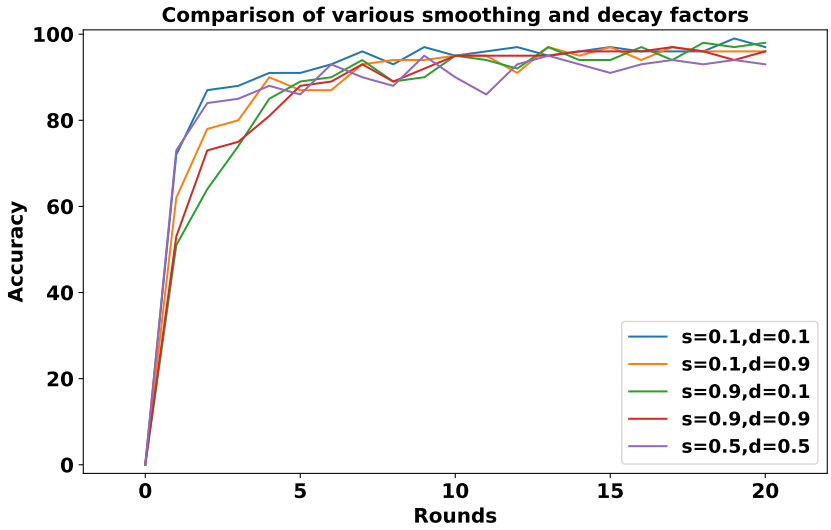

Figure 13. Comparison of various combinations of smoothing and decay factors

```
Client 2 is underperforming
Client 3 is underperforming
Client 4 is underperforming
Client 6 is underperforming
```

(a) Before a spike and drop simulation

```
Client 2 is underperforming
Client 4 is underperforming
Client 5 is underperforming
Client 6 is underperforming
```

(b) After a spike and drop simulation

Figure 14. Validation reports

Differential Privacy approach. This can be attributed to CKKS's ability to operate on encrypted data, real numbers, and approximate arithmetic. The accumulation of noise, which is a common issue in encryption scenarios, is effectively managed in our case. This is because we only consider plaintext-ciphertext multiplication during weighted aggregation, rather than ciphertext-ciphertext multiplication, which helps prevent significant noise accumulation. In this context, the plaintext refers to the normalized reputation weights, and the ciphertext refers to the encrypted local model weights.

To address sudden spikes in performance and reduce the impact of older reputations, smoothing and decay factors are considered. Various combinations of these factors were tested and compared to assess their impact, as shown in Figure 13. To simulate real-time changes in performance, we altered the status of an underperforming client (client 3) to a well-performing client and a well-performing client (client 5) to an underperforming client after round 7. Validation reports before and after this simulation are shown in Figure 14. A rigorous evaluation was conducted using various standard metrics, with the results illustrated in Figures 15-17.

Security Analysis

Formal Analysis

FedARCH is robust not only in terms of performance but also with respect to security. To demonstrate this, we utilized a Python tool called Bandit (Roy (2023)), which is highly effective in scanning Python code

