Security and Verifiability in Federated Learning: A Zero-Knowledge Reputation-based Blockchain Framework

Swetha Ghanta, Ashok Kumar Pradhan, Prasanthi Boyapati, Sujit Biswas, Saraju P Mohanty

Abstract—Federated Learning (FL) enables collaborative training without centralizing sensitive data but faces challenges, including client authenticity, verifiable training participation, and secure aggregation. To overcome these challenges, we propose a novel framework, Zero-Knowledge Reputation-aware Blockchain Federated Learning (ZK-RBFL), which integrates blockchain, FL, Homomorphic Encryption (HE), and zero-knowledge proofs (ZKP). In the proposed ZK-RBFL framework, initially the clients undergo lightweight token-based authentication and then generate ZKP to provide cryptographic evidence of honest local training participation and reported inference accuracy before contributing their model updates. The model updates are encrypted using the CKKS HE mechanism to prevent any potential model inversion attacks. These encrypted model updates are stored on IPFS, with their corresponding CIDs recorded on the blockchain to ensure immutability. Further, ZK-RBFL enables mutual client verification of ZKPs to reduce server bottlenecks and enhance accountability. To ensure fairness and robustness in a distributed environment, we introduce a democratic blockchain consensus mechanism named Proof of Reputation-Weighted Voting (PoRWV) for block acceptance. Once consensus is reached, the encrypted model updates are aggregated using reputation-weighted averaging. We demonstrate the effectiveness of ZK-RBFL for brain tumor classification using a ZKP-compatible LeNet model for proof generation. Despite model simplicity, the global model achieves 94.22% accuracy. In addition, experiments with malicious clients and formal Scyther security analysis demonstrate that ZK-RBFL ensures both security and performance.

Index Terms—Blockchain Federated Learning, Zero Knowledge Proof, Client Authentication, Homomorphic Encryption, Reputation.

1 Introduction

Artificial Intelligence (AI) is rapidly becoming an integral part of our daily lives [1]. Numerous companies, including Amazon, Netflix, and YouTube, have integrated AI into their operations to enhance user experiences through improved product, movie, and video recommendations [2], [3], [4]. The emergence of ChatGPT [5] has further demonstrated the potential of transformer models [6] in powering Large Language Models (LLMs) [7], enabling a wide range of tasks from recipe suggestions [8] and problem-solving [9] to coding assistance [10] and creative image generation [11], [12]. Despite the widespread adoption of AI across various domains, certain sectors, such as healthcare and finance, remain hesitant [13], [14], [15]. AI systems based on machine learning (ML) [16], [17] or deep learning (DL) [18], [19] are heavily dependent on the quality and quantity of training data [20]. However, in sensitive fields like healthcare, data privacy concerns and strict regulatory requirements (e.g., GDPR [21] and HIPAA [22]) prevent organizations from sharing data, hindering collaborative model development [23].

To address this, researchers have proposed FL, a paradigm that enables collaborative training of DL models without transferring the raw data from local hosts [24], [25], [26]. Instead of raw data, only model weights are shared with a central server, where they are aggregated using Federated Averaging (FedAvg) [27].

While promising, the real-world deployment of FL presents several critical challenges:

- Ensuring a secure and reliable framework where only authenticated and authorized clients can participate in the FL process [28], which requires fast authentication and secure storage of client details.
- Defending against model inversion attacks (MIA) [29] that attempt to reconstruct private data from shared model parameters. Differential privacy (DP) [30] and Secure Multiparty Computation (SMPC) [31] are proposed as solutions; however, these approaches result in performance degradation [32].
- Addressing the common assumption that all clients and the server are trustworthy [33]. This ignores the risks of curious, lazy, and malicious participants.
- Verifying that clients execute the required local training steps before submitting their model updates to the central server [34], however, introduces additional overhead.

FL is mostly adopted in IoT-based applications, where the number of devices is large, and the computational capacity at each device is limited [35]. This is referred to as the cross-device scenario. In contrast, FL that usually involves fewer entities with high computational capabilities within medical organisations [36] is referred to as the cross-silo scenario. In the cross-silo setting, a more sophisticated

- *S. Ghanta, AK Pradhan, and P. Boyapati are with the Department of Computer Science and Engineering, SRM University-AP, Amaravati, Andhra Pradesh, India.*
  Corresponding author E-mail: ashokkumar.p@srmap.edu.in
- *S. Biswas is with the Computer Science Department, City St. George's, University of London, London, United Kingdom.*
- *SP Mohanty is with the Department of Computer Science and Engineering, University of North Texas, Denton, United States.*

framework is required, where only authenticated organisations are allowed to participate in the FL process [37]. This is crucial to prevent rogue clients from joining and poisoning the global model. Moreover, client authentication should be a continuous process, not a one-time activity, as malicious attackers can compromise a client system at any round of FL. Therefore, before permitting participation in the training process until the end of the training, a lightweight authentication mechanism is required to validate each client [38].

During FL, when model weights are sent from clients to the server for aggregation, there is a risk that an attacker could perform an MIA [39] to extract sensitive data from the model weights alone [40]. This type of attack is also possible on the server side, particularly if the server is honest-but-curious, meaning it performs its duties correctly but attempts to extract private information from the received data. Standard encryption schemes such as Advanced Encryption Standard (AES) [41] or Elliptic Curve Cryptography (ECC) [42] can help prevent man-in-the-middle (MiTM) attacks, but they are not suitable for protecting data from a curious server. This is because the aggregation of model weights at the server requires decryption. In such cases, HE, particularly the CKKS (Cheon-Kim-Kim-Song) scheme, can be leveraged [43]. CKKS supports a limited number of approximate arithmetic operations, such as addition and multiplication, on real numbers. These operations are performed directly on encrypted data, thus eliminating the need for decryption and mitigating the risk of MIA on the server side. CKKS offers a better performance trade-off compared to DP and SMPC.

Using HE, a curious server can be mitigated; however, clients, which are often assumed to be genuine in most existing works, can also pose risks. In FL, the global model is continuously exchanged between the server and clients. In each round, this model is trained locally by the clients using their private data. However, there is a possibility that some clients may behave dishonestly, such as being lazy and sending back the unmodified global model without training [44], or maliciously injecting harmful updates to degrade the global model's performance [45].

Verifying whether clients are honestly participating in local training and sending authentic updates is not straightforward, as it would typically require access to either the actual training data or the decrypted model weights. However, in a zero-trust environment, where neither the clients nor the server is fully trusted, such verification becomes a significant challenge. To address this, we incorporate ZKPs [46], [47], which allow a prover to demonstrate the correctness of specific computations without revealing sensitive data. In our framework, ZKPs are used to provide cryptographic evidence of honest training participation by proving the correct execution of a prescribed training step and the validity of inference-based performance claims, without disclosing private data or plaintext model weights.

To tackle these challenges holistically, we propose a novel and robust framework, ZK-RBFL, that integrates ZKP with Blockchain-based FL. By incorporating ZKP, we ensure that client training participation can be verified and performance scores can be obtained without compromising model privacy. Additionally, Blockchain technology is utilized to store authenticated client details, providing immutability and enabling automated tracking of client participation and reputation. This mechanism also ensures non-repudiation, preventing clients from falsely denying or claiming submission of a model. Considering scalability, we introduce a novel consensus mechanism to replace traditional approaches such as Proof of Work (PoW) [48], which demands high computational resources, and Proof of Stake (PoS) [49], which often results in a "rich-get-richer" problem. Our proposed Proof of Reputation-Weighted Voting (PoRWV) is a hybrid consensus mechanism that allocates voting power based on the reputation scores of participating nodes. Unlike centralized schemes such as Proof of Authority (PoA) or basic Proof of Reputation (PoR) [50], PoRWV promotes a democratic and trustworthy consensus process by enabling inclusive participation across all the nodes while maintaining efficiency and fairness.

1.1 Contributions

We present a novel ZK-RBFL framework that provides verifiable FL with ZKP guarantees for both training participation and inference accuracy, under a novel blockchain consensus (i.e., PoRWV) that ensures fairness and robustness even under malicious clients. The integration of CKKS HE enables secure aggregation over encrypted model updates, reducing the server's ability to directly inspect model weights and perform model inversion attacks. The key contributions of the proposed ZK-RBFL framework are summarized as follows:

a) We address the challenge of continuous client authentication in dynamic FL environments by integrating a lightweight token-based mechanism that enables fast admission control without introducing cryptographic overhead during training rounds.

b) We design a ZKP-based mechanism to counter false contributions and lazy clients, which allows clients to prove honest participation in training and correctness of the reported inference accuracy without disclosing model weights or data.

c) We propose a novel PoRWV mechanism that combines reputation scores with weighted voting-based consensus to mitigate the influence of false clients during block approval.

d) We introduce a reputation-aware weighted aggregation scheme with dynamic performance management to bound the impact of underperforming and malicious clients while preserving convergence.

