Proceedings of the Indian National Science Academy (2026) 92:147–170
https://doi.org/10.1007/s43538-025-00394-2

RESEARCH PAPER

Advanced federated learning security: NTRU and blockchain synergy

Ashlesha Hota[1] · Arishmita Biswas[1,2] · Sanchita Saha[1,3] · Amitava Nag[1] · Ferdous Ahmed Barbhuiya[4] · Sukumar Nandi[5]

Received: 3 August 2024 / Accepted: 16 January 2025 / Published online: 19 March 2025

Abstract
Federated learning (FL) systems rely on a central server for model aggregation, introducing challenges such as synchronization issues, dishonest servers, curious clients, and participant heterogeneity. To address these challenges, we propose a secure FL framework that integrates Blockchain technology and lattice-based cryptography. Our system employs NTRU (*N*th Degree Truncated Polynomial Ring Units)-based encryption to protect model gradients during transmission, providing robust defense against passive and quantum attacks. Additionally, we leverage Blockchain technology to enable decentralized and asynchronous aggregation, ensuring data integrity and reducing the risks associated with centralized control. We use selective parameter encryption to minimize computational overhead by focusing on the most privacy-sensitive gradients, enhancing both efficiency and security. Through comprehensive simulations and theoretical analysis, we demonstrate that our framework offers strong privacy guarantees and resilience against various adversarial threats. Compared to traditional cryptographic methods such as RSA and Elliptic Curve Diffie–Hellman, our solution provides a practical and scalable approach that maintains model accuracy while safeguarding data, proving it as a robust solution for real-world FL deployments.

Keywords AI · Decentralized security · Asynchronous learning · Privacy · FL · Blockchain · Post quantum security · Lattice based cryptography · NTRU

✉ Amitava Nag
amitava.nag@cit.ac.in

Ashlesha Hota
ashleshahota@gmail.com

Arishmita Biswas
u19cse1037@cit.ac.in

Sanchita Saha
sanchitacse2007@gmail.com

Ferdous Ahmed Barbhuiya
ferdous@iiitg.ac.in

Sukumar Nandi
sukumar@iitg.ac.in

[1] Department of Computer Science and Engineering, Central Institute of Technology Kokrajhar, Kokrajhar, Assam, India

[2] Department of Computer Science and Engineering, Indian Institute of Technology Palakkad, Palakkad, Kerela, India

[3] Department of Computer Science and Engineering, Techno International New Town, Kolkata, West Bengal, India

[4] Department of Computer Science and Engineering, Indian Institute of Information Technology Guwahati, Guwahati, Assam, India

[5] Department of Computer Science and Engineering, Indian Institute of Technology Guwahati, Guwahati, Assam, India

Introduction

Artificial Intelligence (AI) is a key driver of technological progress, with the potential to reshape various industries and how we interact with technology. It automates tasks that were traditionally done by humans. With the explosion of data, AI algorithms can analyze large volumes of information to find valuable insights. Machine learning and deep learning are important parts of AI, focusing on training artificial neural networks to make decisions like humans. Deep learning, in particular, is a hot area of research within AI (Han et al. 2022). It is making big strides in medical fields like radiology, pathology, and dermatology, where it helps to diagnose and predict diseases using patient data (Puttagunta and Ravi 2021). Deep learning is also being used in traffic systems to make transportation networks more efficient and safer (Jiang et al. 2015).

To effectively train and optimize a deep neural network, a significant volume of data is essential, often sourced from users. However, this data, obtained from various sources, frequently contains sensitive information like personally identifiable information (PII), including names, addresses,

phone numbers, and more. The risk of unauthorized access to such data poses threats of privacy breaches and misuse, inevitably impacting user privacy. Consequently, users may become hesitant to share their data, hindering the collection of true data in deep learning research (Table 1) (Han et al. 2022).

Federated learning (FL) offers a solution by allowing AI models to be trained on local data without compromising data privacy. It involves clients and central servers collaboratively share local gradients and global parameters, enabling model training across decentralized devices or servers without directly exposing raw user data (Han et al. 2022). Originally proposed by Google in 2016 to enhance data privacy and security (McMahan et al. 2017), FL has since evolved into a significant research area in AI (Bhagoji et al. 2019; Ng et al. 2021). Some of the major advantages of the FL technique are: it is a decentralized approach, which overcomes the data corruption issues faced by centralized systems, overcomes the problems of tiny datasets and eliminates the tedious task of gathering a single large dataset at a single server.

In the era of Information Technology, data faces numerous risks such as hardware failures, software vulnerabilities, and human errors, all of which can lead to data compromise. Therefore, ensuring data integrity is crucial, especially in the context of FL training processes (Han et al. 2022).

FL has wide and vivid applications in many sectors. In Healthcare, FL allows medical institutions to collaboratively develop models for disease prediction and personalized treatments without sharing patient data, ensuring compliance with privacy regulations. In Banking, FL helps financial institutions build secure fraud detection systems and credit scoring models while keeping transaction data decentralized, thus preventing breaches of financial privacy. In Transportation, FL facilitates traffic flow forecasting and intelligent traffic management by enabling multiple entities to collaborate on models without sharing sensitive vehicle or route data. For Telecommunications, FL supports personalized services and network optimization while maintaining user privacy and regulatory compliance. Finally, in Education, FL enables institutions to develop personalized learning models based on student data while keeping that information secure and localized. FL and Cyber-Physical Systems (CPS) are closely connected, particularly in applications where privacy, real-time data processing, and decentralized control are crucial. In the context of CPS, data generated from sensors, actuators, and other devices is often distributed across multiple locations, such as in smart grids, autonomous vehicles, or industrial automation. CPS typically consists of sensors, actuators, computational algorithms, and communication networks that work together to monitor and control physical processes. These systems are used in a wide range of applications, such as smart grids, autonomous vehicles, industrial automation, and healthcare systems. The physical component includes real-world objects or environment, while the cyber component involves the digital systems that process data, make decisions, and send commands to the physical world. The integration of the cyber and physical elements allows CPS to operate autonomously or semi-autonomously, often requiring real-time data processing, high reliability, and robust communication networks.

Table 1 List of abbreviations

Acronym	Full form
PKC	Public key cryptography
SKC	Symmetric key cryptographic
AKC	Asymmetric key cryptographic
RSA	Rivest Shamir Adleman
DH	Diffie Hellman
ECC	Elliptic curve cryptography
LBC	Lattice based cryptography
LWE	Learning with error
NTRU	Nth Truncated ring polynomial unit
MuS	Multi-use secret sharing
TernGrad	Ternary gradient
CPS	Cyber physical system

FL, while promising, comes with its challenges. It's resource-intensive and vulnerable to attacks from dishonest servers and curious clients, posing threats to both data privacy and result integrity. These challenges underscore the importance of exploring security measures in federated training with deep neural networks (Han et al. 2022). FL in cyber-physical systems (CPS) can face several critical challenges, including the surge in data traffic due to the integration of physical and cyber components, which demands efficient communication and security mechanisms. As FL relies on decentralized nodes for model training, it must address the secrecy and reliability trade-off, where maintaining data privacy amidst potential eavesdropping risks must not compromise the system's ability to reliably transmit model updates. The open nature of wireless communication networks, characteristic of CPS, further heightens the vulnerability to information interception, making secure data aggregation and communication essential (Ju et al. 2024a). Additionally, the dynamic nature of CPS environments, with varying interference and eavesdropper densities, demands adaptive strategies in FL to ensure both security and model convergence. Addressing these challenges require integrating advanced encryption techniques, adaptive transmission strategies, and privacy-preserving mechanisms to ensure secure and efficient FL in CPS environments (Ju et al. 2024a, b).

Existing defense methods in FL include differential privacy (DP) and secure aggregation. Differential privacy introduces noise into the local models to obscure sensitive information. While this technique effectively protects individual data points, it often comes at the cost of model accuracy. The noise added for privacy can degrade performance, particularly in scenarios requiring high precision. In contrast, secure aggregation (Bonawitz et al. 2017) ensures that model updates are only revealed in

aggregate form, protecting individual client contributions. However, this method introduces additional synchronization steps, complicating the training process. It is also highly sensitive to client dropout, a significant limitation in real-world FL applications where clients may experience unstable internet connections, software crashes, or other disruptions. These challenges reduce the practicality of secure aggregation in dynamic and heterogeneous environments.

Homomorphic Encryption-based FL (HE-FL) offers another promising approach by encrypting local models on client devices and performing aggregation directly on ciphertexts at the server. This method preserves privacy without compromising model accuracy, as the encrypted data maintains the same structure as the original. Consequently, HE-FL enables secure FL deployments with performance comparable to traditional, unencrypted FL systems. Several FL frameworks have already adopted HE due to these advantages (Bonawitz et al. 2017; Ou et al. 2020; Zhang et al. 2019, 2020).

However, despite its potential, homomorphic encryption introduces significant computational and communication overheads, making it impractical for many real-world applications. Prior HE-FL solutions often rely on generic HE methods that are not optimized for large-scale FL scenarios. As a result, scalability becomes a major bottleneck, particularly when training large foundation models across resource-constrained devices. This overhead is especially pronounced during encrypted computation and communication, where delays can be up to 15 times longer than actual model training (Gouert et al. 2023). These challenges restrict HE's feasibility in scenarios requiring efficient, large-scale FL deployments.

Masking techniques (Xiong and Zhu 2024) have been explored to shield local model updates. This approach ensures that the details of each update remain private while reducing some of the computational burdens associated with HE. However, further optimizations are essential to make HE-based FL practical and scalable, particularly for environments involving large models and limited computational resources.

To address these concerns, we conducted a thorough review of relevant literature, summarized in section "Related research". Through a concise literature review, we came to the conclusion that, in order to maintain the privacy of user data in FL, various public key encryption techniques are used. However, the traditional public key cryptography techniques fail to achieve post quantum security. The advent of quantum computers has highly endangered the traditional cryptography schemes like RSA and ECC. Our key contributions are as follows:

1. *Novel NTRU-based encryption for FL:* We propose a secure FL framework integrating lattice-based NTRU cryptography, ensuring robust defense against passive and quantum attacks during gradient transmission.
2. *Blockchain integration for decentralized security:* The system incorporates blockchain technology to facilitate asynchronous aggregation and enhance trust, ensuring data integrity and reducing dependency on central servers.
3. *Relay-agent framework for secure communication:* We introduce relay agents to manage encrypted gradient exchange, reducing computational overhead on clients and enhancing security against adversarial threats.
4. *Lightweight post-quantum security:* The proposed method outperforms traditional cryptographic schemes like RSA and ECC, offering computational efficiency and post-quantum resistance without compromising model accuracy.
5. *Multi server FL model:* By distributing gradient shares among multiple servers, the system prevents any single server from accessing complete gradient information, mitigating risks from honest-but-curious adversaries.
6. *Generation of synthetic data:* We generated synthetic data using GANs to perform a large-scale simulation and create a larger set of test samples.
7. *Comprehensive performance evaluation for COVID 19 detection:* Extensive simulations demonstrate the framework's scalability and efficiency. Results show significant overhead reduction and resilience against various privacy threats, highlighting the practical viability of the proposed approach for the COVID 19 detection.

Our main contributions in the paper are highlighted in the Table 2.

The proposed approach tackles limitations of traditional public key encryption methods by ensuring post-quantum security while also enhancing computational efficiency, surpassing the Elliptic Curve Diffie Hellman algorithm in speed.

The paper is carefully structured as follows: section "Introduction" outlines the problems, motivations, and objectives of the research. Section "Related research" provides a concise review of existing solutions. Section "Preliminaries" delves into the cryptographic tools employed in the proposed scheme. Section "Methodology" elaborates on our method, and section "Simulation and results" includes implementation details and simulation outcomes. Section "Discussion" addresses overheads, efficiency, and security aspects, alongside comparisons with recent literature. Section "Conclusion" concludes our efforts.

Related research

This section is devoted to listing prominent security methodologies that have been adopted to attain robust security in FL.

Table 2 Our contributions

Sl. no	Limitation identified	Contribution to overcome the limitation
1	Gradient communication in FL in not completely secured	We encrypt the gradients so that they are not misused or decoded by intruders
2	Encryption schemes like ECC and RSA fail to establish post quantum resistance	We employ encryption method based on a family of classical crypto-systems that are post quantum effective: Lattice based Cryptography
3	Post Quantum schemes like Homomorphic Encryption exhibit huge trade-off in accuracy	To overcome the trade-off, we apply NTRU Encryption to FL setting
4	The crypto-systems are usually hard to operate with demand huge computational resources	Our proposed schemes is computationally faster and lightweight

Hoffstein et al. (2006) proposed and introduced the NTRU crypto-system, which is based on ring theory and designed for public key encryption. The authors focused on the algorithmic aspects and number theory foundations of the NTRU crypto-system within the context of algorithmic number theory.