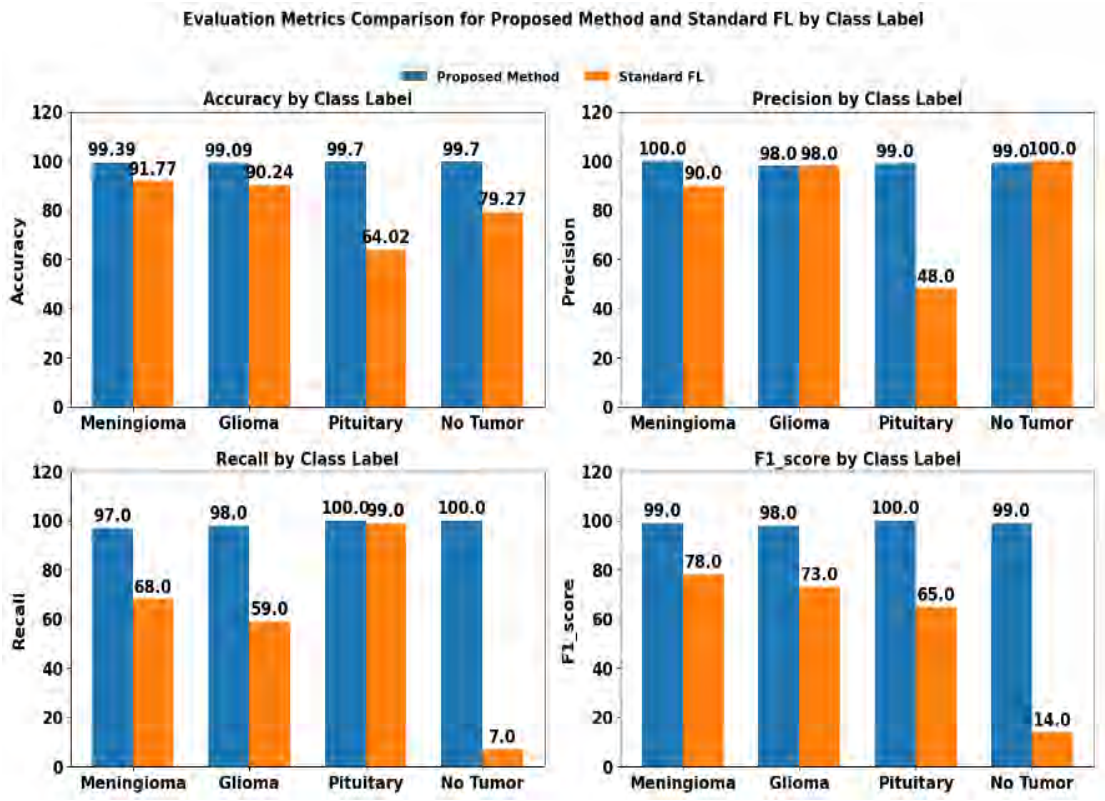

Figure 15. Comparison of evaluation metrics

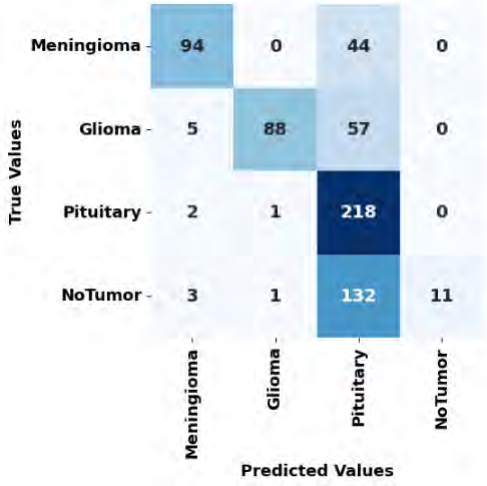

Figure 16. Confusion matrix for Standard FL with 40% noisy clients at a low noise level

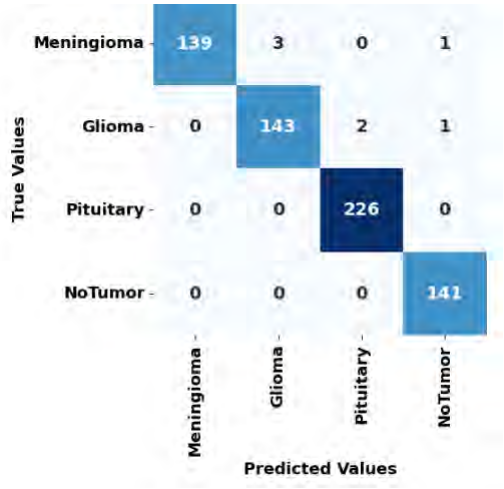

Figure 17. Confusion matrix for Proposed FedARCH approach with 40% noisy clients at a low noise level

```
[main]  INFO    profile include tests: None
[main]  INFO    profile exclude tests: None
[main]  INFO    cli include tests: None
[main]  INFO    cli exclude tests: None
[main]  INFO    running on Python 3.11.7
Run started:2025-02-15 04:19:48.214966

Test results:
        No issues identified.

Code scanned:
        Total lines of code: 174
        Total lines skipped (#nosec): 0

Run metrics:
        Total issues (by severity):
                Undefined: 0
                Low: 0
                Medium: 0
                High: 0
        Total issues (by confidence):
                Undefined: 0
                Low: 0
                Medium: 0
                High: 0
Files skipped (0):
```