1.2 Organisation

The remainder of this article is organised as follows. Section 2 provides a comprehensive review of FL, ZKPs, and blockchain-integrated FL frameworks, highlighting contemporary challenges and existing research gaps. Section 3 presents the threat model, outlining the potential security and privacy threats. Section 4 presents the proposed ZK-RBFL framework, which establishes a secure and trustworthy FL ecosystem. Section 5 describes the experimental setup, discusses the results and performance evaluation, and analyzes the key findings. Finally, Section 6 summarises the overall contributions and outlines potential directions for future research.

2 Related Work

Several studies have explored FL enhanced with blockchain, HE, and ZKPs to improve privacy, security, and fairness. However, core challenges such as verifiable client training participation and performance reporting, client reputation management, dynamic reputation-aware aggregation, FL-specific consensus, and scalable continuous client authentication remain largely unaddressed, particularly in cross-silo settings. We categorize and discuss the most relevant works based on their focused areas.

2.1 Federated Learning

As a decentralized machine learning approach, FL enables collaborative training of a global model by aggregating locally trained models from distributed participants. Instead of sharing raw data, each participant transmits only model updates to the central server for aggregation, thereby preserving data privacy and compliance with security regulations [51].

To handle the communication-related issues, Youqi et al. [52] proposed Bandit Gradient Estimation-aware FL (BGEFL) that estimates participants' gradients with limited bandit feedback. A meta-computing-driven vertical FL (VFL) based approach was proposed in [53], considering a variance-reduced gradient estimator for fast convergence. Although their work achieved high performance and reduced communication complexity, they did not consider possible attack scenarios and verifiable local model execution, and claimed accuracy.

A reputation-aware hierarchical aggregation framework, FedRaHa, is proposed in [54]. To reduce communication overhead, FedRaHa selects only clients with sufficient computational resources for training in each round. Similarly, a genetic algorithm-based client selection approach was proposed in [55]. But it incurs significant optimization overhead, leading to increased training time and reduced scalability. While this client selection strategy is effective in cross-device FL, where the number of participating devices is very large, it is less suitable for cross-silo scenarios in which each client represents a medical organization and the total number of participants is limited. Excluding clients in such settings can result in the loss of institution-specific medical data and introduce bias.

2.2 Federated Learning with Homomorphic Encryption

Different approaches, such as DP [56], SMPC [57], and HE [58], can be considered to prevent MIAs on the server side. The idea of DP is to carefully add noise to the data to protect it from malicious attackers [59], [60], [61], but the noise could degrade the performance of the aggregated model [62]. SMPC enables multiple parties to jointly compute a global model without revealing their local data by performing computations on secret-shared values [63], [64], [65]. While SMPC provides strong privacy guarantees in FL by distributing trust among multiple parties, it typically requires intensive communication among participants for each computation step, leading to significant communication overhead [66].

To mitigate this limitation, HE schemes can be leveraged, as they allow the aggregation of encrypted model updates directly on the server side without requiring decryption, thereby reducing communication costs and simplifying coordination among clients [67]. The authors in [68] used a partial ElGamal HE scheme, a multiplicative homomorphic scheme adjusted and converted to an additive homomorphic scheme to reduce the computational overhead. The partial HE schemes support either addition or multiplication operations, but not both. Somewhat HE (SHE) schemes, on the other hand, support both the operations, but in a limited number. Several works integrated SHE schemes such as fixed arithmetic-based Brakerski-Gentry-Vaikuntanathan (BGV) [69], [70], and Brakerski/Fan-Vercauteren (BFV) [71], [72], and approximate arithmetic-based CKKS [43], [73] with FL. Truhn et al. [74] proposed a privacy-preserving FL framework for cancer image analysis, employing a SHE scheme for secure aggregation on the server. Similarly, the authors in [75] designed an FL framework for Alzheimer's detection using HE to secure the aggregation process. In [76], the authors proposed the FedARCH framework integrating CKKS HE, in which clients mutually validate one another and share their validation accuracies. Based on these accuracies, reputation scores are assigned to the clients, which are then used for reputation-weighted aggregation. Although effective, these works do not guarantee verifiable training participation or consider misreporting by malicious clients, and they also lack mechanisms for authenticating participating clients and immutable model update storage.

2.3 Blockchain-integrated Federated Learning (BFL)

Blockchain is a decentralized and immutable ledger that provides secure, transparent, and tamper-proof data management through cryptographic hash functions and smart contracts [94], [95]. Its consensus mechanism enables multi-party verification and agreement, making it highly suitable for enhancing authentication and trust within the FL ecosystem [96]. A custom blockchain approach for FL-based brain tumor classification leveraging SHA-256 was considered in [77] to ensure immutability. A Nash bargaining theory incentive mechanism with a probabilistic greedy-based client selection approach was proposed in [78] to motivate clients to provide quality model updates. Although it is efficient w.r.t communication cost to select a few clients among all the participating FL clients, this could lead to bias towards certain clients. In cross-silo medical scenarios, it could also lead to missing out on valuable patient information. Similarly, a BFL framework with a Shapley-based incentive mechanism was introduced in [79] to reward clients proportionally to their contributions. The authors of [80] developed a BFL architecture for the NIH Chest X-ray dataset using PoW consensus. Likewise, a decentralized BFL approach for medical image analysis incorporating PoW consensus was proposed in [81]. However, these existing BFL works store model weights directly on the blockchain, leading to significant scalability and latency concerns.

Although BFL was used for traceability and integrity in [82], [83], and [97], the presence of an untrusted server was not sufficiently addressed. A secure Trusted Execution Environment (TEE)-based system was introduced in [84], where model weights were encrypted with AES before aggregation

TABLE 1: Comparison of ZK-RBFL with existing approaches

Category	Reference	Year	Approach	Addressed Issues	Limitations
FL	Ghanta et al. [51]	2025	Standard FL for medical imaging	Protects data privacy via local training	Vulnerable to model inversion and poisoning attacks, lacks verifiable training
	Youqi et al. [52]	2025	Bandit Gradient Estimation-aware FL (BGEFL)	Reduces communication complexity using limited bandit feedback	Ignores server-side security threats and unverifiable client training
	Youqi et al. [53]	2025	Meta-computing-driven Vertical FL (VFL)	Employs variance-reduced gradient estimator for fast convergence under heterogeneous data	High performance but lacks attack resistance and cryptographic verification of client-side training participation
	Panigrahi et al. [54]	2023	Reputation-aware Hierarchical Aggregation (FedRaHa)	Hierarchical edge aggregation to minimize communication overhead and resource-based client selection	Suitable for cross-device FL but ineffective in cross-silo scenarios
	Kang et al. [55]	2023	Genetic Algorithm-based Client Selection	Enhances fairness and optimizes participant contribution in FL	Significantly increases training time, reducing scalability
FL + HE	Zhang et al. [68]	2022	Partial ElGamal-based Homomorphic FL	Reduces computational overhead by converting multiplicative HE to additive form	Supports only one operation type (addition or multiplication), limiting flexibility
	Truhn et al. [74]	2024	Privacy-preserving FL for Cancer Imaging	Employs SHE for secure aggregation of encrypted updates on server	Protects data privacy but lacks client authentication and verifiable training
	Veda et al. [75]	2025	HE-secured FL Framework for Alzheimer's Detection	Uses HE to protect model aggregation in medical FL	No validation of client honesty or training authenticity
	Ghanta et al. [76]	2025	FedARCH Framework with CKKS-based HE	Incorporates mutual validation and reputation-weighted aggregation among clients	Lacks mechanisms for verifying honest training participation and preventing misreporting of performance metrics
BFL	Rajit et al. [77]	2024	Custom Blockchain for FL-based Brain Tumor Classification	Uses SHA-256 to ensure data immutability and model traceability	High computational cost and limited scalability for large-scale FL
	Youqi et al. [78]	2024	Nash Bargaining-based Incentive Mechanism	Motivates clients to provide high-quality updates via one-to-many bargaining and probabilistic selection	Introduces client selection bias; may omit valuable information in cross-silo setups
	Liu et al. [79]	2022	Shapley-based Incentive Mechanism in BFL	Rewards clients based on contribution fairness using consortium blockchain	Relies on centralized coordination; lacks proof of clients' honest training participation
	Myrzashova et al. [80]	2024	PoW-based BFL for NIH Chest X-ray Dataset	Ensures model resilience against multiple cyberattacks using decentralized validation	PoW introduces significant latency and energy overhead
	Bhatia et al. [81]	2023	Decentralized BFL for Medical Image Analysis	Employs PoW consensus for distributed model aggregation	Storing model weights on-chain increases latency and storage cost
	Qu et al. [82]	2020	Blockchained FL for Traceability and Integrity	Provides transparent model updates and traceable transactions	Untrusted server and model inversion threats not fully addressed
	Lu et al. [83]	2020	Communication-efficient BFL Architecture	Reduces communication cost while maintaining integrity via blockchain	Does not ensure client authenticity or protect against adversarial updates
	Kalapaaking et al. [84]	2022	TEE-based BFL with Encrypted Model Aggregation	Secures model updates via AES encryption and trusted enclave aggregation	Lacks continuous client authentication and verifiable training participation
BFL + Authentication	Fan et al. [85]	2023	Lightweight Anonymous Authentication for BFL	Introduces batch verification to reduce latency in participant identity verification	Lacks validation of local model training and reported updates
	Fan et al. [86]	2023	LPBFL: Consortium Blockchain with Paillier HE and Lightweight Signatures	Combines blockchain, HE, and digital signatures; uses reputation-based client selection	Focuses on communication efficiency but lacks verifiable training authenticity
	Mahato et al. [87]	2024	PPVFL: Privacy-preserving BFL Framework with HE and ECDSA	Integrates HE and Byzantine fault tolerance for secure authenticated communication	No mechanism to verify correctness of local updates or client-side computation integrity
	Ji et al. [88]	2023	LAFED: Lightweight Authentication for BFL using ZKP	Employs ZKPs for client identity verification	Lacks mechanisms to verify local training authenticity and reported metric correctness
FL+ZKP	Chakraborty et al. [89]	2024	Decentralized Leader Election using ZKP and PoR	Employs secure shuffling and proof-of-retrievability with theoretical ZKP for verifiable participation	Limited to conceptual validation; lacks practical implementation and training verification
	Xing et al. [90]	2023	ZKP-FL using Groth16 for Computation and Aggregation Verification	Verifies local computation and aggregation correctness across multiple ML tasks	Groth16 requires new trusted setup for each circuit; unsuitable for dynamic FL environments
	Petrosino et al. [91]	2025	BFL-ZKP for Blood Glucose Prediction using LSTM Models	Integrates DIDs and ZKP for identity privacy and inference verifiability	Verification limited to inference phase; lacks continuous authentication and dynamic reputation tracking
	Zhang et al. [92]	2023	ZKVM with BGV for ML tasks on the IRIS dataset	Provides execution integrity proofs at learning nodes using ZKVM and HE	Focuses mainly on ML tasks, DL image-based tasks needing complex circuit handling are not addressed
	Tang et al. [93]	2025	zkFL framework integrating ZKPs (zk-SNARKs) with FL	Handles Byzantine and malicious servers; ensures verifiable aggregation; maintains gradient privacy	Proof of client-side training is not ensured, allowing potential submission of outdated or fabricated updates
Proposed: ZK-RBFL			ZKP-compatible BFL with reputation-weighted aggregation and an FL-centric PoRWV consensus for block approval	Enables verifiable client participation and accuracy claims through cryptographic proofs while maintaining privacy and scalability in FL	-