Secure Multi-Party Computation (SMPC) involves cryptographic methods for securely computing a public function over private inputs contributed by multiple parties, ensuring that no individual input is revealed. In the context of FL, Bonawitz et al. (2017) developed a secure aggregation protocol resilient to client dropouts, employing techniques such as random value blinding, Shamir's Secret Sharing, and symmetric encryption to protect client data during training. Building on this, Kadhe et al. (2020) introduced a more efficient aggregation protocol that utilizes the fast Fourier transform and multi-secret sharing, enhancing computational performance and empowers the server to aggregate clients' models while ensuring data privacy and offers notable advantages in terms of computation, communication, and resilience to client dropouts. Choi et al. (2020) proposed a streamlined solution that offers data privacy while significantly reducing computational resource requirements compared to the current secure solution. In contrast to the existing solution, the proposed scheme designs the secret-sharing nodes' topology as a sparse random graph rather than a complete graph. ossain Fereidooni et al. (2021) presented a robust FL solution, SAFELearn, that incorporates secure aggregation to mitigate inference attacks when analyzing model updates from individual clients.

Xu et al. (2022) developed a novel multi-use secret sharing scheme (MuS) employing lattice-based cryptography, empowering participants to locally update their secret shares while ensuring the privacy of their gradients against quantum attacks. Utilizing MuS, the authors also devised LaF, a post-quantum FL protocol that enhances communication efficiency. Ou et al. (2020) designed a vertical FL system that integrates homomorphic encryption to enable Bayesian machine learning, which effectively resolves the data island problem and protects users' privacy.

Dong et al. (2020) analyzed the privacy risks associated with TernGrad and introduced the solution EaSTFLy to effectively mitigate the privacy issues. The authors devised privacy-preserving protocols by combining TernGrad with secret sharing and homomorphic encryption to defend against semi-honest opponents. Zhang et al. (2019) proposed an approach to safeguarding the privacy of the local gradients of the participants using a Paillier homomorphic crypto-system. However, there was a significant trade-off in accuracy. Liu et al. (2022) analyzed the PPAgg protocols and their efficacy in addressing privacy and security concerns in the context of FL systems. The authors also highlighted the OTP (one-time pad) crypto-system that provides security in distributed systems. The system has the obvious disadvantage of generating the OTP at each communication round.

Ulhaq and Burmeister (2020) explored the potential of the differential privacy by design (dPbD) setting to strengthen data privacy in FL systems, emphasizing its scalability and robustness. Zhu et al. (2020) introduced and formalized a novel concept known as weighted FL problem within the setting of secure multiparty computation, which demonstrate security against honest-but-curious servers. The authors also proposed a streamlined implementation of a Beaver triple generator utilizing the El Gamal encryption scheme, making it an ideal complement to the successful implementation of MASCOT. Feng et al. (2021) presented an asynchronous blockchain-based FL setting that leverages the benefits of blockchain, such as non-repudiation, transparency, and decentralization, to resolve the issues of privacy, security, and trust in FL.

The authors in Pan et al. (2024) propose an adaptive segmented encryption method for the CKKS scheme, overcoming its encryption length constraints to support large neural networks. They analyze the relationship between security parameters (like polynomial modulus degree) and performance, offering a methodology to optimize multiplication depth. Additionally, they introduce FedSHE, a FL framework using adaptive segmented CKKS encryption, integrated with FedAvg. Evaluations demonstrate that FedSHE outperforms existing homomorphic encryption-based

FL methods in accuracy, efficiency, communication cost, and security.

The Secured Medical Homomorphic Encryption Algorithm (SMHEA) in Mohandas et al. (2024) introduces a novel masking strategy combining homomorphic encryption with secure computing for FL, using a weighted mean approach based on data quality rather than quantity. It ensures dropout tolerance and individual collision resistance through Diffie–Hellman key exchange and Shamir secret sharing. Additionally, the proposed FL model, tested on real skin cancer datasets, demonstrates confidentiality and effectiveness in handling sensitive medical information.

After a thorough literature survey, we observed that the majority of these papers could not completely solve the issues of honest but curious servers. However, some researchers have addressed the challenges of privacy and security with compromised accuracy in their systems. The proposed solutions are not computationally efficient when it comes to deployment in real-time applications where the desired response time is minimal. Our approach is equally efficient in preserving privacy without charging much computational resources and is computationally much faster. The trained weights are stripped into secrets so that individual secrets cannot reveal any information about the original gradient through reverse engineering practices.

Preliminaries

Lattice based cryptography

The traditional Public Key Cryptography (PKC) include the RSA algorithm, the DH algorithm, and the ECC algorithm. A public-key cryptographic system involves two asymmetric keys, i.e., one is the *public key* known to all while the other is the *private key* that is, as the name suggests, private to the user intended to decrypt the message. It is considered more reliable and secure as compared to symmetric key encryption, but it is computationally more expensive in terms of time. It is also not effective against eavesdropping attacks (Chaudhary et al. 2018).

Threat of Quantum Computing: Quantum computers are highly powerful computers equipped with multiple processors that could easily perform huge mathematical formulations to evade or break into the current security system within seconds.

PKC-based encryption techniques fail miserably to ensure post-quantum security. As a result, in an effort to address these issues, a novel approach to post-quantum cryptography called lattice-based public-key crypto-system (LB-PKC) was introduced by Feng et al. (2021). The main motive for the transition from traditional public key encryption to a lattice-based crypto-system is to achieve post-quantum security.

Let us briefly discuss about what is a lattice? A lattice is a discrete set of points arranged in a regular, grid-like pattern. Specifically, a lattice is a set of points in n-dimensional space that can be generated by integer linear combinations of a set of basis vectors. Assuming that $b_1, b_2, b_3 \ldots .b_n \in R^n$ are n-linearly independent vectors then, the lattice generated by them is the set of vectors given by;

$$\mathscr{L}(b_1, \ldots, b_n) = \left(\sum_{i=1}^{n} b_i x_i : x_i \in \mathbb{Z} \right)$$

The vectors $b_1, b_2, b_3 \ldots .b_n$ are known as the basis of lattice (Micciancio and Regev 2009).

The lattice based cryptographic construction is based on the presumed hardness of the lattice problems, most basic of which are the Shortest Vector Problem (SVP) and the Closest Vector Problem (CVP) (Micciancio and Regev 2009).

Shortest vector problem

The shortest vector problem (SVP) (Ajtai 1996) is a computationally NP—Hard problem to compute the shortest non-zero vector in a high dimensional lattice. Given, a Lattice, L in a n-dimensional space, the task to find the smallest non-zero vector v in L that connects two discrete points in a high dimensional lattice (Ajtai 1996). Owing to its difficulty to solve, this problem is extremely popular in the field of crypto-systems for constructing quantum resistant solutions.

Closest vector problem

The closest vector problem (CVP) is a computational NP-hard problem in mathematics and computer science, particularly in the field of lattice-based cryptography. The problem involves finding the lattice point closest to a given target vector in a lattice. Given a basis for a lattice L in n-dimensional Euclidean space and a target vector t in the same space, the CVP computes the lattice point v in L that is closest to t in terms of Euclidean distance, i.e., the distance between t and v is minimized. However, there are approximation algorithms and heuristic methods that can solve the problem with reasonable efficiency for practical instances of the problem (Chaudhary et al. 2018).

NTRU

Nth Truncated Polynomial Ring Unit (NTRU) is a lattice-based cryptographic technique that is based on the hardness of finding short lattice vectors in polynomial rings. It is a well-known hard problem in the field of lattice-based cryptography. The NTRU crypto-system has several advantages over other lattice-based crypto-systems. It is faster and more efficient than most other lattice-based crypto-systems, and it also provides high-levels of

Table 3 Notations and their meaning

Symbol	Meaning
$\mathscr{C}$	Client
$\mathscr{A}$	Relay agent
$\mathscr{S}$	Server
N	Prime number
f, f_p, f_q	Private keys
Y	Public key
E	Cipher text

security against both classical and quantum attacks. The notations and their meanings are shown in Table 3.

NRTU: KE key exchange protocol

The proposed scheme employs NTRU-KE protocol proposed by Lei and Liao (2013). We redefine the NTRU-KE construction proposed by Lei and Liao (2013). The communication begins from the client's side (Lei and Liao 2013). The notations and their meaning are shown in Table 3.

- A client $\mathscr{C}_i$ chooses two random polynomials f_i and g_i where the polynomial f_i must additionally satisfy the inverse property in the truncated polynomial ring R_q, where $R_q = \mathbb{Z}_q[x]/(X^N - 1)$. The $\mathscr{C}_i$ computes $h_i \equiv f_i^{-1} * g_i (mod\, q)$ and communicates the h_i to its $\mathscr{A}_i$.
- Upon receipt of h_i, $\mathscr{A}_i$ chooses f_j such that it satisfies the inverse property in R_q, and g_j and r_j. $\mathscr{A}_i$ then computes $h_j \equiv f_j^{-1} * g_j (mod\, q)$ and $e_j \equiv pr_j * h_i + f_j (mod\, q)$ and communicates h_j and e_j to $\mathscr{C}_i$.
- Upon receipt of h_j and e_j, $\mathscr{C}_i$ chooses r_i and computes $e_i \equiv pr_i * h_j + f_i (mod\, q)$ and communicates e_i to the $\mathscr{A}_i$ and further computes $a_i = f_i * e_j (mod\, q)$, $K_i = a_i (mod\, p) = f_i * f_j (mod\, p)$.
- $\mathscr{A}_i$ receives e_i and computes $a_j = f_j * e_i (mod\, q)$, $K_j = a_j (mod\, p) = f_i * f_j (mod\, p)$
- The common key established is $K_i = K_j = f_i * f_j (mod\, p)$.

The security of this algorithm is mapped to solving the closest vector problem. NTRU—KE is quantum resistant, is better resistance to distributed attack and computationally faster than Elliptic Curve Diffie Hellman algorithm (Lei and Liao 2013).

NTRU encryption

The NTRU Encryption algorithm proposed by Lei and Liao (2013) is redefined.

For encrypting, a message m, $\mathscr{C}_i$ first chooses a random polynomial α then computes the cipher text as $E \equiv p\alpha * K_i + (mod\, q)$, using the common key K_i and shares the encrypted text with $\mathscr{A}_i$.

NTRU decryption

The cipher text received by $\mathscr{A}_i$ is decrypted using the NTRU Decryption algorithm as proposed by Lei and Liao (2013). For decryption $\mathscr{A}_i$ uses common key and private keys f_i and f_{p_i} where f_{p_i} is the inverse of f_i with respect to p.

- The agent $\mathscr{A}_i$ begins by calculating the polynomial $\theta = f_i * E (mod\, q) = p\alpha * g_i + f_i * m (mod\, q)$. As absolute values of coefficients is small and q is considered to be large so, we almost derive $\theta = p\alpha * g_i + f_i * m$.
- Finally the message m is recovered as $m = f_p * \theta (mod\, p)$.

Blockchain

Blockchain is a distributed ledger technology that serves to maintain transaction records, ensuring the security and transparency of data (Antal et al. 2021a). It operates as an ever-expanding list of records, or blocks, linked and secured using cryptographic hash algorithms. As a result, hash-based data structures form the backbone of Blockchain technology, offering advantages such as collision-free properties, data binding, and data hiding (Antal et al. 2021b).

Each block comprises a set of transactions verified by a network of nodes in a peer-to-peer network. If a proposed alteration passes validation, the block is added to the chain and the change is approved; otherwise, the block is discarded. Prior to universal acceptance into the ledger, each node independently monitors and verifies transactions. Once a block is appended to the chain, it becomes immutable; modifications or deletions require consensus from the network.

Transparency, immutability, and security are among the key benefits of Blockchain technology. These are achieved through consensus algorithms and cryptographic techniques that ensure the integrity and validity of stored data. Blockchain also provides control over servers, similar to cloud infrastructure in data collection, and records data consumption across the network for tracking purposes (Dai et al. 2020).

Methodology

This section delves into the detailed design of our proposed scheme, which comprises a three-layered architecture: a private trusted zone, an un-trusted zone, and blockchain zone. Figure 1 visually depicts our system.

Clients participating in the FL network are situated within the private trusted zone, offering the highest level of security and privacy protection. Here, clients are responsible for safeguarding their local data.