Figure 18. Bandit security analysis report of FedARCH

for security vulnerabilities and generating a comprehensive security report. We specifically chose Bandit because it can efficiently detect dangerous code execution commands, code injection vulnerabilities, insecure key usage, and weak cryptographic practices, issues that are particularly relevant in FL scenarios. We have also used the Scyther tool (Egala et al. (2023)), which is popular for formal security analysis of communication protocols. It can detect several attacks like Man-in-the-middle (MITM) attacks, Denial-of-Service (DoS) vulnerabilities, Replay attacks, Authentication weaknesses, and Key exchange security. Given the security-sensitive nature of FL, we aimed to identify and eliminate such vulnerabilities in our proposed framework. The Bandit report and scyther report, presented in Figure 18 and Figure 19 respectively, serves as concrete evidence of FedARCH's resilience against security threats.

Informal Analysis

The CKKS HE scheme, which we considered in our proposed FedARCH framework, facilitates the secure aggregation of encrypted weights at the server without requiring decryption in an untrustworthy environment. CKKS is based on the RLWE problem, which is NP-Hard, thus providing potential post-quantum resistance (Lyubashevsky et al. (2010)). Since the clients are assumed to be honest in our framework, the risk of collusion attacks—where clients collude with the server to infer other clients'

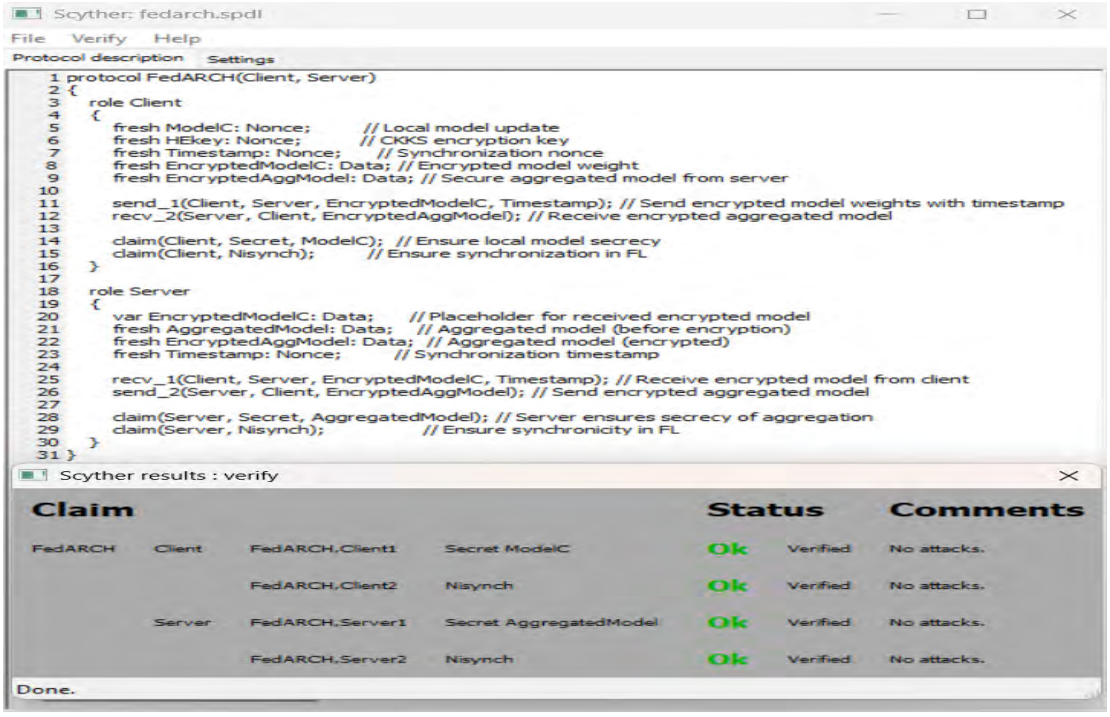

Figure 19. Scyther security analysis report of FedARCH

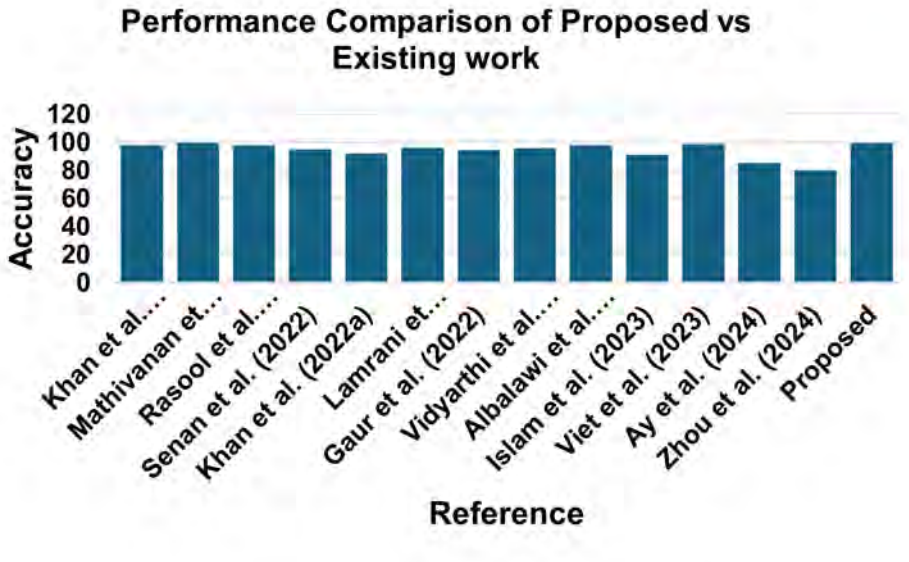

Figure 20. Comparison of the the proposed FedARCH approach and existing related work

Table 4. Evaluation Metrics Comparison for Proposed FedARCH and Standard FL by Class Label for 40% noisy clients with lower noise

Class Label	Metric	FedARCH	Standard FL
Meningioma	Accuracy	99	92
	Precision	100	90
	Recall	97	68
	F1-Score	99	78
Glioma	Accuracy	99	90
	Precision	98	98
	Recall	98	59
	F1-Score	98	73
Pituitary	Accuracy	100	64
	Precision	99	48
	Recall	100	99
	F1-Score	100	65
No Tumor	Accuracy	100	79
	Precision	99	100
	Recall	100	7
	F1-Score	99	14

private data—is minimized. Additionally, since each client acts as a validator only for one neighboring client, it can only access one neighboring client's data, thereby preventing any single client from accessing information about all other clients.

Discussion