within the TEE. Confidentiality, decentralization, and transparency were emphasized, but client authentication was not incorporated. Most existing BFL works focus on storing models and secure aggregation, but it is also crucial to ensure that only authenticated clients can participate in the FL process, thereby preventing random or malicious clients from joining. This authentication should be continuous, rather than a one-time process, to prevent malicious clients from participating throughout the FL process.

2.4 BFL with Authentication Mechanism

Several authentication approaches have been proposed to ensure only authenticated clients participate in FL training [28], [98], [99]. These authentication methods have evolved from identity-based signcryption [100], [101] to hignchryption [102], [103], where hidden-identity-based signcryption techniques are employed to provide both digital signatures and identity concealment. A lightweight Identity-based Cryptography (IBC) authentication mechanism was proposed in [104] based on the digital signature algorithm for energy demand prediction. A lightweight anonymous authentication scheme with batch verification was introduced for BFL systems in [85] to reduce the latency involved in identity verification. In [86], the authors proposed LPBFL, which combined consortium blockchain with Paillier HE, lightweight digital signatures, and batch verification. In [87], the authors proposed a privacy-preserving FL framework (PPVFL) by integrating BFL with HE. Byzantine fault tolerance and ECDSA signatures were used to secure client communications. In [88], the authors proposed LAFED, a lightweight authentication approach for BFL that employs ZKPs for client identity verification. While these approaches address client authentication and selection, and protect data integrity during transmission, they lack cryptographic guarantees for verifiable training participation and the correctness of reported metrics.

2.5 Federated Learning with Zero Knowledge Proof

ZKP enables a prover (e.g., client) to prove the validity of a statement to a verifier (e.g., the server) without revealing any additional information [105]. A leader election protocol using secure shuffling and PoR, incorporating ZKP in a theoretical form, was proposed in [89] to ensure verifiability. A BFL-ZKP integrated system was proposed for blood glucose level prediction using LSTM models in [91]. The framework incorporated Decentralized Identifiers (DIDs) and ZKP for identity privacy and inference verifiability. The authors used Groth16, which requires a new trusted setup for every distinct circuit. Similarly, a ZKP-based FL (ZKP-FL) was proposed in [90] for computation and aggregation verification using the Groth16-based ZKP scheme. Two different machine learning tasks, such as house price prediction and iris classification, are considered. However, in FL, the circuit changes frequently, making Groth16 less suitable. Instead, a universal setup is required. In addition, although these studies explore ZKPs for verifiability, the implementation is limited to inference. They do not support continuous authentication or dynamic client reputation tracking.

In [92], the authors provided execution integrity proofs at the learning nodes using a zero-knowledge virtual machine (ZKVM) along with a BGV HE scheme for an ML-based task on the IRIS dataset. However, since BGV operates on integers rather than real numbers, this limitation could affect computational accuracy. Furthermore, existing works primarily focus on simple ML tasks [106], [107], [108], whereas DL image-based tasks require more complex circuit handling [109], [110], which is not addressed in these studies. Tang et al. [93] proposed zkFL, a Byzantine-robust FL framework for image classification that integrates ZKPs to enable verifiable aggregation under malicious servers while preserving gradient privacy. However, it lacks a mechanism to verify client-side training participation, which may allow the submission of outdated or fabricated model updates.

In summary, existing works (e.g., [74], [86], [89]) address important aspects of security and privacy in FL; however, they largely overlook critical FL-specific requirements. In particular, current approaches lack mechanisms for verifiable training participation, lightweight and continuous client authentication, and consensus designs tailored to cross-silo FL settings. As a result, ensuring accountability without compromising scalability or privacy remains an open challenge. To address these limitations, ZK-RBFL is proposed as a unified framework that enables lightweight token-based authentication, ZKPs-based verifiability of training participation and claimed accuracy, FL-centric blockchain consensus, and reputation-aware aggregation. A comparison between existing work and the proposed ZK-RBFL framework is presented in Table 1. By combining accountability with privacy-preserving guarantees, ZK-RBFL provides a practical and secure foundation for sensitive domains such as healthcare.

3 System and Threat Model

Assume an FL system with one central server S and a set of clients $\mathcal{C} = \{C_1, \ldots, C_N\}$, where each client C_i holds a private dataset $\mathrm{D_i}$. At training round r, the server broadcasts model parameters GM^r, clients compute local updates LM_i^r on their data, and return them to the server. The server aggregates these updates into a new global model $GM^{r+1} = \mathsf{Agg}(\{LM_i^r\})$. In our *zero-trust setting*, neither the server nor the clients are assumed fully honest: both may behave in ways that compromise privacy, integrity, or availability. On the **server side**, the adversary is modeled as *honest-but-curious*: while following the protocol correctly, the server may attempt to infer sensitive client data from local updates. We define its inference advantage in terms of membership inference or model inversion, for example $\mathrm{Adv}^{\mathrm{mem}}_{S,i,r}(z)$, the probability difference that the server correctly decides whether a record z belongs to client i's dataset, while $\mathrm{Adv}^{\mathrm{inv}}_{S,i,r}(z)$ quantifies the server's ability to reconstruct sensitive features of z from client i's data. A secure system aims to limit information leakage from exchanged messages. We seek to reduce such leakage under the adopted cryptographic protections, while recognizing that some residual leakage may arise from protocol execution and system-level interactions.

On the **client side**, multiple adversarial behaviors are possible. A *lazy client* may send random or stale updates

$\tilde{LM}_i^r$, introducing bias. An *underperforming client* may increase gradient variance by a factor $\kappa > 1$, degrading convergence. More severely, a *malicious client* may poison the model by submitting updates $m_i^t = LM_i^r + \delta_i^r$, where δ_i^t is chosen to maximize the global loss. *False participation threats* arise when a client claims contribution without a valid update, or later denies having sent it. This is prevented by binding each update to a digital signature

$$Sign_i^r = \mathsf{Sign}_{\mathsf{Priv\_key}_i}(SHA256(LM_i^r)),$$

ensuring accountability and non-repudiation.

Integrity threats arise when an adversary tampers with the transmitted model update LM_i^r during communication, either by modifying its content or by injecting corrupted data, leading the server to aggregate incorrect values. To prevent such manipulation, each update is bound to a cryptographic hash, $h_i^r = SHA256(LM_i^r)$, which serves as a compact fingerprint of the client's contribution. Upon reception, the server recomputes the hash and verifies consistency with the reported h_i^r, ensuring that the update remains unaltered in transit and preserving data integrity.

Additional adversarial strategies include *Sybil attacks*, where a single entity spawns multiple fake identities $\mathcal{F}$ to increase its weight in aggregation. Formally, its influence weight, $W(\mathcal{F}) = \sum_{j\in\mathcal{F}} w_j$, should be bounded to prevent disproportionate impact on the global model. In addition, *rogue clients* may attempt to participate without authorization. A secure system, therefore, requires that updates originating from unauthenticated or unauthorized clients be detected and rejected.