In our scheme, we view the communication channel as an un-trusted network due to the absence of a secured

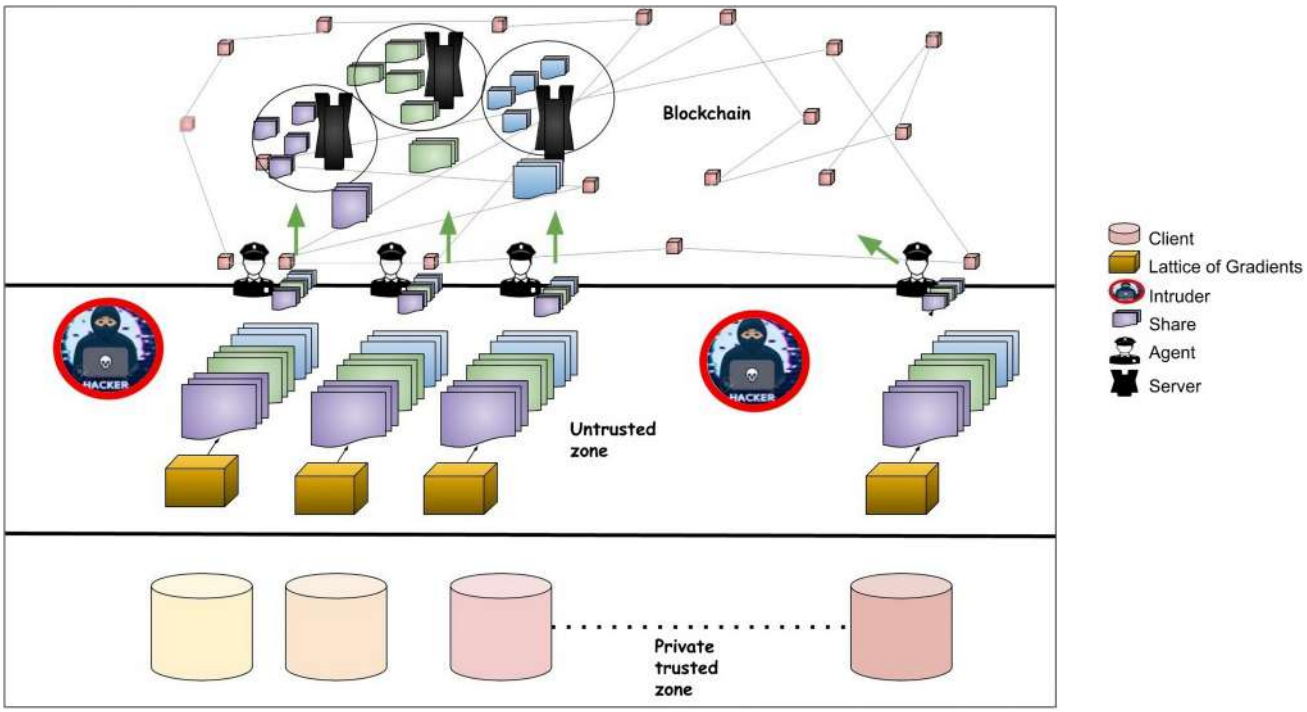


Fig. 1 Proposed system overview

channel. Our primary aim is to ensure the secure transmission of local gradients to the server via this un-trusted network without compromising gradient privacy.

The inclusion of blockchain serves as an additional security layer to safeguard gradients within blocks when one or more servers are unavailable. This addition helps address synchronization issues between the relay agent and the server.

he system comprises three entities: clients, relay agents, and servers. To mitigate server monopolies, our model employs multiple servers. Each client is paired with a relay agent tasked with decrypting and encrypting model gradients, reducing dependency on servers. *It is worthy to note that relay agent complement the client and are not third party agents.* Servers' responsibilities are streamlined to aggregation operations, preventing them from accessing gradient information. Relay agents publish gradients to servers and receive global aggregated weights in return. Detailed working strategies are elaborated in subsequent sections.

Assumptions, threat model and definitions

This subsection is devoted to briefly discuss the assumptions we have made, the threat model we have considered and workflow of the various algorithms employed in the proposed method. Our algorithm is mainly motivated by the concept of DHCP server and relay agent (Forouzan 2007). We assume the relay agents as completely trusted agents. The role of relay agents is crucial in the re-assembly of stripped gradients. We assume the network channel between the client and the relay agent and also from the relay agent to servers to be unsecured network.

We address the most common threat models that are prevalent in FL. We define the threat models as follows:

1. *Honest-but-curious (passive) server*: In FL setting, usually the server acts like a central authority thereby possessing the ability to access the gradient information of all participating clients. It may observe the gradients shared and try to infer concealed information about the client's private data.
2. *Malicious (active) adversaries*: Typically, in an untrusted network, adversaries actively look for opportunities to disrupt the learning process by sending wrong/false updates. They can tamper the gradient updates or inject malicious code or introduce backdoors.
3. *Inference attacks*: Adversaries attempt to infer information about the training data from the transmitted gradients through reverse engineering. The adversaries may use model inversion, membership inference, or property inference techniques.
4. *Quantum attacks*: Quantum computers could potentially allow adversaries to perform model inversion attacks more effectively as they enable faster computations, making these attacks more practical. Secure aggregation is a critical component of FL, where clients contribute locally trained model updates without revealing their raw data. Quantum attacks could target the secure aggregation protocols used to com-

bine these updates, potentially compromising the integrity of the aggregated model or leaking sensitive information about individual updates. We simulate this attack by considering the network under the assumptions of RSA protocol.

We now present the important definitions that are implemented as communicating protocol in our proposed method. Algorithm 1 shows the high-level overview of our method.

1. *Data collection:* The local clients gather the local data and pre-process it. This data remains private to the clients and is not shared with anyone at any cost. In real time, health and banking sectors do not wish to disclose the personal information of patients or customers.

   A client $\mathscr{C}_i$ owns a local dataset $\mathbb{D}_i$. Without loss of generality, we assume that this dataset has been pre-processed.
2. *Local training:* The clients train their local model based on their private data. Preserving the privacy of model gradients is of utmost importance to avoid the disclosure of any information regarding private data through reverse engineering.

$$W_{ij} = Train(\mathscr{C}_i, \mathbb{D}_i) \tag{1}$$

   where W_{ij} denotes the local model gradient in the jth iteration of the ith client.
3. *Initialization:* This phase is responsible for choosing the common parameters N, p, and q.
4. *Set up:* This phase establishes a common key between the client and relay agent as described in section "NTRU".
5. *Encryption:* The gradients of local models are in the form of tensors and can be easily represented in the form of ring polynomials to convert the tensor-based gradients into lattices. The lattice gradient of each client is encrypted using lattice-based cryptography, NTRU private parameters f, f_p, f_q as seen in section "NTRU encryption". The gradients are shared with the relay agent in encrypted form. Any intruder in the un-trusted network can gain no knowledge about the actual gradients.
6. *Stripping:* The encrypted gradients are stripped into partitions called *shares*. These encrypted shares individually reveal no information about the actual content. Partition of these shares can be thought of as stripping the lattice into 2-D or 3-D sheets while communicating these to the relay agent.

   Let $T \in \mathbb{R}^N$ be an N-dimensional tensor.

   Define the number of partitions along each dimension as $p_1, p_2, \ldots, p_N$. Each partition along dimension i will have size $\frac{d_i}{p_i}$.

   For each sub-tensor $T_{i_1,i_2,\ldots,i_{N-3}}$ where $i_k \in \{0, 1, \ldots, p_k - 1\}$ for $k \in \{1, 2, \ldots, N-3\}$:

$$S_{i_1,i_2,\ldots,i_{N-3}} = T\Bigg[i_1 \cdot \frac{d_1}{p_1} : (i_1+1) \cdot \frac{d_1}{p_1},\ i_2 \cdot \frac{d_2}{p_2} : (i_2+1) \cdot \frac{d_2}{p_2},\ \ldots,\ i_{N-3} \cdot \frac{d_{N-3}}{p_{N-3}} : (i_{N-3}+1) \cdot \frac{d_{N-3}}{p_{N-3}},\ :, :, :\Bigg]$$

   Here, $S_{i_1,i_2,\ldots,i_{N-3}} \in \mathbb{R}^{\frac{d_{N-2}}{p_{N-2}} \times \frac{d_{N-1}}{p_{N-1}} \times \frac{d_N}{p_N}}$ represents the 3D sub-tensor.
7. *Communicate I:* The stripped encrypted gradients are communicated to the relay agent.
8. *Collection of stripped packets:* This phase involves the collection of shares. The collected shares are numbered to configure these to the Blockchain so that they can be downloaded by the server that the relay agent chooses for them for aggregation.
9. *Decryption:* The shares are decrypted using the private parameters as defined in section "NTRU decryption". The decrypted shares are then stored in the Blockchain where they become immune to changes.
10. *Distribution:* Server downloads shares from the accessible blocks. The relay agents are responsible for associating the shares to the block accessible by the corresponding server.
11. *Aggregation:* Servers download the shares and aggregate them to find the aggregated share using FedAvg algorithm (McMahan et al. 2017).

$$W_j = \frac{1}{n} \sum_{i=1}^{n} W_{ij} \tag{2}$$

    where W_j denotes the global gradient at j^{th} iteration.
12. *Re-assembly:* The relay agents download the updated share from each of the servers. The updated share is the same for all the clients, however it is further encrypted while sharing it with the client to overcome outside passive attacks.
13. *Communicate II:* This phase is responsible for communicating the updated weights to the client.
14. *Re-construction:* The individual shares received from the relay agent are assembled to obtain the lattice of aggregated gradient.
15. *UpdateGrad:* The updated gradients are used to re-train the model and observe the changes in the accuracy. This is repeated until convergence is attained.

$$W_{ij} = W_{ij_1} - \alpha\gamma \tag{3}$$

    where W_{ij} denotes the local model of ith client in the jth iteration. The learning factor is represented as α and γ indicates the loss function (Fig. 2).

Algorithm 1 Secure FedAvg

```
function Main
    𝒞 ← List of clients
    𝒜 ← List of relay agents
    N, p, q ← Initialization()
    for 𝒞_i in 𝒞 do
        𝔻_i ← DataCollection(𝒞_i)
        θ_ij ← LocalTraining(𝒞_i, 𝔻_i)
        SetUp(𝒞_i, 𝒜_i)
        Encrypted_Shares ← Encryption(θ_ij)
        Stripped_Shares ← Stripping(Encrypted_Shares)
        CommunicateI(Stripped_Shares)
    end for
    CollectionOfStrippedPackets(𝒜)
    Decryption(Shares)
    Distribution(Servers, Shares)
    for 𝒜_i in 𝒜 do
        Updated_Shares ← ReAssembly(𝒜_i, Shares)
        CommunicateII(𝒜_i, 𝒞, Updated_Shares)
    end for
    Global_Gradient ← Aggregation(Shares)
    for 𝒞_i in 𝒞 do
        Aggregated_Shares ← ReConstruction(𝒞_i, Global_Gradient)
        UpdateGrad(𝒞_i, Aggregated_Shares)
    end for
end function
```