The results clearly demonstrate that our proposed framework effectively mitigates the impact of underperforming clients on the final global model, whereas the standard FedAvg approach fails as the number of noisy clients and the level of noise increase. The various evaluation metrics further validate that the proposed model significantly reduces false positives and false negatives, thereby avoiding unnecessary panic and delayed treatments. Table 4 highlights the class-wise evaluation metrics obtained by the proposed approach compared to the standard approach. Figure 20 illustrates the robustness of the proposed framework compared to existing approaches. While Mathivanan et al. (2024) achieves the highest accuracy of 99.75%, it lacks the federated learning setup and security guarantees provided by our framework, which achieves the next highest accuracy of 99.39%.

CONCLUSION AND FUTURE DIRECTIONS

In this paper, we proposed FedARCH, a novel FL framework that integrates reputation-aware weighted aggregation and optimized CKKS HE for brain tumor multi-classification in a cross-silo environment. Compared to state-of-the-art solutions, FedARCH not only demonstrated superior performance but also proved more robust in mitigating the impact of underperforming clients on the global model. In addition, underperforming clients receive feedback on their performance, enabling them to enhance their training and contribute more effectively to the collaborative learning process. This, in turn, increases prediction accuracy, ultimately facilitating better treatment options and preventive measures for patients. By integrating optimized CKKS HE, we reduce the computational overhead, balancing both security and performance. The robustness of FedARCH is proved using security analysis tools like Bandit and Scyther.

The proposed framework also shows potential for extension to other medical image analysis tasks, offering significant benefits for automated diagnosis, early detection, and treatment. Although this study assumes that all clients are honest, future work could investigate the FedARCH's applicability in a zero-trust environment, and incorporating performance-based incentives for clients in a decentralized framework.

REFERENCES

Abdusalomov, A. B., Mukhiddinov, M., and Whangbo, T. K. (2023). Brain tumor detection based on deep learning approaches and magnetic resonance imaging. *Cancers*, 15(16):4172.