To counter these threats, the proposed ZK-RBFL framework, detailed in Section 4, enforces the following key guarantees. First, **accountability and admission control** ensure that only authenticated clients can participate, all client updates are cryptographically signed, and Sybil influence is effectively bounded. Second, **privacy** is maintained against a curious server by using the CKKS HE approach. CKKS primarily mitigates direct leakage from transmitted updates, thereby reducing the adversary's effective advantage in mounting direct membership inference and model inversion attacks. Third, **integrity** is ensured against tampering in transit by binding each client update LM_i^r to a cryptographic hash h_i^r. Upon receiving the update, the server verifies that it is consistent with the reported hash. Assuming a secure cryptographic hash, any modification of the update during transmission is detected with a higher probability, ensuring that the updates used for aggregation reflect the actual updates sent by the clients. Fourth, **reliability** is ensured in the presence of Byzantine clients through robust, reputation-based aggregation rules that limit the influence of malicious updates, even when some participating clients behave adversarially. Collectively, these guarantees establish a practical zero-trust federated learning environment in which both server- and client-side threats are systematically constrained.

4 Proposed ZK-RBFL Framework

The implementation of the proposed ZK-RBFL framework is divided into four key phases: initialisation and authentication, FL training and ZKP-based client verification, blockchain integration, and server-side aggregation.

TABLE 2: Notation Table for FL Training and Verification

Notation	Description
CA	Certificate Authority
sk_{CA}	signing key of CA
pk_{CA}	public key of CA
$\mathsf{rt_i}$	Refresh token of Client C_i
$\mathsf{at_i}$	Access token of Client C_i
R	Total no. of rounds
N	Total no. of clients
$\mathcal{S}$	Server
r	Round number
$\mathrm{D_i}$	Local training data of Client C_i
$\mathrm{V^r}$	Validation data at round r
$\tilde{V}_i^r$	A randomly sampled subset of $\mathrm{V^r}$
GM^r	Global model at round r
EGM^r	Encrypted Global model at round r
$\mathcal{C}$	Set of clients
$\mathcal{C}_{\text{legit}}$	Set of legitimate clients
C_i	i-th client
$ClientID_i$	Client ID of i-th client
LM_i^r	Local model trained by C_i at round r
ELM_i^r	Encrypted local model using CKKS
$HELM_i^r$	Hash of the encrypted local model
$Priv\_key_i$	Private key of client C_i
Pub_key_i	Public key of client C_i
$Prov\_key_i$	Proving key of client C_i
$Verify\_key_i$	Verification key of client C_i
$Sign_i^r$	Digital signature on $HELM_i^r$
ZKP_i^r	Zero-knowledge proof generated by C_i
Acc_i^r	Validation accuracy claimed by C_i
TS_i^r	Timestamp from client C_i at round r
CID_i^r	IPFS content identifier for ELM_i^r
$TXN\_Hash_i^r$	Transaction hash of C_i stored in mempool at r
$Merkle\_hash^r$	Root of Merkle tree of approved transactions
VT	Voting power Table
vw	Vote weight
V_{yes}^r	No. of yes votes at round r
V_{no}^r	No. of no votes at round r

The overall architecture of the framework is illustrated in Figure 1, and Table 2 lists the notations along with their corresponding descriptions used in the proposed ZK-RBFL framework.

4.1 Initialization and Authentication Phase

First, each participating client undergoes authentication to prevent unauthorized or rogue clients from joining the FL process. In real-world scenarios, hospitals or medical organizations have medical license certificates. Similarly, in our ZK-RBFL framework, a trusted Certificate Authority (CA) issues an X.509 certificate to each client, which serves as a digital equivalent of a medical license and is valid for the entire FL session. Clients register for participation in the FL process using this X.509 certificate. Only legitimate clients with valid certificates are allowed to participate in FL.

Let $\mathcal{C} = \{C_1, \ldots, C_N\}$ denote the set of participating clients and CA the trusted Certificate Authority. Each C_i represents a hospital or medical organization that must be authenticated before joining the FL protocol. A trusted certificate authority CA verifies each C_i and issues a unique X.509 certificate

$$\mathsf{cert}_i = (\mathsf{ClientID}_i, \mathsf{pk}_i, \mathsf{Sign}_{\mathsf{sk_{CA}}}(SHA256(\mathsf{ClientID}_i \parallel \mathsf{pk}_i))),$$

where $\mathsf{ClientID}_i$ is the client identity, pk_i is the client's public key, and $\mathsf{sk_{CA}}$ is the CA's signing key. During the FL protocol initialization, C_i presents its certificate, and authentication succeeds if $\mathsf{Verify}_{\mathsf{pk_{CA}}}(\mathsf{cert}_i) = 1$. A client without a valid certificate cannot join the FL process. This inherently mitigates Sybil attacks, as one entity cannot feasibly obtain multiple valid certificates.

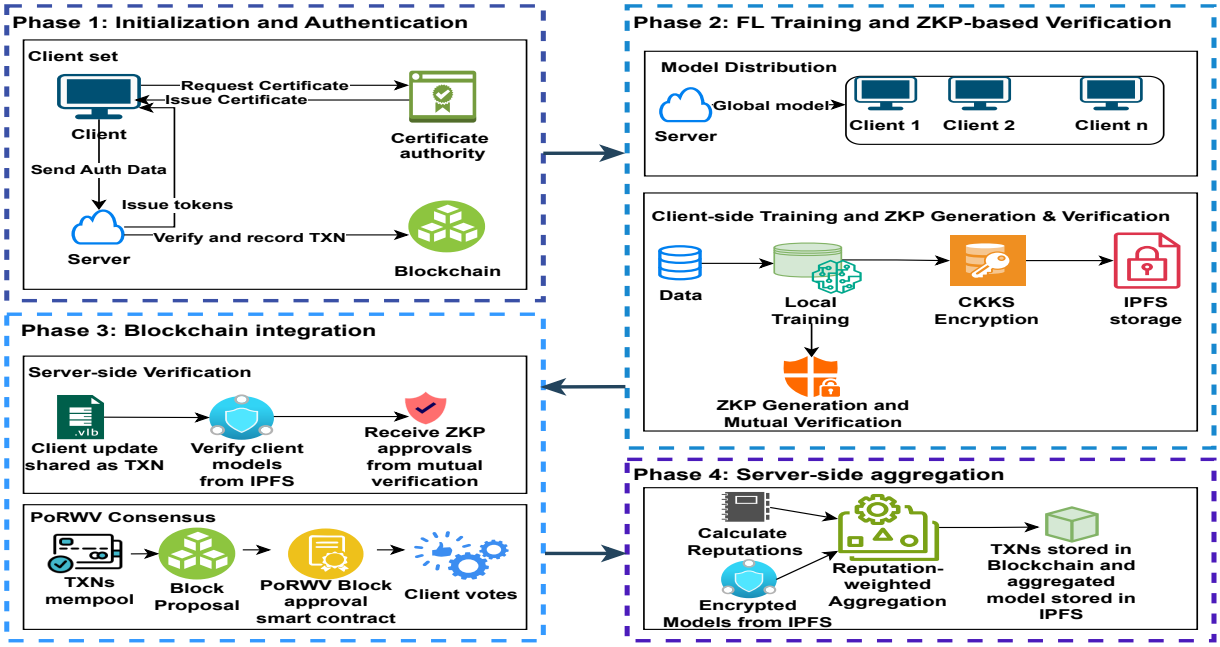

Fig. 1: Conceptual overview of ZK-RBFL framework

Algorithm 1 Multi-level Authentication in ZK-RBFL Framework

Input: $[\mathcal{C}, \mathsf{CA}, \Delta]$
Output: Set of legitimate clients $\mathcal{C}_{\text{legit}}$
Level I: Certificate-based Initialisation
for each $C_i \in \mathcal{C}$ **do**
  C_i submits identity ID_i and public key pk_i to CA
  CA issues $\mathsf{cert}_i = (\mathsf{ID}_i, \mathsf{pk}_i, \mathsf{Sign}_{\mathsf{sk}_{\mathsf{CA}}}(\mathrm{SHA256}(\mathsf{ID}_i \parallel \mathsf{pk}_i)))$
  Server verifies $\mathsf{Verify}_{\mathsf{pk}_{\mathsf{CA}}}(\mathsf{cert}_i) \stackrel{?}{=} 1$
  if verification succeeds **then**
    add C_i to $\mathcal{C}_{\text{legit}}$
    Issue refresh token $\mathsf{rt}_i = \mathrm{hex}(R_{rt}),\ R_{rt} \xleftarrow{\$} \{0,1\}^{256}$
  else
    Reject C_i
  end if
end for
Level II: Access Token Generation
for each FL round r **do**
  for each authenticated client $C_i \in \mathcal{C}_{\text{legit}}$ **do**
    C_i submits rt_i to server
    if rt_i is valid and not expired **then**
      Issue access token $\mathsf{at}_i^r = \mathrm{hex}(R_{at}),\ R_{at} \xleftarrow{\$} \{0,1\}^{128}$
      at_i^r valid for rounds $[r,\ r+\Delta]$
    else
      remove C_i from $\mathcal{C}_{\text{legit}}$
      C_i must re-register with certificate cert_i
    end if
  end for
end for
Level III: Continuous Verification
for each participation request from $C_i \in \mathcal{C}_{\text{legit}}$ in round r **do**
  if $\mathsf{Verify}(\mathsf{at}_i^r) = 1 \ \wedge\ r \le r_0 + \Delta$ **then**
    Accept C_i for round r
  else
    Reject request
    remove C_i from $\mathcal{C}_{\text{legit}}$
  end if
end for