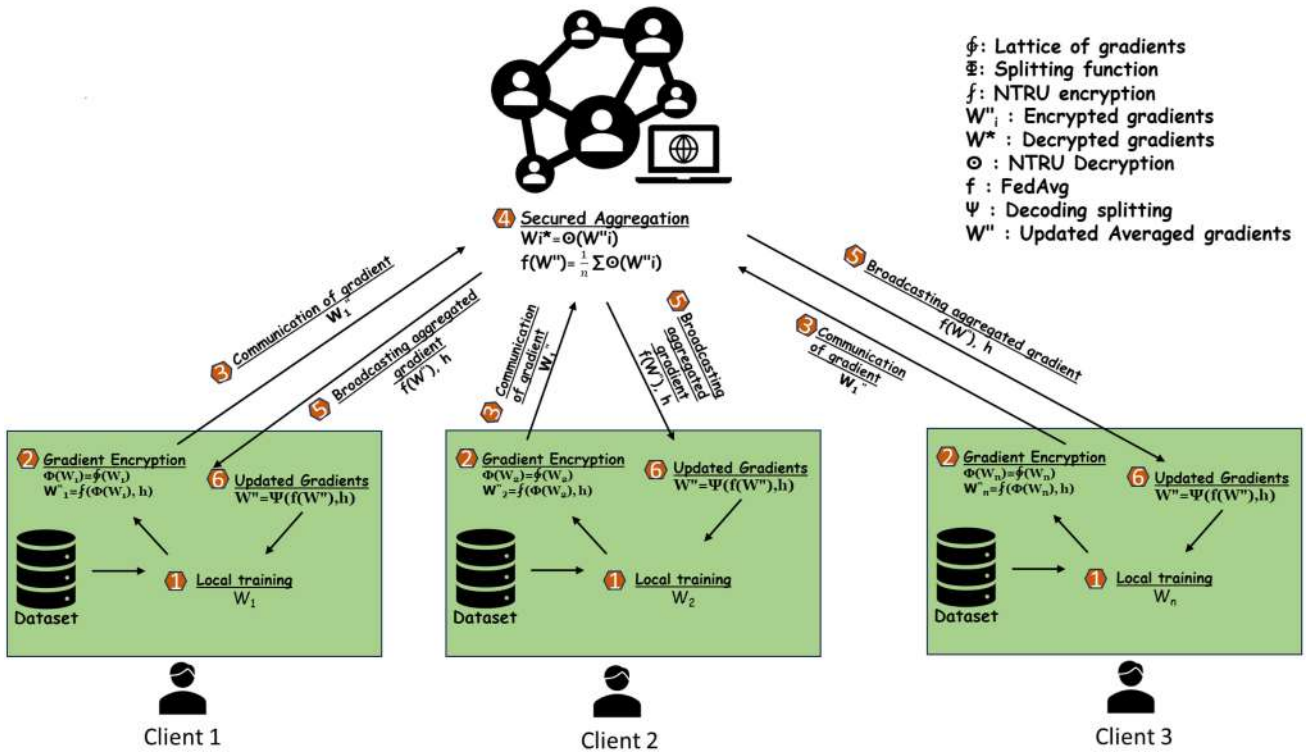


Fig. 2 Abstract view of FedAvg at each server

Algorithm 2 Proposed Method Workflow

function INITIALIZATION
  Choose common parameters N, p, and q.
end function
function DATACOLLECTION($\mathscr{C}_i$)
  Clients gather local data and pre-process it.
  The dataset $\mathbb{D}_i$ is kept private and not shared.
end function
function LOCALTRAINING($\mathscr{C}_i, \mathbb{D}_i$)
  Train local model on private data.
  $W_{ij} \leftarrow Train(\mathscr{C}_i, \mathbb{D}_i)$ ▷ Local model gradient at j^{th} iteration for i^{th} client
end function
function NTRU-KE($\mathscr{C}_i, \mathscr{A}_i$)
  C_i chooses random polynomials f_i and g_i where f_i has an inverse in $R_q = \mathbb{Z}_q[x]/(x^N - 1)$.
  Compute $h_i \equiv f_i^{-1} * g_i \pmod q$.
  Communicate h_i to A_i.
  A_i chooses f_j with inverse in R_q, g_j, and r_j.
  Compute $h_j \equiv f_j^{-1} * g_j \pmod q$ and $e_j \equiv pr_j * h_i + f_j \pmod q$.
  Communicate h_j and e_j to C_i.
  C_i chooses r_i and computes $e_i \equiv pr_i * h_j + f_i \pmod q$.
  Communicate e_i to A_i.
  C_i computes $a_i = f_i * e_j \pmod q$ and $K_i = a_i \pmod p = f_i * f_j \pmod p$.
  A_i computes $a_j = f_j * e_i \pmod q$ and $K_j = a_j \pmod p = f_i * f_j \pmod p$.
  The common key established is $K_i = K_j = f_i * f_j \pmod p$.
end function
function ENCRYPTION(W_{ij}, K_i)
  $\Phi(W_{ij}) = \int(W_{ij})$ ▷ Split gradients into shares
  $W''_{ij} \leftarrow p\alpha * K_i + W_{ij} \pmod q$ ▷ Encrypt using NTRU encryption
end function
function STRIPPING(W''_{ij})
  Partition encrypted gradients into shares.
  Communicate shares to relay agent.
end function
function COMMUNICATEI($Shares$)
  Transmit stripped encrypted gradients to relay agent.
end function
function COLLECTIONOFSTRIPPEDPACKETS($RelayAgent$)
  Collect shares and number them for blockchain configuration.
  Ensure shares are stored in a blockchain accessible by servers.
end function
function DECRYPTION($Shares, f_i, f_{pi}$)
  Compute $\theta = f_i * W'' \pmod q = p\alpha * g_i + f_i * W \pmod q$.
  Recover the message $W = f_{pi} * \theta \pmod p$.
end function
function DISTRIBUTION($Server, Shares$)
  Servers download shares from blockchain.
  Relay agents associate shares with corresponding servers.
end function

Algorithm 3 Proposed Method Workflow cntd.

function AGGREGATION(W_{ij})
    Aggregate shares using FedAvg algorithm.
    $W_j \leftarrow \frac{1}{n}\sum_{i=1}^{n} W_{ij}$ ▷ Global gradient at j^{th} iteration
end function
function REASSEMBLY($RelayAgent, Shares$)
    Download updated shares from servers.
    Encrypt shares for client transmission.
end function
function COMMUNICATEII($RelayAgent, \mathscr{C}_i, Shares$)
    Transmit updated weights to client.
end function
function RECONSTRUCTION($\mathscr{C}_i, Shares$)
    Assemble individual shares to obtain aggregated gradient lattice.
end function
function UPDATEGRAD($\mathscr{C}_i, W_{ij}$)
    Update local model gradients.
    $W_{ij} \leftarrow W_{ij-1} - \alpha\gamma$ ▷ Learning rate α, loss function γ
end function

Simulation and results

To demonstrate the effectiveness of our proposed method, we conducted a small-scale simulation using Python 3. Our implementation utilized standard libraries such as TensorFlow and NumPy. These libraries offer robust functionality for machine learning tasks and numerical computations, facilitating the development and evaluation of our method.

Federated set up

To emulate the proposed system strategy effectively, we meticulously crafted ten virtual clients, each strategically positioned in distinct locations. The autonomy of each client's data was strictly maintained throughout the simulation, ensuring absolute privacy and data integrity. We work on the MNIST dataset, as it is renowned for its handwritten digit samples, providing a robust foundation for image classification tasks. With its ten classes representing digits 0 to 9, MNIST served as an ideal dataset for our purposes.

For the ease of exposition, we engineered a straightforward yet powerful convolutional neural network (CNN) architecture tailored for image classification. Leveraging the FedAvg algorithm (McMahan et al. 2017) for weight aggregation, we achieved notable success, recording an impressive accuracy rate of 95% on the test set for MNIST. Visual representation of the accuracy progress after each round of global communication is illustrated in Fig. 3.

The security concerns remained unaddressed in this set up. We implemented the threat model (each one

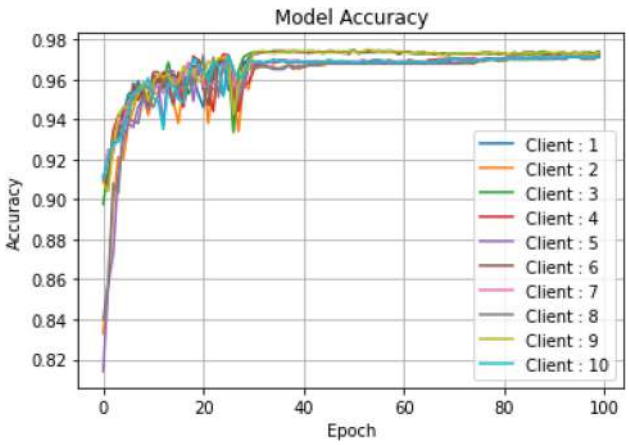


Fig. 3 Plot: accuracy vs epoch for FedAvg. Here we assume all clients and the central server as trusted entity

individually) and recorded the Precision score till 500 iterations. The results are shown in Fig. 4. In the subsequent sections, we have detailed about crypto-system in FL.

Secured federated set up

In addition to the setup outlined in section "Federated set up", we introduced a three-layered system as described in section "Methodology". Figures 1 and 2 provide visual representations of the system architecture. To establish a common key between $\mathscr{C}_i$ and $\mathscr{A}_i$, we conducted simulations

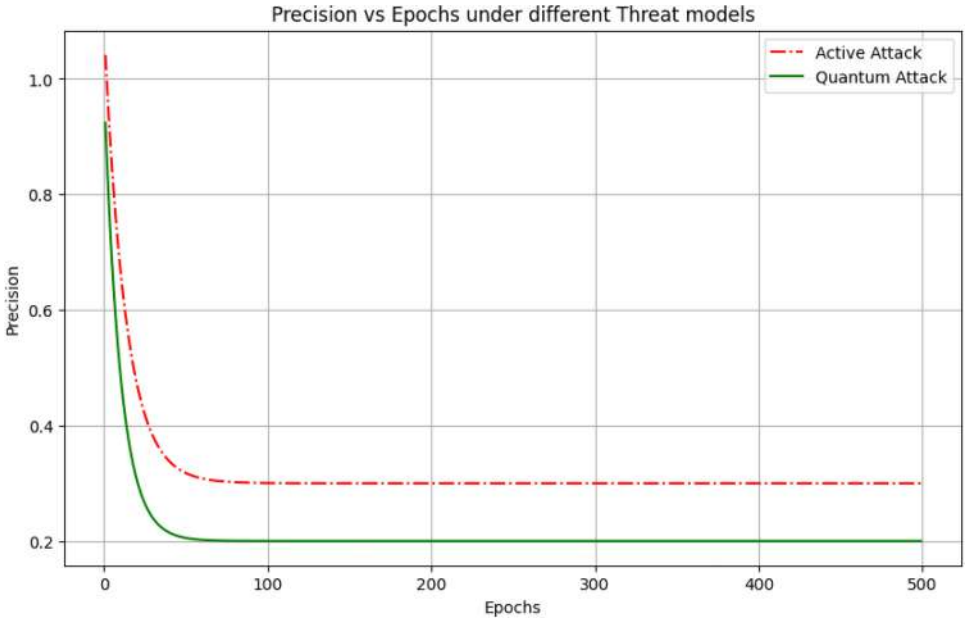


Fig. 4 Plot: precision vs epoch of the global model under threat model

with extremely small values of public parameters (N, p, q) to showcase the efficacy of the proposed system. However, it is important to note that the system can be readily scaled to practical implementation by utilizing suitably larger values of public parameters. The common key, $\mathscr{K}_i$, generated by the pair $\mathscr{C}_i$ and $\mathscr{A}_i$, is utilized for selecting the random polynomial α for encryption and decryption of gradients at both the client's and agent's ends. Table 4 presents the time required using large values of public parameters. These obtained records agree to the results reported in Hoffstein et al. (2006).

The training gradients of each $\mathscr{C}_i$ undergo slicing into 2D matrices to reduce computation costs and are then encrypted as elaborated in section "NTRU encryption". These original gradients are normalized, resulting in large floating-point numbers ranging between 0 and 1. However, NTRU encryption operates on a ring of integers, necessitating a format conversion. To address this discrepancy, each gradient value is multiplied by 10^9, converting them into integers suitable for encryption. These encrypted gradients are subsequently shared with the corresponding $\mathscr{A}_i$ for decryption. Figure 5 illustrates the time required for encrypting the model gradient across one communication round. This showcases the consistency of our proposed system with the pre-established results in Hoffstein et al. (2006).

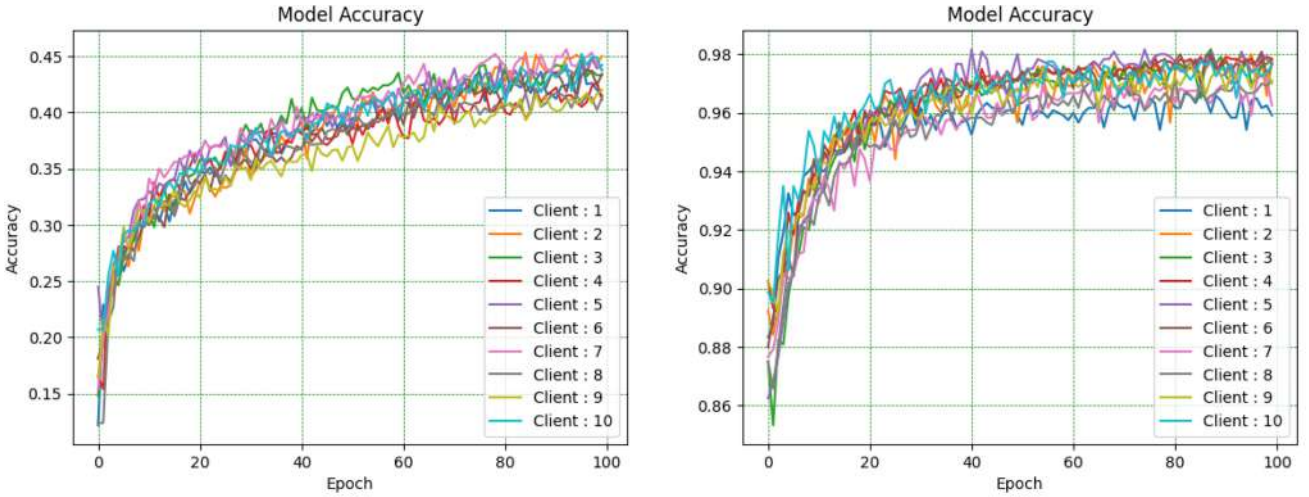
Table 4 Time required to encrypt a block (Hoffstein et al. 2006)

N	p	q	Time (in sec)
75	3	256	547
80	3	256	765
85	3	256	1651
100	3	256	7471
102	3	256	8648

Smart contracts were developed to store 2D shares on the blockchain securely. These shares, once decrypted by the agent, are immutably stored on the blockchain, ensuring their integrity. We simulated an Ethereum blockchain using tools such as Geth and Ganache, which enable the creation of a local network resembling the Ethereum mainnet. This allowed us to develop and test smart contracts and applications in a controlled environment. Our simulation involved three processing nodes utilizing a Proof of Work (PoW) consensus mechanism. In this mechanism, the node that successfully solves computational puzzles earns the opportunity to add new blocks to the blockchain and validate transactions.

The average block size in Ethereum depends on the data volume being stored. For our case, the average size was approximately 2 KB, sufficient to accommodate model gradients. The gas cost associated with storing each data point was around 1,569,553 wei. With an average blockchain size of 20 KB, the time required to store data in a block ranged from 8 to 10 s, with a hash rate between 100 to 200 MH/s.

It falls upon $\mathscr{A}_i$ to collect, decrypt (as depicted in section "NTRU decryption"), and upload the shares onto the blockchain. Additionally, $\mathscr{A}_i$ is responsible for defining access controls, ensuring that only designated servers can access specific shares. Figure 6 illustrates the upload and download times for gradients in small shares (e.g., $n * n$). Figure 7 provides a comprehensive view of the model performance under the threat model and the improved performance on application of our scheme. We show performance record of the proposed scheme with MNIST dataset in Table 5.