Albalawi, E., TR, M., Thakur, A., Kumar, V. V., Gupta, M., Khan, S. B., and Almusharraf, A. (2024). Integrated approach of federated learning with transfer learning for classification and diagnosis of brain tumor. *BMC Medical Imaging*, 24(1):110.

Ay, Ş., Ekinci, E., and Garip, Z. (2024). A brain tumour classification on the magnetic resonance images using convolutional neural network based privacy-preserving federated learning. *International Journal of Imaging Systems and Technology*, 34(1):e23018.

Bhatia, L. and Samet, S. (2023). A decentralized data evaluation framework in federated learning. *Blockchain: Research and Applications*, 4(4):100152.

Bonawitz, K., Ivanov, V., Kreuter, B., Marcedone, A., McMahan, H. B., Patel, S., Ramage, D., Segal, A., and Seth, K. (2017). Practical secure aggregation for privacy-preserving machine learning. In *proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 1175–1191.

Brakerski, Z. (2012). Fully homomorphic encryption without modulus switching from classical gapsvp. In *Annual cryptology conference*, pages 868–886. Springer.

Brakerski, Z., Gentry, C., and Vaikuntanathan, V. (2014). (leveled) fully homomorphic encryption without bootstrapping. *ACM Transactions on Computation Theory (TOCT)*, 6(3):1–36.

Cheon, J. H., Kim, A., Kim, M., and Song, Y. (2017). Homomorphic encryption for arithmetic of approximate numbers. In *Advances in Cryptology–ASIACRYPT 2017: 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part I 23*, pages 409–437. Springer.

Dwork, C., Roth, A., et al. (2014). The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407.

Egala, B. S., Pradhan, A. K., Dey, P., Badarla, V., and Mohanty, S. P. (2023). Fortified-chain 2.0: Intelligent blockchain for decentralized smart healthcare system. *IEEE Internet of Things Journal*, 10(14):12308–12321.

Fan, M., Ji, K., Zhang, Z., Yu, H., and Sun, G. (2023). Lightweight privacy and security computing for blockchained federated learning in iot. *IEEE Internet of Things Journal*, 10(18):16048–16060.

Fredrikson, M., Jha, S., and Ristenpart, T. (2015). Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, pages 1322–1333.

Gaur, L., Bhandari, M., Razdan, T., Mallik, S., and Zhao, Z. (2022). Explanation-driven deep learning model for prediction of brain tumour status using mri image data. *Frontiers in genetics*, 13:822666.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Huynh, D. (2020). Ckks explained.

Islam, M., Reza, M. T., Kaosar, M., and Parvez, M. Z. (2023). Effectiveness of federated learning and cnn ensemble architectures for identifying brain tumors using mri images. *Neural Processing Letters*, 55(4):3779–3809.

Kang, D. and Ahn, C. W. (2023). Ga approach to optimize training client set in federated learning. *IEEE Access*.

Khan, A. H., Abbas, S., Khan, M. A., Farooq, U., Khan, W. A., Siddiqui, S. Y., and Ahmad, A. (2022a). Intelligent model for brain tumor identification using deep learning. *Applied Computational Intelligence and Soft Computing*, 2022(1):8104054.

Khan, M. S. I., Rahman, A., Debnath, T., Karim, M. R., Nasir, M. K., Band, S. S., Mosavi, A., and Dehzangi, I. (2022b). Accurate brain tumor detection using deep convolutional neural network. *Computational and Structural Biotechnology Journal*, 20:4733–4745.

Kim, S., Park, H., Kang, M., Jin, K. H., Adeli, E., Pohl, K. M., and Park, S. H. (2024). Federated learning with knowledge distillation for multi-organ segmentation with partially labeled datasets. *Medical Image Analysis*, 95:103156.

Lamrani, D., Cherradi, B., El Gannour, O., Bouqentar, M. A., and Bahatti, L. (2022). Brain tumor detection using mri images and convolutional neural network. *International Journal of Advanced Computer Science and Applications*, 13(7).

Lytvyn, O. and Nguyen, G. (2023). Secure multi-party computation for magnetic resonance imaging classification. *Procedia Computer Science*, 220:24–31.

Lyubashevsky, V., Peikert, C., and Regev, O. (2010). On ideal lattices and learning with errors over rings. In *Advances in Cryptology–EUROCRYPT 2010: 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, French Riviera, May 30–June 3, 2010. Proceedings 29*, pages 1–23. Springer.