For continuous verification of clients throughout the FL lifecycle, after initial certificate verification, ZK-RBFL employs lightweight multi-level token-based authentication using refresh and access tokens:

- **Refresh Token:** After certificate validation, the client C_i receives a refresh token rt_i from the server. The refresh token is long-lived relative to access tokens and allows C_i to periodically obtain new access tokens.

$$\mathsf{rt}_i = \mathrm{hex}(R_{rt}), \qquad R_{rt} \xleftarrow{\$} \{0,1\}^{256}.$$

- **Access Token:** At FL round r, client C_i presents a valid access token at_i^r, generated as

$$\mathsf{at}_i^r = \mathrm{hex}(R_{at}), \qquad R_{at} \xleftarrow{\$} \{0,1\}^{128}.$$

  The access token remains valid only for Δ consecutive rounds, i.e., tokens presented beyond this validity window are rejected. Expired tokens must be refreshed using rt_i.

In this way, the ZK-RBFL framework ensures:

1) **Uniqueness:** $\forall i \neq j$, $\mathsf{cert}_i \neq \mathsf{cert}_j$. Each client identity is unique and traceable.
2) **Non-repudiation:** Any participation by C_i in round r is linked to $(\mathsf{cert}_i, \mathsf{at}_i^r)$; thus, denial of authorship is infeasible with negligible probability.
3) **Freshness:** Access tokens are bounded by Δ rounds, limiting the window of compromise. Even if at_i^r leaks, adversarial use is confined to Δ rounds.

The proposed framework delegates continuous authentication to lightweight token validation, instead of repeated certificate verification at every round, thereby reducing overhead for clients. The multi-level authentication phase ensures that only legitimate, continuously verified clients participate in the FL process, while Sybil resistance, accountability, and freshness are cryptographically guaranteed. The detailed procedure is presented in Algorithm 1.

4.2 FL Training and ZKP-based Client Verification

Upon successful client authentication, the FL process is initiated by the server. A global model GM^0 is initialized and broadcast to all participating clients. Each client C_i receives the current global model GM^r at round r, trains it locally on its private data to produce a local model LM_i^r. To prove the honest participation of client-side training, C_i generates a zero-knowledge proof ZKP_i^r using the proving key $Prov\_key_i$, demonstrating that C_i participated in model training and that the reported validation accuracy Acc_i^r is valid.

$$ZKP_i^r = \mathsf{ZKPGen}(Prov\_key_i, LM_i^r, Acc_i^r) \tag{1}$$

Then, C_i encrypts LM_i^r using CKKS HE, resulting in ELM_i^r. The encrypted model ELM_i^r is then stored in IPFS to maintain a client-side record, resulting in a content identifier CID_i^r.

$$ELM_i^r = \mathsf{Enc_{CKKS}}(LM_i^r) \tag{2}$$

Next, each client computes a cryptographic hash of ELM_i^r to obtain $HELM_i^r$, ensuring model integrity. To bind this hash to the client, $HELM_i^r$ is digitally signed using the client's private key $Priv\_key_i$ via the Elliptic Curve Digital Signature Algorithm (ECDSA), producing the signature $Sign_i^r$.

$$HELM_i^r = \mathsf{SHA256}(ELM_i^r) \tag{3}$$

$$Sign_i^r = \mathsf{Sign}(Priv\_key_i, HELM_i^r) \tag{4}$$

Each client C_i sends the following tuple to the central server: $[ClientID_i, CID_i^r, ELM_i^r, Sign_i^r, Pub\_key_i, ZKP_i^r, Acc_i^r, TS_i^r, HELM_i^r, Verify\_key_i^r]$

At the server side, the server fetches the encrypted model ELM_i^r from IPFS using CID_i^r. Computes the hash and compares it with the received $HELM_i^r$ for integrity. If it is verified, the server stores the selected client information temporarily in the mempool as a transaction TXN_i^r along with a transaction hash $TXN\_Hash_i^r$.

$$TXN\_Hash_i^r = \mathsf{SHA256}(TXN_i^r) \tag{5}$$

Each transaction is structured in JSON format as:

```
[
  {
    "ClientId": "Client_03",
    "Signature": "16185194f18ef...",
    "Public_key": "VerifyingKey.from_string(b'..')",
    "ZKP": "proof.json",
    "Accuracy": 0.9422,
    "Timestamp": "1743501469.822508",
    "HELM_Hash": "8c17772fc937..."
  }
]
```

To ensure mutual verification, each client C_i retrieves information of the previous client C_{i-1} from the mempool. C_i verifies the signature $Sign_{i-1}^r$ with the previous client's public key Pub_key_{i-1} to confirm authenticity and non-repudiation, then validates ZKP_{i-1}^r using the verifying key $Verify\_key_{i-1}^r$. If both verifications are successful, C_i signs the $TXN\_Hash_{i-1}^r$ to indicate approval and sends it to the server.

$$\mathsf{Verify}\big(\mathsf{Pub\_key}_i,\ \mathsf{HELM}_i^r,\ \mathsf{Sign}_i^r\big) = \begin{cases} 1, & \text{if sign is valid} \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

$$\mathsf{ZKPVerify}(Verify\_key_i, ZKP_i^r) = 1 \tag{7}$$

After receiving all approvals, the server collects the set of transaction hashes TXN_Hash^r, computes the Merkle root $Merkle\_hash^r$, and proposes a new block containing all transactions and the Merkle root. The detailed procedure is outlined in Algorithm 2.

$$TXN\_Hash^r = \{TXN\_Hash_i^r\}_{i=1}^N \tag{8}$$

$$Merkle\_hash^r = \mathsf{MerkleRoot}(TXN\_Hash^r) \tag{9}$$

Algorithm 2 FL Training and ZKP-based Client Verification

Input: $[GM_i^r, \mathrm{D_i}, \mathrm{V^r}, Priv\_key_i, Pub\_key_i]$
Output:$[LM_i^r,\ ELM_i^r,\ HELM_i^r,\ Sign_i^r,\ CID_i^r,\ ZKP_i^r,\ Acc_i^r,\ Verify\_key_i^r, Merkle\_hash^r]$
Server $\mathcal{S}$ initializes GM^0 and sends to all authenticated clients $C_i \in \mathcal{C}_{\text{legit}}$
for each round r **do**
  for each client C_i **do**
    Train $GM^r \rightarrow LM_i^r$
    Evaluate model accuracy Acc_i^r on randomly sampled subset $\tilde{V}_i^r$ of round-specific validation set $\mathrm{V^r}$
    Generate ZKP_i^r using $Prov\_key_i$
    Encrypt LM_i^r using CKKS: ELM_i^r
    Store ELM_i^r in IPFS $\rightarrow CID_i^r$
    Hash $ELM_i^r \rightarrow HELM_i^r$
    Sign $HELM_i^r$ using $Priv\_key_i \rightarrow Sign_i^r$
    Send $[ClientID_i, CID_i^r, ELM_i^r, Sign_i^r, ZKP_i^r, Acc_i^r, TS_i^r, HELM_i^r, Verify\_key_i^r] \rightarrow \mathcal{S}$
  end for
  for each received txn at $\mathcal{S}$ **do**
    Retrieve ELM_i^r from IPFS using CID_i^r
    Verify hash and signature $Sign_i^r$
    Add to mempool as txn with TXN_Hash if valid
  end for
  for each client C_i **do**
    Retrieve $[ZKP_{i-1}^r, Acc_{i-1}^r, TS_{i-1}^r, HELM_{i-1}^r, Verify\_key_{i-1}^r]$ from mempool
    Verify ZKP_{i-1}^r using $Verify\_key_{i-1}^r$
    if Valid **then**
      Sign and send $TXN\_Hash_{i-1}^r \rightarrow \mathcal{S}$
    end if
  end for
  $\mathcal{S}$ aggregates all approved TXN_Hash^r
  Compute $Merkle\_hash^r$
  $\mathcal{S}$ proposes new block containing TXN_Hash^r and $Merkle\_hash^r$
end for

4.2.1 ZKP generation and verification

We employ ZKPs to ensure that clients in a zero-trust FL environment are honestly participating in local training and achieving the reported performance without exposing sensitive data or model weights. However, proving the entire training process of deep networks is currently infeasible. Hence, in our proposed work, the client provides cryptographic evidence of training participation by generating a ZKP attesting to the correct execution of one valid stochastic gradient descent (SGD) step on the model using local data. The SGD batch $\mathrm{B_i}$ is selected via a round-dependent challenge, preventing offline precomputation or replay of valid proofs. After local training, the client runs inference on a randomly sampled subset of the round-specific validation set to provide evidence of the claimed model accuracy.