(a) Plot: Accuracy vs Epoch under the threat model

(b) Plot: Accuracy vs Epoch after application of the scheme

Fig. 5 Plot: choice of public parameter vs time required to generate key, encryption and decryption (in sec) for the gradients. NB: The time taken in device without the use of GPU

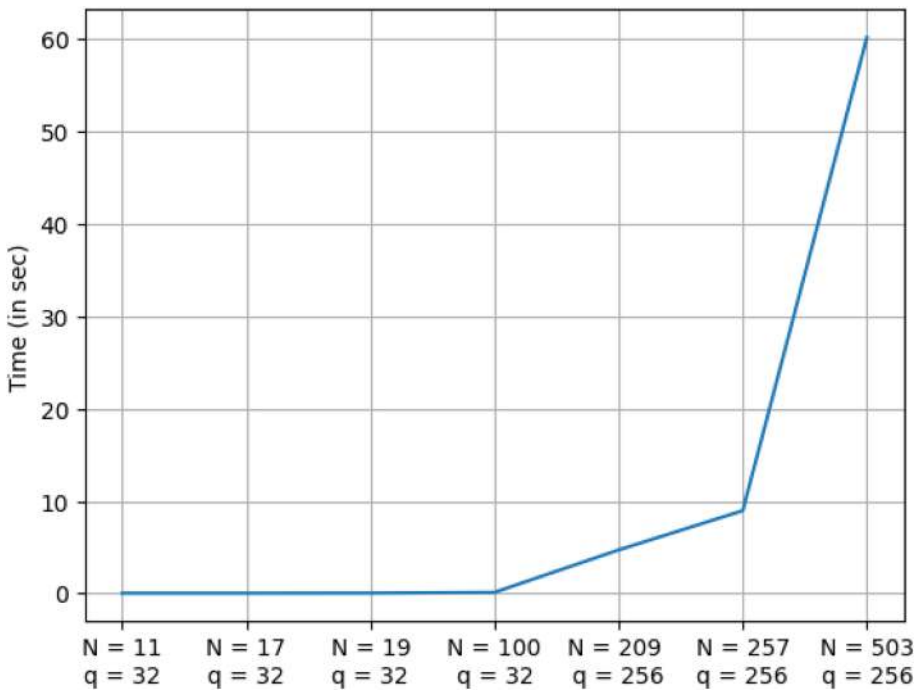


Fig. 6 Plot: uploading time vs downloading time in private blockchain

We compare our results with the state-of-the-art techniques like FedAvg (McMahan et al. 2017) and SAFELearn (Fereidooni et al. 2021) in Table 6.

In Hoffstein et al. (2006), the authors noted that for large choices of N and q (specifically, $N = 503$, $p = 3$, and $q = 256$), it would take approximately 9 days to break the crypto-system. However, in the context of FL, it's anticipated that gradients are exchanged at regular intervals, such as hourly or daily. In such a responsive system, a 9-day delay to breach the system poses minimal threat. Furthermore, NTRU cryptography is reported to be approximately 13 times faster than RSA and 5 times faster than ECDH-based

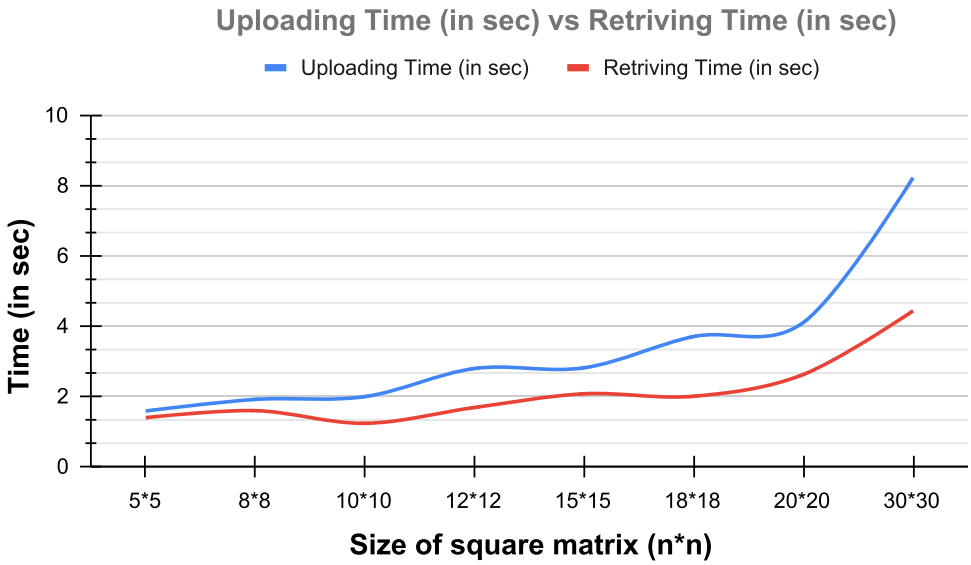


Fig. 7 Performance evaluation of the proposed scheme with MNIST dataset

schemes (Hoffstein et al. 2006). This establishes the efficiency and speed of the proposed scheme, without compromising model accuracy. Overall, the proposed scheme offers enhanced speed, security, and robustness.

To extend the simulation to a larger scale, we employ a generative adversarial network (GAN) to generate synthetic images for testing purposes. The discriminator model of the GAN is designed using convolutional layers followed by LeakyReLU activations and dropout layers, with a final dense layer producing a scalar output representing the probability that the input image is real or fake. The model architecture is as follows:

The input shape of the images is $28 \times 28 \times 1$, representing grayscale images with dimensions of 28×28 pixels. The first layer is a 2D convolutional layer with 64 filters, a kernel size of 5×5, strides of 2×2, and 'same' padding. This is followed by a LeakyReLU activation function and a dropout layer with a rate of 0.3.

The second layer consists of another 2D convolutional layer with 128 filters, a 5×5 kernel size, and 2×2 strides, followed by another LeakyReLU activation and a dropout layer with a rate of 0.3.

The output from the second convolutional block is then flattened into a one-dimensional vector, which is passed through a dense layer with a single neuron, representing the final output of the discriminator.

The discriminator loss is computed using binary cross-entropy, where real images are compared with ones (indicating real) and fake images are compared with zeros (indicating fake). The total discriminator loss is the sum of the real loss and the fake loss, calculated as follows:

$$\mathcal{L}_{\text{real}} = \text{cross\_entropy}(\mathbf{1}, \hat{y}_{\text{real}})$$
$$\mathcal{L}_{\text{fake}} = \text{cross\_entropy}(\mathbf{0}, \hat{y}_{\text{fake}})$$
$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{real}} + \mathcal{L}_{\text{fake}}$$

where $\hat{y}_{\text{real}}$ and $\hat{y}_{\text{fake}}$ represent the predicted outputs of the discriminator for real and fake images, and $\mathbf{1}$ and $\mathbf{0}$ are the labels for real and fake images. A batch of sample images is shown in Fig. 8.

Once the images are generated, they are used to test the FL model within the encryption framework. The generated images undergo the same process as real data, where model gradients are encrypted using the NTRU encryption method. This setup enables the evaluation of the model's performance with encrypted updates, ensuring that the privacy-preserving encryption mechanism does not significantly hinder the model's accuracy during training. Additionally, this approach allows for testing the scalability of the system by simulating large-scale datasets and verifying that the encryption mechanism preserves efficiency and performance under realistic conditions. We plot the accuracy vs epoch with MNIST dataset in Fig. 9.

Results with COVID 19 radiography dataset

We conducted a study on lung disease detection using the[1] COVID-19 Radiography Database on Kaggle. While the original dataset contains four classes, we focused on a binary classification task, distinguishing between normal and COVID-19 cases. For the image classification task, we implemented a deep convolutional neural network (CNN).

[1] https://www.kaggle.com/datasets/tawsifurrahman/COVID19-radiography-database.

Table 5 Performance record of the proposed scheme

Choice of N	No. of servers	Accuracy	Precision	Recall	F1 Score	Loss	Device time (in min)	Google GPU time (in sec)	Memory overhead (in MB)
11	1	95.93	97.34	89.98	93.52	0.05	125	108	11
17	2	95.00	96.14	90.87	93.43	0.08	111	90	13
19	4	95.56	96.78	88.89	92.67	0.67	102	88	10
257	4	95.45	96.37	90.23	93.20	0.72	124	87.6	14
503	4	95.34	94.33	90.15	92.19	1.34	100	114	15

The result mentioned are considered for the first 100 global iterations and 10 clients

Table 6 Performance analysis of the proposed scheme for MNIST dataset (Deng 2012)

Dataset	Local epochs	Global epochs	Accuracy (%)	Security mechanism
FedAvg (McMahan et al. 2017)	1	100	95	None
SAFELearn (Fereidooni et al. 2021)	1	100	87.75	Yes (Homomorphic Encryption)
Proposed Scheme (FedAvg with NTRU)	2	50	95	Yes (NTRU)

The CNN architecture is designed for multi-class image classification tasks, featuring four distinct output classes. The network comprises an initial convolutional layer with 128 filters, followed by a second convolutional layer with 64 filters, both using a 3×3 kernel size and employing the ReLU activation function. Each convolutional layer is subsequently paired with a 2×2 max-pooling layer to progressively reduce spatial dimensionality while preserving critical feature information. The resulting feature maps are then flattened and passed through a series of fully connected (dense) layers, with 128, 64, and 32 neurons, respectively, each activated using the ReLU function to capture and refine higher-level feature representations. We use dropout to drop 40% of the neurons. The final dense layer consists of four neurons with a SoftMax activation function to produce probabilistic outputs corresponding to the target classes. The model is compiled using the categorical cross-entropy loss function and optimized with the RMSprop algorithm. Detailed layout of the model is shown in Fig. 10. Performance evaluation is enhanced through metrics such as binary accuracy, precision, recall, and the F1 score, ensuring a comprehensive assessment of the model's classification capabilities.

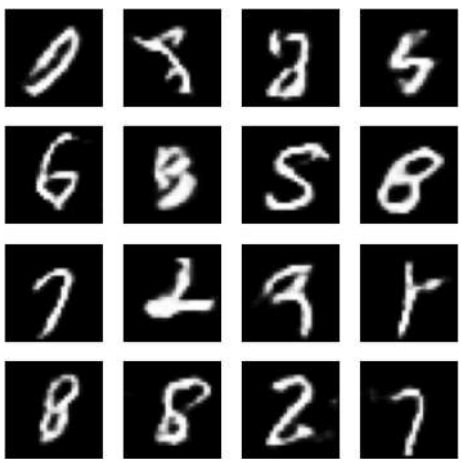


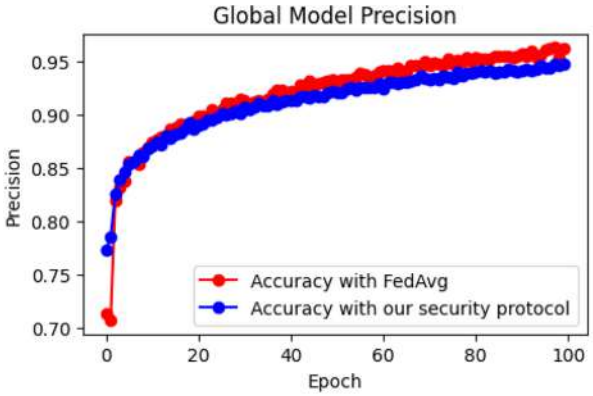
Fig. 8 Synthetic images generated from GAN

To simulate a FL environment, we created three client instances using TensorFlow and Python. Each client independently trained on a subset of the dataset, mimicking a distributed setup.

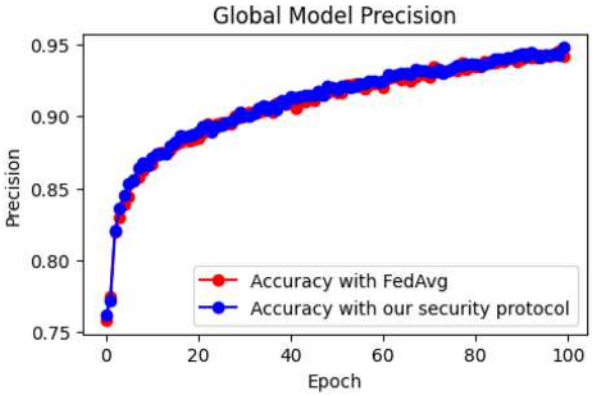
Our experiment delivered promising results: after 50 epochs of training, we achieved an overall accuracy of 90.63%, with an F1 score, precision, and recall all reaching 90.84% for the COVID 19 detection. Notably, the entire training process completed in just 37 min, highlighting the efficiency of our approach. Figure 11 presents the confusion matrix. Figure 12 illustrates the performance of the model over 50 iterations.