Mathivanan, S. K., Sonaimuthu, S., Murugesan, S., Rajadurai, H., Shivahare, B. D., and Shah, M. A. (2024). Employing deep learning and transfer learning for accurate brain tumor detection. *Scientific Reports*, 14(1):7232.

McMahan, B., Moore, E., Ramage, D., Hampson, S., and y Arcas, B. A. (2017). Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR.

Nickparvar, M. (2021). Brain tumor mri dataset.

Panigrahi, M., Bharti, S., and Sharma, A. (2023). A reputation-aware hierarchical aggregation framework for federated learning. *Computers and Electrical Engineering*, 111:108900.

Rasool, M., Ismail, N. A., Boulila, W., Ammar, A., Samma, H., Yafooz, W. M., and Emara, A.-H. M. (2022). A hybrid deep learning model for brain tumour classification. *Entropy*, 24(6):799.

Remzan, N., Hachimi, Y. E., Tahiry, K., and Farchi, A. (2024). Ensemble learning based-features extraction for brain mr images classification with machine learning classifiers. *Multimedia Tools and Applications*, 83(19):57661–57684.

Roy, S. (2023). Bandit tool.

Senan, E. M., Jadhav, M. E., Rassem, T. H., Aljaloud, A. S., Mohammed, B. A., and Al-Mekhlafi, Z. G. (2022). Early diagnosis of brain tumour mri images using hybrid techniques between deep and machine learning. *Computational and Mathematical Methods in Medicine*, 2022(1):8330833.

Sheller, M. J., Edwards, B., Reina, G. A., Martin, J., Pati, S., Kotrotsou, A., Milchenko, M., Xu, W., Marcus, D., Colen, R. R., et al. (2020). Federated learning in medicine: facilitating multi-institutional collaborations without sharing patient data. *Scientific reports*, 10(1):12598.

Singamsetty, S., Ghanta, S., Biswas, S., and Pradhan, A. (2024). Enhancing machine learning-based forecasting of chronic renal disease with explainable ai. *PeerJ Computer Science*, 10:e2291.

Thiriveedhi, A., Ghanta, S., Biswas, S., and Pradhan, A. K. (2025). All-net: integrating cnn and explainable-ai for enhanced diagnosis and interpretation of acute lymphoblastic leukemia. *PeerJ Computer Science*, 11:e2600.

Truhn, D., Arasteh, S. T., Saldanha, O. L., Müller-Franzes, G., Khader, F., Quirke, P., West, N. P., Gray, R., Hutchins, G. G., James, J. A., et al. (2024). Encrypted federated learning for secure decentralized collaboration in cancer image analysis. *Medical image analysis*, 92:103059.

Vidyarthi, A., Agarwal, R., Gupta, D., Sharma, R., Draheim, D., and Tiwari, P. (2022). Machine learning assisted methodology for multiclass classification of malignant brain tumors. *IEEE Access*, 10:50624–50640.

Viet, K. L. D., Le Ha, K., Quoc, T. N., and Hoang, V. T. (2023). Mri brain tumor classification based on federated deep learning. In *2023 Zooming Innovation in Consumer Technologies Conference (ZINC)*, pages 131–135. IEEE.

Yang, D., Xu, Z., Li, W., Myronenko, A., Roth, H. R., Harmon, S., Xu, S., Turkbey, B., Turkbey, E., Wang, X., et al. (2021). Federated semi-supervised learning for covid region segmentation in chest ct using multi-national data from china, italy, japan. *Medical image analysis*, 70:101992.

Zhang, L., Xu, J., Vijayakumar, P., Sharma, P. K., and Ghosh, U. (2022). Homomorphic encryption-based privacy-preserving federated learning in iot-enabled healthcare system. *IEEE Transactions on Network Science and Engineering*, 10(5):2864–2880.

Zhao, C., Zhao, S., Zhao, M., Chen, Z., Gao, C.-Z., Li, H., and Tan, Y.-a. (2019). Secure multi-party computation: theory, practice and applications. *Information Sciences*, 476:357–372.

Zhou, L., Wang, M., and Zhou, N. (2024). Distributed federated learning-based deep learning model for privacy mri brain tumor detection. *arXiv preprint arXiv:2404.10026*.