We utilize the *ezkl* library [111], which transforms neural network computations into ZK-SNARK-compatible circuits

Algorithm 3 ZKP Generation and Verification with One-Step SGD Attestation

Input: $[GM^{r-1}, LM_i^r, \mathrm{B_i}, \mathrm{V^r}, Prov\_key_i, Verify\_key_i]$
Output: $[ZKP_i^r, Acc_i^r]$
Client C_i operations:
Receive global model GM^{r-1} from server $\mathcal{S}$
Select prescribed training batch $\mathrm{B_i}$ based on round challenge
Perform one valid SGD update on $\mathrm{B_i}$:

$$LM_i^r \leftarrow GM^{r-1} - \eta \nabla \mathcal{L}(GM^{r-1}, \mathrm{B_i})$$

Run inference using LM_i^r on randomly sampled subset $\tilde{V}_i^r \subset V^r$ and compute accuracy Acc_i^r
Define ZKP circuit encoding:
(i) correctness of one SGD update, and
(ii) inference and accuracy computation
Generate proof: $ZKP_i^r \leftarrow \mathtt{Prove}(\mathcal{C}, GM^{r-1}, LM_i^r, \mathrm{B_i}, \tilde{V}_i^r, Prov\_key_i)$
Send $[ZKP_i^r, Acc_i^r] \rightarrow \mathcal{S}$ and previous client C_{i-1}
Client C_{i-1} operations:
if $\mathtt{Verify}(ZKP_i^r, Verify\_key_i)$ is True **then**
    Accept model update and approve $TXN\_Hash_i^r$
else
    Reject update and flag C_i for potential misbehavior
end if

using the Halo2 proving system [112]. The detailed procedure is outlined in Algorithm 3 and the workflow is shown in Fig. 2. The key phases of the ZKP process are:

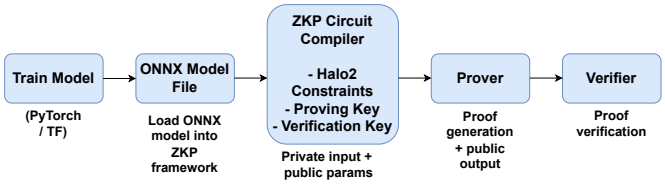

Fig. 2: Workflow of ZKP generation and verification

a) **Model Compilation:** In this phase, the trained neural network model is converted into an arithmetic circuit that can be executed inside a ZKP system. Here, the LeNet CNN model is exported to the ONNX format [113]. The *ezkl* library parses the ONNX model and compiles it into a Halo2 circuit. This circuit encodes the model's computation graph, such as convolutions, ReLU, and pooling, in constraint form. The resulting circuit representation can then be used to generate proofs for any compatible inputs. In the corresponding ZKP circuit, the model weights and local training data are private inputs, the validation dataset is a public input, and the resulting model accuracy is a public output.
b) **Setup Phase:** The proving system requires a setup phase to generate cryptographic parameters. This process generates a Structured Reference String (SRS), a set of public parameters such as *kzg_srs.params*, to create and verify proofs. Unlike Groth16, which requires a new trusted setup for every distinct circuit, Halo2's universal setup can be reused across many circuits, as long as they fall within a predefined degree bound. This makes Halo2 particularly well-suited for FL applications, where the proving circuit may change frequently.
c) **Witness Generation:** The witness consists of all intermediate values produced during the computation (e.g., activations and layer outputs) given the model and inputs. During execution, the computation engine records these intermediate results, which serve as private input (the witness) to the ZKP. The recorded values are stored in a file (e.g., *witness.json*) and are used to prove correct computation.
d) **Proof Generation:** Using the compiled circuit, SRS, and witness, the client generates a succinct ZK-SNARK proof, *proof.json*, showing that it correctly participated in training, and a claimed accuracy is obtained. A ZKP file *proof.json* is sent to the verifier for verification.
e) **Proof Verification:** The verifier receives the proof, *proof.json*, the verification key, and the claimed public output of the computation, i.e., the model accuracy. The verifier then runs "*verify*" to check that the client honestly participated in the training, and this claimed accuracy is consistent with the hidden computation, without learning any private data, intermediate values, or model weights. The verifier is cryptographically convinced of the correctness of the computation if and only if the proof is valid.

4.3 Blockchain integration

Algorithm 4 PoRWV Consensus Algorithm

Input: Client report $[ClientID_i, CID_i^r, Sign_i^r, ZKP_i^r, Acc_i^r, TS_i^r, HELM_i^r]$, Voting power Table VT
Output: Finalized block $Block^r$ with $[ClientID_i, r, TS_i^r, Sign_i^r, HELM_i^r, Block\_Hash]$ stored on Blockchain
Server Operations:
Retrieve ELM_i^r from IPFS using CID_i^r
Compute hash of retrieved model and compare with $HELM_i^r$ for integrity
Temporarily store transaction TXN_i^r in mempool
Compute transaction hash $TXN\_Hash^r \leftarrow \mathtt{SHA256}(TXN_i^r)$
$\mathcal{S}$ proposes a block containing all approved TXN_Hash^r
Compute Merkle root $Merkle\_hash^r$ for the block
Calculate reputation score for each client: $Rep_i^{r+1} = (sf \cdot Rep_i^r + (1 - sf) \cdot Acc_i^r) \cdot df$
Normalize reputation scores: $\bar{R}_i^{r+1} = \frac{Rep_i^{r+1}}{\sum_{j=1}^{N} Rep_j^{r+1}}$
Client Validation Phase:
Each client C_i pulls TXN_{i-1}^r from mempool
Verify $Sign_{i-1}^r$ using Pub_key_{i-1} to ensure authenticity
Verify ZKP_{i-1}^r using EZKL
if Valid **then**
    Sign and approve $TXN\_Hash_{i-1}^r$ back to $\mathcal{S}$
else
    Report invalid transaction
end if
Consensus Phase (PoRWV):
Use z-score analysis on $\bar{R}_i^{r+1}$ to classify clients into reputation tiers
Each client casts a vote (up/down) based on their reputation tier vote weight from VT
Compute total weighted upvotes V_{yes}^r and downvotes V_{no}^r
if V_{yes}^r exceeds V_{no}^r **then**
    Finalize and broadcast $Block^r$ to the blockchain
else
    Discard block and penalize false voters: $vw_i^r \leftarrow vw_i^r \times 0.9$
end if

The proposed ZK-RBFL framework leverages blockchain technology and IPFS to ensure immutability and non-repudiation for storing model updates, transactions, and client participation details. Instead of storing ELM_i^r directly on-chain, it is stored on IPFS, and its content identifier (CID_i^r) is shared with the server for verification. Only transaction details are recorded on the blockchain to reduce storage overhead. The decision to store these details is made through consensus among all participating clients. To ensure scalability, we introduce a new FL-centric consensus algorithm, PoRWV, which is detailed in Algorithm 4. It overcomes the high computational cost of PoW, the centralization risk of PoS, and the validator dependency of PoA and PoR.

Our proposed hybrid consensus mechanism, PoRWV, builds on PoR and integrates it with a reputation-aware weighted voting strategy. In this mechanism, each client is assigned a reputation score during the FL process based on their performance. Based on these scores, clients are categorized into *high*, *medium*, and *low* reputation tiers. Each client is assigned a voting power proportional to its reputation.

When the server proposes a block containing approved client transactions, all clients participate in a time-bound voting phase. Each client casts an upvote or downvote for the block. Once voting ends, all votes are collected and weighted based on the clients' assigned voting power as per the voting power table (VT) in Table 3.

Higher-reputation clients have more influence in the final decision, while lower-reputation clients still contribute to the outcome, ensuring democratic participation. To discourage dishonest behavior, clients who vote incorrectly (e.g., approve a block that is later deemed malicious) will have their voting power reduced by 10% for each false vote, promoting accountability and encouraging honest participation. This way, PoRWV consensus mechanism maintains decentralization while incentivizing trustworthy behavior, aligning with the security and transparency goals of FL on blockchain.

TABLE 3: Reputation Tiers and Assigned Voting Power

Reputation Tier	Voting Power Weight
High Reputation	10
Medium Reputation	5
Low Reputation	1

Reputation-based Tier Classification

To determine each client's voting power, we classify them into reputation tiers based on the z-score of their normalized reputation scores $\bar{R}_i^r$. The z-score is computed as:

$$Z_i^r = \frac{\bar{R}_i^r - \mu^r}{\sigma^r} \tag{10}$$

where μ^r and σ^r are the mean and standard deviation of the reputation scores in round r. Based on the z-score, clients are divided as follows: i) Clients with $Z_i^r \geq 0.5$ are assigned to the high-reputation tier; ii) Clients with $-0.5 < Z_i^r < 0.5$ are assigned to the medium-reputation tier; iii) Clients with $Z_i^r \leq -0.5$ are assigned to the low-reputation tier.