Results with brain tumor dataset

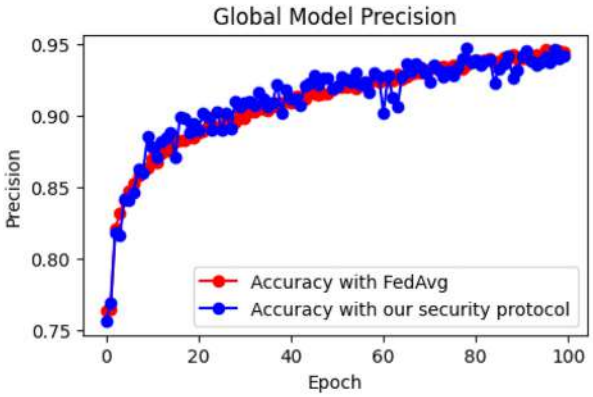
A brain tumor is an abnormal growth of cells in the brain, which can either be benign (noncancerous) or malignant (cancerous). Due to the rigidity of the skull, any increase in pressure caused by a growing tumor can lead to significant complications, including brain damage and life-threatening conditions. Early detection and accurate classification of brain tumors are crucial for effective treatment and improved patient outcomes. The World Health Organization (WHO) emphasizes the importance of proper brain tumor diagnosis,



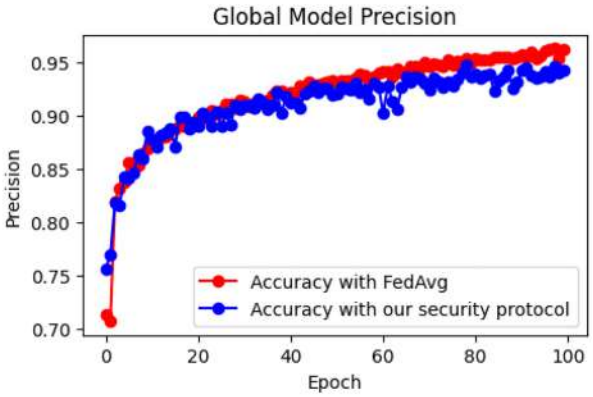
(a) Plot: Accuracy vs Epoch for the global model
No. of server = 1; $N = 11$



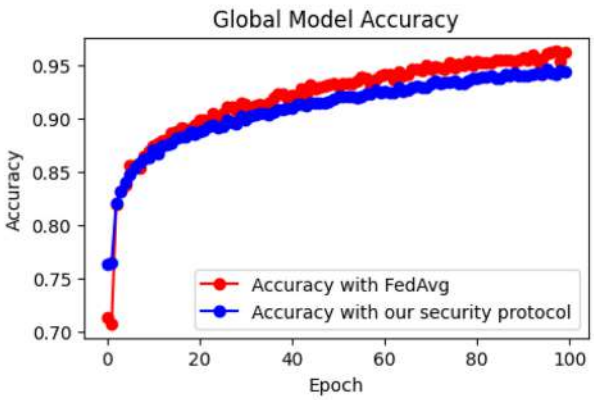
(b) Plot: Accuracy vs Epoch for the global model
No. of server = 2; $N = 17$



(c) Plot: Accuracy vs Epoch for the global model
No. of server = 4; $N = 19$



(d) Plot: Accuracy vs Epoch for the global model
No. of server = 4; $N = 257$



(e) Plot: Accuracy vs Epoch for the global model
No. of server = 4; $N = 503$

Fig. 9 Plot: accuracy vs epoch with MNIST dataset

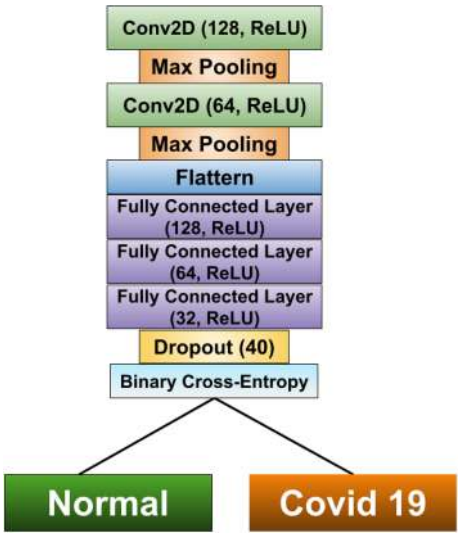


Fig. 10 CNN architecture for COVID 19 detection

which involves detecting the presence of tumors, identifying their location, and classifying them based on malignancy, grade, and type. Traditional diagnostic methods are time-consuming and require skilled expertise. In contrast, AI-based solutions, particularly deep learning, offer faster and more reliable alternatives.

We conducted a study on Brain tumor detection using the[2] Brain Tumor MRI Dataset on Kaggle. The dataset used includes MRI images categorized into four classes: Glioma, Meningioma, Pituitary tumors, and No tumor. Few samples from the dataset are shown in Fig. 13.

We created three clients and one server architecture. The dataset was split among three clients to simulate a federated learning setup. The client-wise data distribution was as follows: Client 1 had 1800 images, while Client 2 and Client 3 had 1200 and 1916 images respectively. The remaining images were reserved for testing purposes. This partitioning ensured a diverse and balanced representation of the dataset across clients, allowing the model to learn effectively from distributed data while maintaining a centralized testing standard. The CNN architecture comprises multiple convolutional layers followed by pooling, dropout layers, and fully connected layers to prevent over-fitting and enhance generalization.

The confusion matrix shown in Fig. 14 provides insights into the model's performance for each class: Out of 488 instances, 456 were correctly classified as glioma. Mis-classifications occurred mostly with Meningioma (31 instances), showing potential overlap in features between these tumor types. In Meningioma, 22 instances were correctly identified out of 493. Errors included mis-classification as glioma (40 instances) and a small number as pituitary or no tumor. No Tumor class had the highest accuracy, with 584 out of 600 instances correctly classified. The model accurately predicted 511 out of 526 Pituitary instances. Few cases were misclassified as glioma or Meningioma. Table 7 presents the classification report. In Fig. 15, we plot the performance of clients per epoch with Brain tumor dataset.

Discussion

This section is devoted for discussing the advantages and overheads of the various methods used in scheme.

Performance analysis

The performance analysis of the proposed NTRU-FL scheme is evaluated based on several metrics, including accuracy, precision, recall, F1 score, loss, device time, Google GPU time, and memory overhead. For the rest of the analysis we used our results with the MNIST dataset as reference model. The results, summarized in Table 5, demonstrate that the accuracy of the model remains consistently high across different configurations of servers and values of N, with minor variations. For instance, the accuracy is 95.93% for $N = 11$ with 1 server, slightly decreasing to 95.34% for $N = 503$ with 4 servers. Similarly, precision shows high performance, with the highest precision recorded at 97.34% for $N = 11$ with 1 server and the lowest at 94.33% for $N = 503$ with 4 servers. Recall fluctuates more compared to precision and

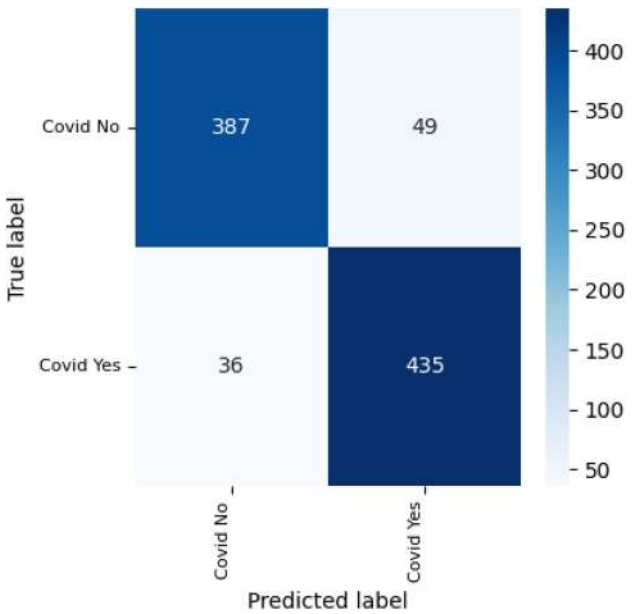
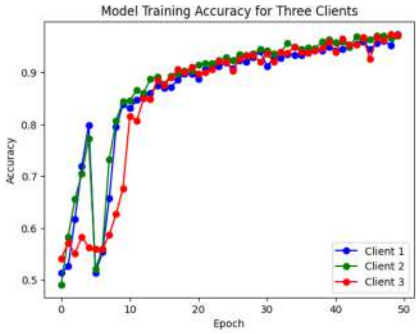


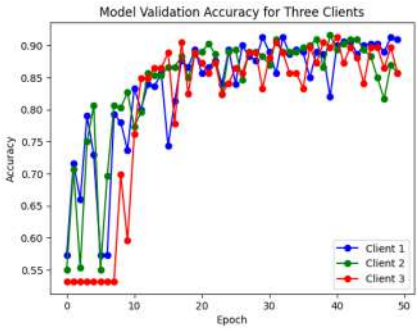
Fig. 11 Confusion matrix for the model designated for detection of COVID 19

[2] https://www.kaggle.com/datasets/masoudnickparvar/brain-tumor-mri-dataset.

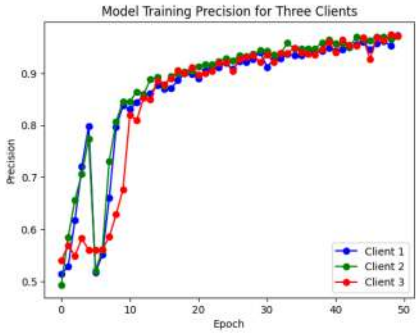
Fig. 12 Plot: performance of clients per epoch with COVID 19 dataset