The vote weight vw_i^r is then assigned as:

$$vw_i^r = \begin{cases} 10, & \text{if } Z_i^r \geq +0.5 \\ 5, & \text{if } -0.5 < Z_i^r < +0.5 \\ 1, & \text{if } Z_i^r \leq -0.5 \end{cases} \tag{11}$$

This approach ensures a statistically grounded and adaptive distribution of voting power that reflects fluctuations in client performance between rounds.

4.4 Server-side Aggregation

The server aggregates the model updates only if the proposed block is approved; otherwise, the updates are discarded, and the clients are called for a new round of training. Upon block approval, the server retrieves the encrypted local models ELM_i^r from IPFS using their respective CID_i^r and performs reputation-weighted aggregation instead of simple FedAvg. This is because FedAvg treats all client models equally, which may not be ideal in real-world scenarios.

In our work, client models are aggregated using their reputation scores Rep_i^r as weights. These scores are derived from the client's validation accuracy Acc_i^r, and dynamically updated using smoothing sf and decay factors df. The reputation update rule is given as:

$$Rep_i^{r+1} = (sf \cdot Rep_i^r + (1 - sf) \cdot Acc_i^r) \cdot df \tag{12}$$

These reputation scores are then normalized to the range $[0, 1]$ to ensure they form a proper weighted distribution for aggregation:

$$\bar{R}_i^{r+1} = \frac{Rep_i^{r+1}}{\sum_{j=1}^{N} Rep_j^{r+1}} \tag{13}$$

The global model for the next round is aggregated using the normalized reputation scores $\bar{R}_i^{r+1}$ as follows:

$$EGM^{r+1} = \sum_{i=1}^{N} \bar{R}_i^{r+1} \cdot ELM_i^r \tag{14}$$

After aggregation, the new global model EGM^{r+1} is stored in IPFS, and its content identifier EGM_CID^{r+1} is recorded on the Blockchain. This allows all clients to securely access the latest global model for the next round of FL training. Clients decrypt EGM^{r+1} to obtain GM^{r+1}, and the FL process is repeated for R rounds or until the model converges.

$$GM^{r+1} = \mathsf{Dec}_{\mathsf{CKKS}}(EGM^{r+1}) \tag{15}$$

5 Evaluations and Analysis

5.1 Experimental Setup

The proposed ZK-RBFL framework is evaluated using a brain tumor classification task on a dataset comprising 7,023 MRI images categorized into meningioma, glioma, pituitary tumor, and no tumor classes [114]. The dataset is formed by combining three popular brain tumor classification datasets, SARTAJ [115], Figshare [116], and Br35H [117]. Experiments are conducted in an FL environment with ten clients and one central server, where the dataset is evenly distributed among clients. All experiments are conducted under an IID data partition to focus on the security and verifiability properties of the proposed framework. However, handling non-IID data distributions is left for future work.

The LeNet architecture is selected as the global model due to its lightweight structure and ZKP compatibility. LeNet is particularly suitable for ZKP generation due to its lightweight design with a reduced number of parameters and minimal computational overhead, enabling faster and more efficient proof generation without compromising classification performance. It consists of two convolutional layers followed by three fully connected layers, ending with a four-unit output layer corresponding to the four brain tumor classes. We apply ReLU activations after each layer, except for the final one, where no activation is used. This is because the CrossEntropyLoss function applied during training internally handles the softmax computation. The

model is trained for 20 rounds, each with 30 local epochs, using an SGD optimizer (learning rate = 0.01, momentum = 0.9, batch size = 32) and cross-entropy loss.

All FL and private custom blockchain components, including IPFS, are developed from scratch in Python (Jupyter Notebook), with ZKP generation using ezkl in Google Colab. Training and intensive computations are run on an NVIDIA DGX server, while integration and end-to-end testing are conducted on an HP desktop (Intel i7, 16 GB RAM).

5.2 Results and analysis

5.2.1 ZKP validation

ZKP validation requires a sequence of operations, including circuit setup, proving and verification key generation, proof generation, and verification. The time required at each stage is provided in Table 4.

TABLE 4: ZKP experimentation

Task	Time (s)
Settings Generation	16
Circuit Compilation	0.022
SRS Retrieval	0.517
Witness Generation	2.328
Mock Proving	0.003
Setup Phase	0.052
Verification Key Generation	70.27
Proving Key Generation	191.6
Proof generation	360.77
Proof verification	0.12

TABLE 5: Evaluation results

Class label	Accuracy: 94.22%		
	Precision	Recall	F1-score
Menigioma	0.95	0.86	0.90
Glioma	0.88	0.92	0.90
Pituitary	0.99	0.99	0.99
No Tumor	0.93	0.99	0.96

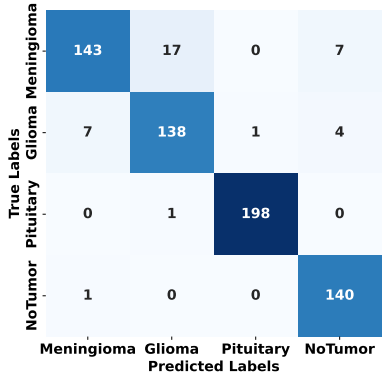

Fig. 3: Confusion matrix of the proposed ZK-RBFL framework

5.2.2 Model Performance Evaluation

The global model is recursively trained by all 10 clients using their local data throughout the FL process, and an aggregated final global model is obtained. The performance of this model is tested using various evaluation metrics, and the obtained results are shown in Table 5. The confusion matrix of the proposed ZK-RBFL framework is presented in Figure 3. Figure 4 shows a sample snapshot of the custom blockchain storing the model hashes, digital signatures, ZKP proofs, and related client information from two clients across FL rounds.


```
[
  {
    "BlockId": "Block-1",
    "Timestamp": "2025-06-20 16:08:53 (1750415933.13
        5921)",
    "RoundNumber": 0,
    "GlobalModelHash": "0",
    "MerkleRoot": "0",
    "PreviousHash": null,
    "BlockHash": "1a38a...",
    "ClientSubmissions": []
  },
  {
    "BlockId": "Block-2",
    "Timestamp": "2025-06-20 16:08:53 (1750415933.14
        34681)",
    "RoundNumber": 1,
    "GlobalModelHash": "def6c...",
    "MerkleRoot": "6ca8f...",
    "PreviousHash": "1a38a...",
    "BlockHash": "045b4...",
    "ClientSubmissions": [
      {
        "ClientId": "client_0",
        "HELM_Hash": "f7e94...",
        "Signature": "6552f...",
        "ZKP": "client_0_r1_proof.json",
        "Accuracy": 0.90
      },
      {
        "ClientId": "client_1",
        "HELM_Hash": "ae519...",
        "Signature": "d07b1...",
        "ZKP": "client_1_r1_proof.json",
        "Accuracy": 0.91
      }
    ]
  }
]
```


Fig. 4: A snapshot of blockchain storage

5.2.3 Time-based analysis

The time required for X.509 certificate generation and verification is shown in Figure 5a. As the number of clients increases, the time required also increases, and the time required for verification is more than that for generation.

As shown in Figure 5b, the centralized signature verification model shows a clear linear growth in total verification time with increasing client count. This highlights the scalability limitation of centralized server-based verification. In contrast, mutual verification in our ZK-RBFL framework maintains constant per-client overhead by distributing verification responsibilities, thereby avoiding server-side bottlenecks. Figures 5c and 5d show the comparison for block finalization time and transactions processed per second (TPS) with an increasing number of clients.

Each client trains the local model, then encrypts it using CKKS HE, and uploads the encrypted trained model to IPFS. This process is performed in parallel by all the clients participating in the FL process. To facilitate this, we have considered a multiprocessing simulation environment. IPFS returns a CID, which is sent to the server along with the other transaction details. Upon receiving the CIDs, the server downloads the models and verifies that the downloaded model matches the model signed and uploaded by the client using the hash function. Figures 5e and 5f illustrate the time required for IPFS upload and download as the number of clients increases. Here, the time is again compared for a normal model and an encrypted model. The normal model is 214 KB, but the encrypted model is 7.33 MB in size. Hence, the time required for the encrypted model is more than that of the normal model.