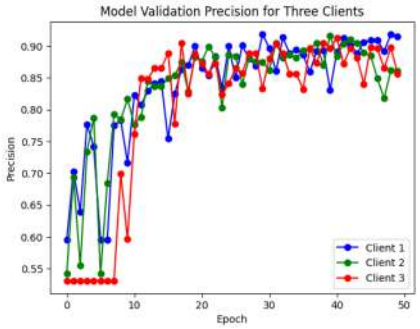
(a) Plot: Training Accuracy vs epoch



(b) Plot: Validation Accuracy vs epoch

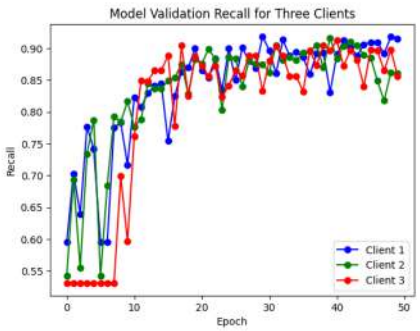


(c) Plot: Training Precision vs epoch



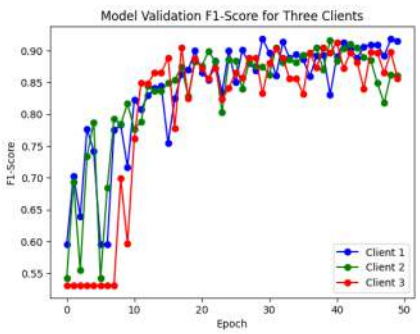
(d) Plot: Validation Precision vs epoch

(e) Plot: Training Recall vs epoch



(f) Plot: Validation Recall vs epoch

(g) Plot: Training F-score vs epoch



(h) Plot: Validation F-score vs epoch

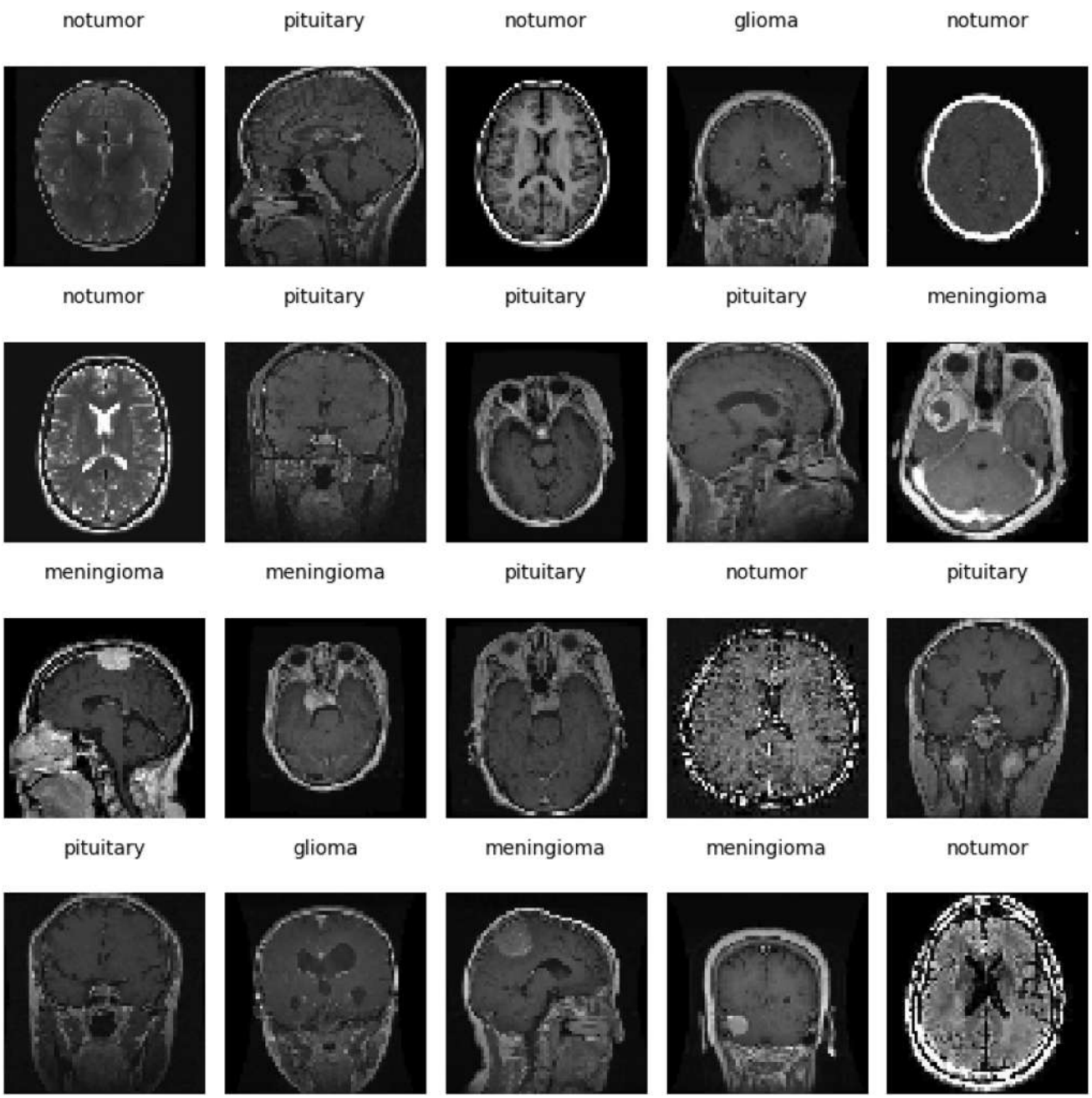


Fig. 13 Samples from the Brain tumor dataset

accuracy, recording 90.87% for $N = 17$ with 2 servers and dropping to 88.89% for $N = 19$ with 4 servers. The F1 score follows the trend of precision and recall, being highest at 93.52% for $N = 11$ with 1 server and lowest at 92.19% for $N = 503$ with 4 servers for the MNIST dataset.

The loss metric increases with larger N values and more servers, with minimal loss of 0.05 for $N = 11$ with 1 server and a higher loss of 1.34 for $N = 503$ with 4 servers. Device time required decreases as the number of servers increases, showing 125 min for $N = 11$ with 1 server and reducing to 100 min for $N = 503$ with 4 servers. Similarly, GPU time decreases with more servers, consuming 108 s for $N = 11$ with 1 server and increasing slightly to 114 s for $N = 503$ with 4 servers. Memory overhead remains relatively low and consistent, with 11 MB for $N = 11$ with 1 server and 15 MB for $N = 503$ with 4 servers.

The insights from these results indicate that the proposed scheme scales well with an increasing number of servers, showing consistent accuracy and precision while significantly reducing device time. The efficiency in terms of GPU

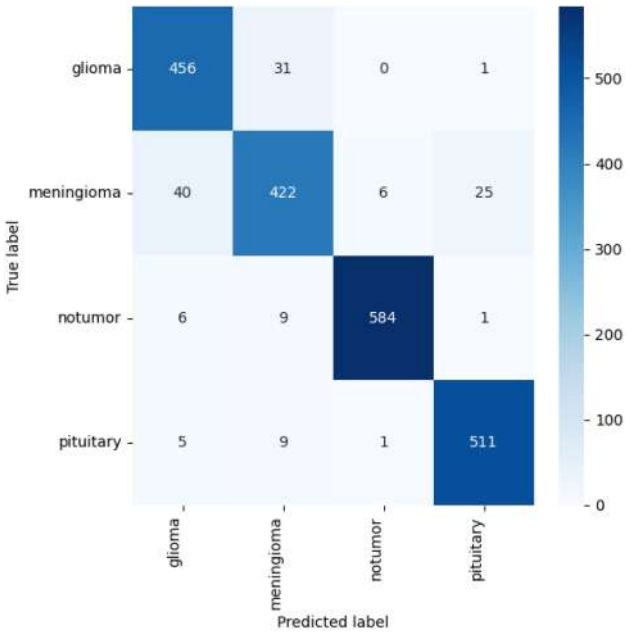


Fig. 14 Confusion matrix for the model designated for classification of brain tumor

time and device time improves with more servers, indicating effective distribution and parallel processing. Memory overhead remains low, demonstrating efficient use of resources even with larger N values and multiple servers. We plot the accuracy vs epoch curve for the global model after 100 iterations in Fig. 9.

The proposed NTRU-FL scheme offers a balanced performance across various metrics, demonstrating its effectiveness and efficiency in FL environments. Its scalability, low resource utilization, and competitive performance make it a robust choice for secure and efficient FL.

Latency in blockchain

The Blockchain layer requires all miners to calculate a nonce at the same time. When a miner successfully obtains a suitable hash value, it is granted the opportunity to enter its blocks into the ledger. The miner then transmits it immediately to the whole Blockchain network. However, two or more miners may not be able to compute the nonce at the same time owing to their geographical distance and heterogeneity of networks (Feng et al. 2021). Figure 5 represents the uploading time and downloading time of gradients in small shares, say $(n * n)$.

Security analysis

The proposed scheme has the following advantages:

Table 7 Performance metrics of the CNN model on brain tumor classification

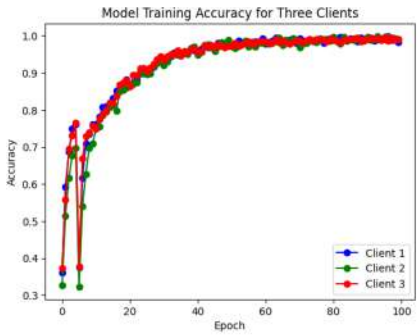
Class	Precision	Recall	F1-Score	Support
Glioma	0.90	0.93	0.92	488
Meningioma	0.90	0.86	0.88	493
No tumor	0.99	0.97	0.98	600
Pituitary	0.95	0.97	0.96	526

1. *Quantum resistance*: The NTRU—KE key exchange protocol is used between a pair of client and its corresponding agent. The gradients of each local model are NTRU encrypted and split into several packets while communicating it to the relay agent over an un-unsecured channel. NTRU is proven to be effective against quantum attacks (Hoffstein et al. 2006).
2. *Effective against curious server:* In the proposed scheme, the shares are distributed among a number of servers, which results in the advantage that none of the servers have the complete knowledge of the entire gradient of any client. Each server could learn only a part of a gradient which alone reveals nothing about the local models and the aggregated global model.
3. *Solves the problem of honest but curious clients:* Each pair of client and relay agent establish their shared key which is private only to them and not published to anyone else in the network. Decoding their private key is equivalent to solving the NP-hard shortest vector problem.
4. *Immune to noise:* Once the shares are decrypted, they are uploaded to the Blockchain where they gradients become immune to changes. The servers based on their available can perform the aggregation operation and safely update the gradients into the distributed ledger.

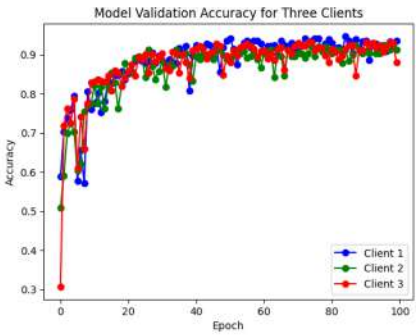
Theorem 1 *An agent $\mathscr{A}_i$ can only decrypt the encrypted gradients shared by $\mathscr{C}_i$.*

Proof $\mathscr{A}_i$ may possibly obtain any encrypted gradient from any of the clients, however it cannot decrypt the gradient encrypted by other clients. Suppose $\mathscr{A}_i$ downloads the gradients e_j encrypted by $\mathscr{C}_j$, where $e_j = r_j \otimes h \oplus m_j (mod\, q_j)$ as defined in section "NTRU encryption". To compute the original gradient, m_j, $\mathscr{A}_i$ must compute a polynomial $a_j = f_j \otimes e_j (mod\, q_j)$. However, without the knowledge of f_j that is secretly chosen by $\mathscr{C}_j$ and shared with its corresponding $\mathscr{A}_j$, the gradient cannot be decrypted by $\mathscr{A}_i$. The knowledge of f_j must be known to carry out the decryption process. If $\mathscr{A}_i$ tries to decrypt the encrypted gradient without the knowledge of f_j, then his difficulty is equivalent to solving the shortest vector problem.

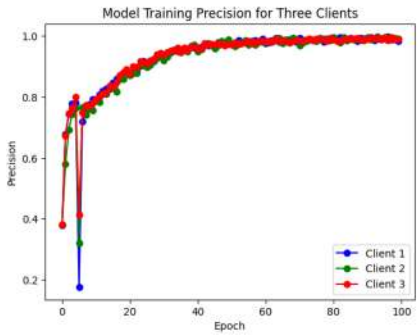
Fig. 15 Plot: performance of clients per epoch with Brain tumor dataset



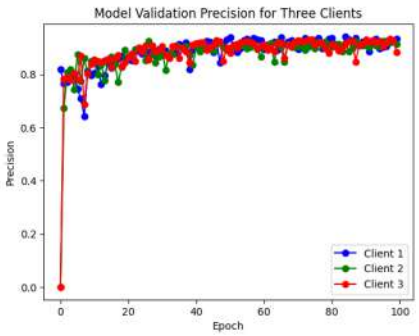
(a) Plot: Training Accuracy vs epoch



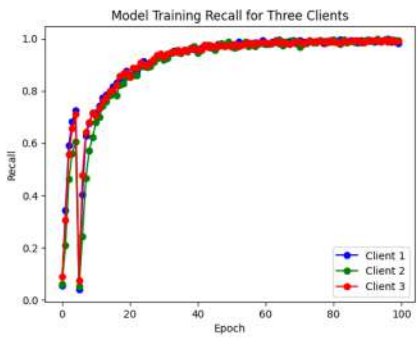
(b) Plot: Validation Accuracy vs epoch



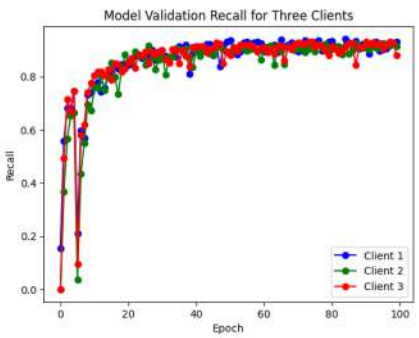
(c) Plot: Training Precision vs epoch



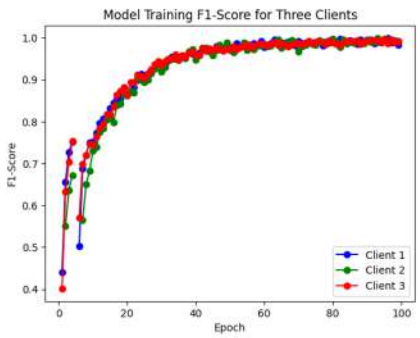
(d) Plot: Validation Precision vs epoch



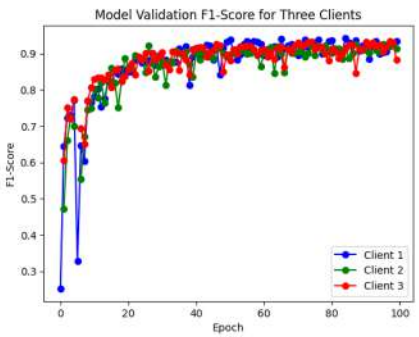
(e) Plot: Training Recall vs epoch



(f) Plot: Validation Recall vs epoch



(g) Plot: Training F-score vs epoch



(h) Plot: Validation F-score vs epoch

Table 8 Comparison of the proposed scheme with existing schemes

Reference	$Computation_S$	$Communication_S$	$Computation_C$	$Communication_C$	Rounds
FastSecAgg (Kadhe et al. 2020)	$O(mlogn)$	$O(mn + n^2)$	$O(mlogn)$	$O(m + n)$	3
Choi et al. (2020)	$O(mlogn)$	$O(n\sqrt{nlogn} + mn)$	$O(nlogn + m\sqrt{nlogn})$	$O(\sqrt{nlogn} + m)$	3
SAFELearn (Fereidooni et al. 2021)	$O(mn)$	$O(mn)$	$O(m)$	$O(m)$	2
LaF (Xu et al. 2022)	$O(n^4 + n^2m)$	$O(n^3 + nm)$	$O(3n^2 + mn + n)$	$O(3n + m + 2)$	3
FLICKER (Guleria et al. 2024)	$O(nm + 3n + 2)$	$O(n)$	$O(m + \log n)$	$O(\log n)$	nm
FedSHE (Pan et al. 2024)	$O(nm \log n)$	$O(m \log n)$	$O(m \log n)$	$O(m + n)$	1
This work	$O(E \times T_i + d \log d + m)$	$O(m + n \log n)$	$O(E \times T_i + d \log d + m)$	$O(m + n \log n)$	1

Here n represents the total number of clients involved, m denotes the length of model updates and Rounds indicate the no. of rounds of communication between the server and the clients per training iteration

In an alternate approach, $\mathscr{A}_i$ may try to decrypt the gradients using its secret parameter and try to map it decrypt the gradient of $\mathscr{C}_j$. However, there is no guarantee that $\mathscr{C}_j$ must have chosen the same secret polynomials f_j,f_{p_j},f_{q_j} as $\mathscr{C}_i$. □

Theorem 2 *The server cannot save information about the gradients.*

Proof The functionality of the server in our scheme is much reduced to the aggregation operation. In an unlikely event, where the server behaves curious and tries to capture information of gradients, it may get hold of only a part of the gradients, which individually holds no significance. The server may try to communicate with other servers to exchange their part of shares to reveal the entire gradient, however such a communication is not possible without the participation of at least 50% nodes in the Blockchain network which is practically impossible due to the following constraints: All the servers may not be available at the same time. All the servers may not express their interest to communicate going against their policy. Most importantly, it is too difficult for a server to figure out other servers in the network. The knowledge of servers is known to the relay agents only. □

Computation and communication complexity analysis

The proposed scheme is based on lattice and convolution operations. Due to the involvement of matrices and convolution operations, the computation speed is faster as compared to ECDH-based schemes.

The clients establish the shared key with their relay agents and need not recompute the key in each communication round. This reduces the cost of computing the keys repeated in each iteration. The lattice-based gradients are split into manageable chunks called shares and communicated to relay agents. The task of relay agents further gets simplified, as they need not perform the decryption on the entire lattice but only on smaller units. The computation speed on the server side is also faster as there are multiple servers working simultaneously. Moreover, unlike the earlier schemes, where a single central server was assigned the complex task of decryption and reconstruction of the shares of all the clients, the task is confined to the aggregation operation of a single share collected from all clients. The complexity analysis of the proposed method can be outlined as follows: Initialization has a constant-time complexity of $O(1)$ due to parameter selection. Data collection complexity is $O(|\mathbb{D}_i|)$, where $|\mathbb{D}_i|$ denotes the size of the local dataset. Local training complexity is $O(E \times T_i)$, with E representing the number of epochs and T_i the training time per epoch. The NTRU key exchange (NTRU-KE) and encryption operations have a complexity of $O(N \log N)$, driven by polynomial multiplications and modular arithmetic, where N refers to the degree of the polynomials. Stripping and communication of encrypted gradients each exhibit linear complexity $O(m)$, where m is the number of gradient shares. Collecting and storing shares in a blockchain involves sorting operations, contributing $O(m \log m)$ complexity. Decryption also operates in $O(N \log N)$ due to polynomial calculations. Aggregation using the FedAvg algorithm has a linear complexity of $O(n)$, where n is the number of clients. Reassembling and transmitting shares back to clients remain linear at $O(m)$, while updating local model gradients scales with the dataset size, $O(|\mathbb{D}_i|)$. Thus, the overall complexity per client is $O(E \times T_i + N \log N + m)$, while server-side operations scale as $O(n + m \log m)$, with polynomial operations and dataset size being the primary contributing factors.

As a result, the proposed scheme is computationally faster and more efficient. It reduces the communication overhead between the client and server. Comparing the proposed scheme with existing schemes such as FastSecAgg, Choi et al., and SAFELearn reveals several advantages. The proposed scheme maintains an efficient computation complexity of $O(m \log m)$ using Fast Fourier Transform (FFT), which is competitive with the other schemes. Additionally, the

communication complexity is efficient, with an upper bound of $O(n \log m)$ for aggregation, which is lower compared to some existing schemes like FastSecAgg. The proposed scheme also requires fewer rounds of communication per training iteration, enhancing the overall speed and reducing latency. Table 8 illustrates the relative performance of the proposed work with other existing schemes reported in the literature.

Limitations and future scope

In the future, several hardware-related directions could significantly enhance the performance and scalability of our proposed method. One important avenue is hardware acceleration, such as utilizing GPUs, TPUs, or specialized cryptographic processors to speed up the computation-intensive operations in FL and cryptographic schemes like NTRU. Additionally, integrating edge computing can help by decentralizing computation and reducing latency, especially for real-time applications. The use of FPGA-based designs could further optimize both cryptographic and FL tasks, offering a balance between flexibility and performance. Another promising direction is the development of dedicated secure hardware, such as Trusted Execution Environments (TEEs), to protect sensitive data during FL and reduce system vulnerability to hardware attacks. As quantum computing advances, quantum-resistant hardware solutions will be essential to maintain security, particularly with post-quantum cryptographic algorithms like NTRU. Moreover, hardware-efficient communication systems and power-efficient designs tailored for IoT and mobile devices will be crucial to ensure the scalability and sustainability of FL in resource-constrained environments. Hardware-accelerated blockchain solutions could speed up the verification and storage of encrypted gradients, optimizing overall system performance and data integrity. These hardware-related improvements could facilitate the widespread adoption of secure FL in various real-world applications.

Conclusion

This research presents an innovative blockchain and lattice-based cryptography approach utilizing NTRU to safeguard gradient communication between the server and client. Leveraging NTRU offers computational speed advantages over ECDH schemes, making it suitable for resource-constrained devices. Additionally, the communication overhead is significantly minimized. Furthermore, the scheme eliminates the need for third-party storage, as the gradients are securely stored in the blockchain network, impervious to alterations, and subject to regular verification. Moreover, the decentralized nature of blockchain storage ensures data immutability and integrity, enhancing overall system trustworthiness.

The proposed decentralized and secure FL system holds significant potential for applications across diverse domains, including healthcare, finance, and IoT, where data privacy and integrity are paramount. Future research could explore adaptive encryption strategies and further optimization for large-scale deployments. Ultimately, our method lays a robust foundation for the next generation of secure and efficient FL frameworks, empowering organizations to harness collaborative AI while maintaining stringent security standards.

Funding This research did not receive any specific funding.

Data availability MNIST Dataset: https://keras.io/api/datasets/mnist/.

Declarations

Conflict of interest The authors declare no conflict of interest.

Ethical approval Not applicable.

Informed consent Not applicable.

References

Ajtai, M.: Generating hard instances of lattice problems. In: Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing, pp. 99–108 (1996)

Antal, C., Cioara, T., Anghel, I., Antal, M., Salomie, I.: Distributed ledger technology review and decentralized applications development guidelines. Future Internet **13**(3), 62 (2021a)

Antal, C., Cioara, T., Antal, M., Anghel, I.: Blockchain platform for COVID-19 vaccine supply management. IEEE Open J. Comput. Soc. **2**, 164–178 (2021b)

Bhagoji, A.N., Chakraborty, S., Mittal, P., Calo, S.: Analyzing federated learning through an adversarial lens. In: International Conference on Machine Learning, pp. 634–643. PMLR (2019)

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

Chaudhary, R., Aujla, G.S., Kumar, N., Zeadally, S.: Lattice-based public key cryptosystem for internet of things environment: challenges and solutions. IEEE Internet Things J. **6**(3), 4897–4909 (2018)

Choi, B., Sohn, J.-Y., Han, D.-J., Moon, J.: Communication-computation efficient secure aggregation for federated learning. arXiv preprint arXiv:2012.05433 (2020)

Dai, H.-N., Imran, M., Haider, N.: Blockchain-enabled internet of medical things to combat COVID-19. IEEE Internet Things Mag. **3**(3), 52–57 (2020)

Deng, L.: The mnist database of handwritten digit images for machine learning research. IEEE Signal Process. Mag. **29**(6), 141–142 (2012)

Dong, Y., Chen, X., Shen, L., Wang, D.: Eastfly: efficient and secure ternary federated learning. Comput. Secur. **94**, 101824 (2020)

Feng, L., Zhao, Y., Guo, S., Qiu, X., Li, W., Yu, P.: Bafl: a blockchain-based asynchronous federated learning framework. IEEE Trans. Comput. **71**(5), 1092–1103 (2021)

Fereidooni, H., Marchal, S., Miettinen, M., Mirhoseini, A., Möllering, H., Nguyen, T.D., Rieger, P., Sadeghi, A.-R., Schneider, T., Yalame, H., et al.: Safelearn: secure aggregation for private federated learning. In: 2021 IEEE Security and Privacy Workshops (SPW), pp. 56–62. IEEE (2021)

Forouzan, B.A.: Data Communications and Networking. Huga Media (2007)

Gouert, C., Mouris, D., Tsoutsos, N.: SoK: New insights into fully homomorphic encryption libraries via standardized benchmarks. In: Proceedings on Privacy Enhancing Technologies (2023)

Guleria, A., Harshan, J., Prasad, R., Bharath, B.: On homomorphic encryption based strategies for class imbalance in federated learning. arXiv preprint arXiv:2410.21192 (2024)

Han, G., Zhang, T., Zhang, Y., Xu, G., Sun, J., Cao, J.: Verifiable and privacy preserving federated learning without fully trusted centers. J. Ambient Intell. Humaniz. Comput. 1–11 (2022)

Hoffstein, J., Pipher, J., Silverman, J.H.: Ntru: A ring-based public key cryptosystem. In: Algorithmic Number Theory: Third International Symposiun, ANTS-III Portland, Oregon, USA, June 21–25, 1998 Proceedings, pp. 267–288. Springer, Cham (2006)

Jiang, T., Chen, X., Ma, J.: Public integrity auditing for shared dynamic cloud data with group user revocation. IEEE Trans. Comput. **65**(8), 2363–2373 (2015)

Ju, Y., Yang, M., Chakraborty, C., Liu, L., Pei, Q., Xiao, M., Yu, K.: Reliability-security tradeoff analysis in mmWave ad hoc-based cps. ACM Trans. Sens. Netw. **20**(2), 1–23 (2024a)

Ju, Y., Gao, Z., Wang, H., Liu, L., Pei, Q., Dong, M., Mumtaz, S., Leung, V.C.: Energy-efficient cooperative secure communications in mmWave vehicular networks using deep recurrent reinforcement learning. IEEE Trans. Intell. Transp. Syst. (2024b)

Kadhe, S., Rajaraman, N., Koyluoglu, O.O., Ramchandran, K.: Fastsecagg: scalable secure aggregation for privacy-preserving federated learning. arXiv preprint arXiv:2009.11248 (2020)

Lei, X., Liao, X.: NTRU-KE: a lattice-based public key exchange protocol. Cryptology ePrint Archive (2013)

Liu, Z., Guo, J., Yang, W., Fan, J., Lam, K.-Y., Zhao, J.: Privacy-preserving aggregation in federated learning: a survey. IEEE Trans. Big Data (2022)

McMahan, B., Moore, E., Ramage, D., Hampson, S., Arcas, B.A.: Communication-efficient learning of deep networks from decentralized data. In: Artificial Intelligence and Statistics, pp. 1273–1282. PMLR (2017)

Micciancio, D., Regev, O.: Lattice-based cryptography. In: Post-Quantum Cryptography, pp. 147–191 (2009)

Mohandas, D.R., Veena, D.S., Kirubasri, G., Mary, I.T.B., Udayakumar, D.R.: Federated learning with homomorphic encryption for ensuring privacy in medical data. Indian J. Inf. Sources Serv. **14**(2), 17–23 (2024)

Ng, K.L., Chen, Z., Liu, Z., Yu, H., Liu, Y., Yang, Q.: A multi-player game for studying federated learning incentive schemes. In: Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence, pp. 5279–5281 (2021)

Ou, W., Zeng, J., Guo, Z., Yan, W., Liu, D., Fuentes, S.: A homomorphic-encryption-based vertical federated learning scheme for rick management. Comput. Sci. Inf. Syst. **17**(3), 819–834 (2020)

Pan, Y., Chao, Z., He, W., Jing, Y., Hongjia, L., Liming, W.: FedSHE: privacy preserving and efficient federated learning with adaptive segmented CKKS homomorphic encryption. Cybersecurity **7**(1), 40 (2024)

Puttagunta, M., Ravi, S.: Medical image analysis based on deep learning approach. Multimed. Tools Appl. **80**, 24365–24398 (2021)

Ulhaq, A., Burmeister, O.: COVID-19 imaging data privacy by federated learning design: a theoretical framework. arXiv preprint arXiv:2010.06177 (2020)

Xiong, J., Zhu, H.: PrivMaskFL: A private masking approach for heterogeneous federated learning in IoT. Comput. Commun. **214**, 100–112 (2024)

Xu, P., Hu, M., Chen, T., Wang, W., Jin, H.: LaF: Lattice-based and communication-efficient federated learning. IEEE Trans. Inf. Forensics Secur. **17**, 2483–2496 (2022)

Zhang, J., Chen, B., Yu, S., Deng, H.: Pefl: A privacy-enhanced federated learning scheme for big data analytics. In: 2019 IEEE Global Communications Conference (GLOBECOM), pp. 1–6. IEEE

Zhang, C., Li, S., Xia, J., Wang, W., Yan, F., Liu, Y.: {BatchCrypt}: Efficient homomorphic encryption for {Cross-Silo} federated learning. In: 2020 USENIX Annual Technical Conference (USENIX ATC 20), pp. 493–506 (2020)

Zhu, H., Goh, R.S.M., Ng, W.-K.: Privacy-preserving weighted federated learning within the secret sharing framework. IEEE Access **8**, 198275–198284 (2020)