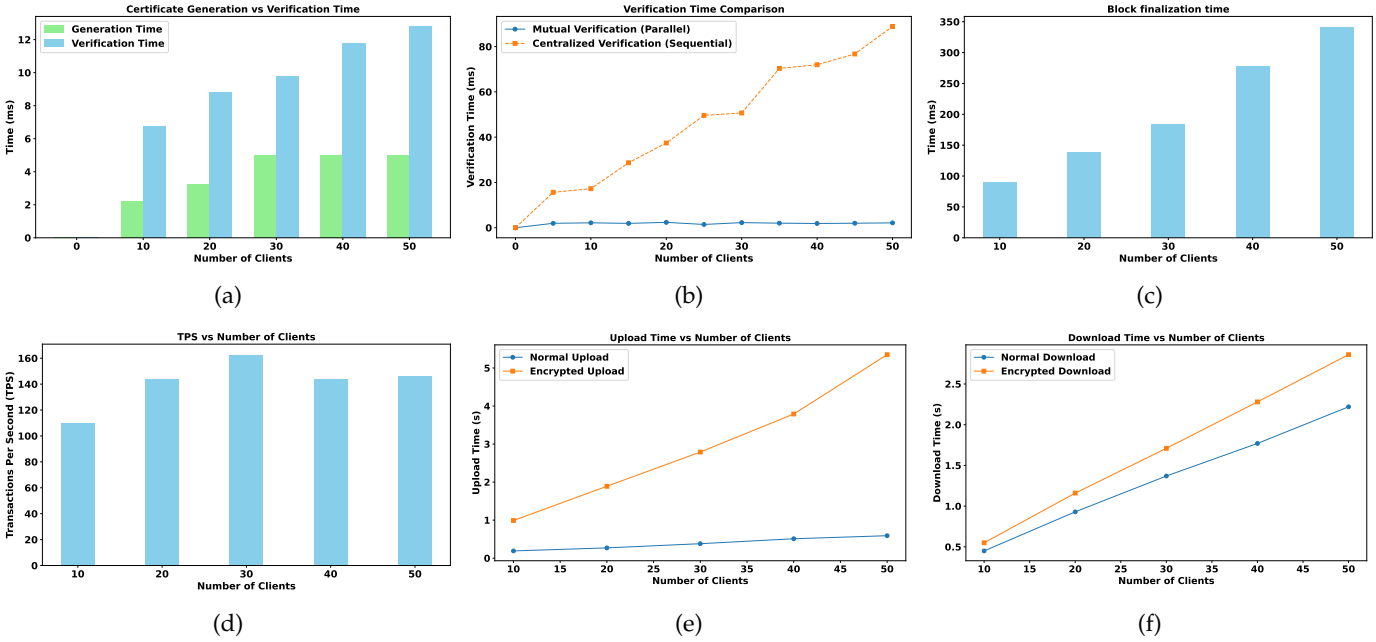

Fig. 5: Comparison across varying numbers of clients: (a) Certificate Generation and Verification Time (b) Mutual Verification vs Centralized Verification (c) Block Finalization Time (d) Transactions Per Second (TPS) (e) IPFS Upload Time (f) IPFS Download Time

TABLE 6: Ablation Study on Model Accuracy under Cryptographic Constraints

Dataset	Model	# Clients	CHE	ZC	A (%)
Brain MRI	ResNet18	10	Yes	No	98.32
	ResNet18	10	No	No	98.48
	LeNet	10	No	Yes	95.43
	LeNet (Proposed)	10	Yes	Yes	94.22
	LeNet (Proposed)	15	Yes	Yes	92.24
MNIST	LeNet	10	No	No	99.44
	LeNet (Proposed)	10	Yes	Yes	99.02
	LeNet (Proposed)	15	Yes	Yes	99.24

CHE = CKKS HE, ZC = ZKP Compatibility, A = Accuracy

5.2.4 *Ablation Studies*

We conduct ablation experiments under multiple scenarios to analyze the contribution of each component in the proposed ZK-RBFL framework. The results are tabulated in Tables 6 and 7. From Table 6, ResNet18 achieves higher classification accuracy than LeNet; however, current ZKP frameworks support only lightweight models such as LeNet, and ZKP-based verification is infeasible for complex and deeper architectures. The integration of CKKS HE results in a marginal performance degradation in exchange for robust privacy guarantees. We further evaluated the proposed framework on the MNIST dataset along with the Brain MRI dataset. The performance remains consistent with the standard FL approach with minimal degradation.

The number of clients is intentionally kept low to reflect cross-silo FL scenarios, where participants correspond to a limited number of hospitals. However, the experiments are also repeated with 15 clients to analyze scalability. In Table 7, we can observe that the decision flip rate is significantly lower when both ZKP and PoRWV are enabled compared to scenarios where either or both are absent. ZKP prevents clients from falsely reporting validation accuracy, while PoRWV limits the influence of malicious clients during the decision-making.

In ZK-RBFL, the blockchain integration enforces non-repudiation and auditability by immutably recording authenticated client updates, reputation scores, and voting outcomes. This ensures that clients cannot deny their participation or alter past contributions, thereby complementing ZKP- and PoRWV-based verification with system-level trust guarantees.

5.3 Formal Security Analysis

In order to ensure that our proposed ZK-RBFL framework is resistant to any potential attacks, we have simulated some underperforming clients. The validation accuracy across the 10 clients can be seen in Figure 6a with no underperforming clients. While Figure 6b illustrates the validation accuracy with 3 underperforming clients. At round 7, we performed a drop and spike simulation and made client 3 a well-performing client and client 5 an underperforming client. Figure 6c highlights the resistance of the proposed ZK-RBFL framework approach to these simulations, where ZK-RBFL significantly outperformed the Standard FL approach. The performance comparison of the proposed ZK-RBFL with existing works is shown in Figure 6d.

We further compared the impact of collusion on the proposed PoRWV consensus approach with the existing ones and observed that the proposed PoRWV has better collusion resistance, as shown in Figure 6e. A tier-wise collusion impact on the proposed PoRWV consensus is also analysed in Figure 6f.

To further prove that the ZK-RBFL framework is secure against all potential attacks, we conducted a formal security analysis using the Scyther tool. The results of this analysis are illustrated in Figure 7. As shown, the framework successfully resists the identified adversarial attempts, demon-

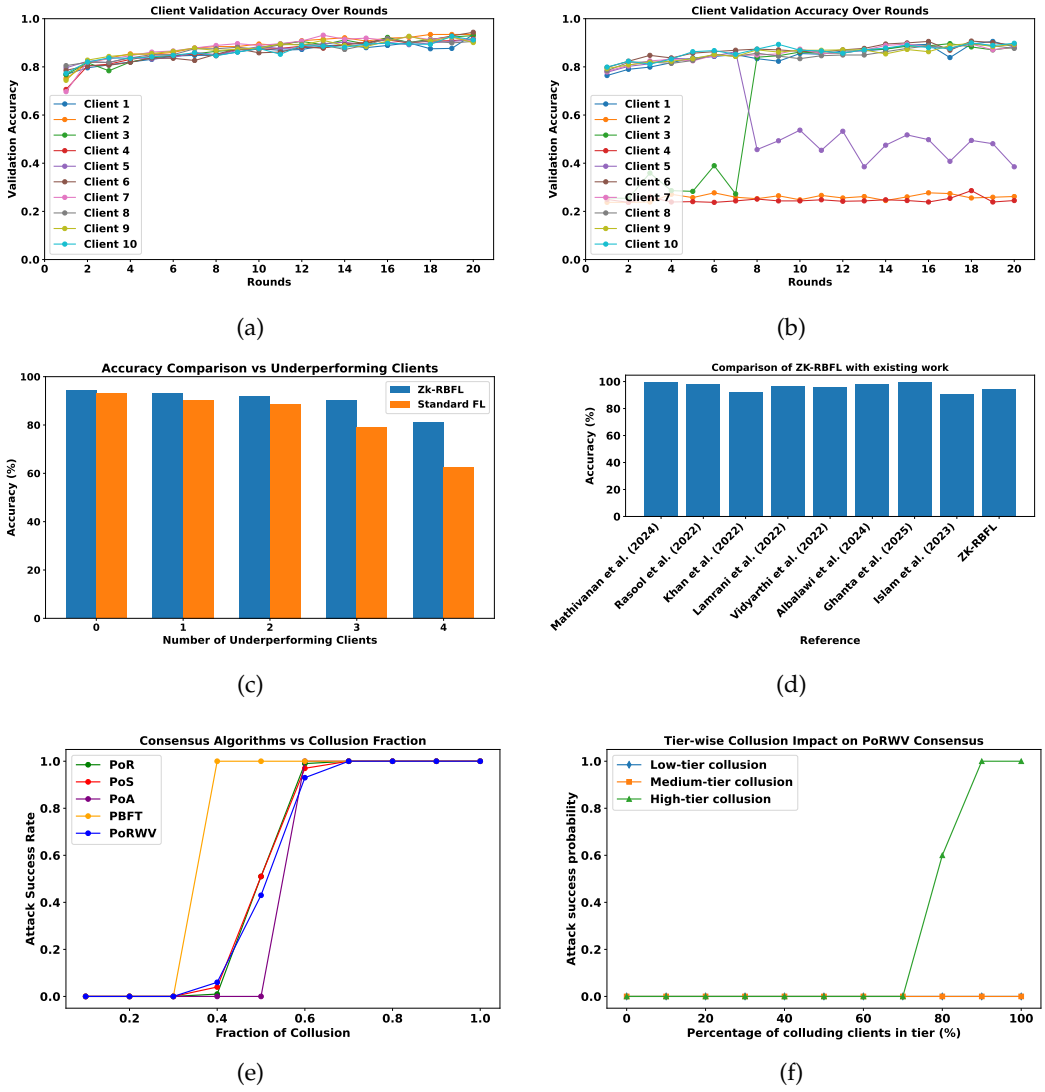

Fig. 6: Results comparison: (a) Validation accuracy with no underperforming clients, (b) Validation accuracy with three underperforming clients under drop-and-spike simulation, (c) Accuracy comparison with varying numbers of underperforming clients, (d) Accuracy comparison of the proposed ZK-RBFL with existing work, (e) Impact of collusion on consensus approaches, (f) Tier-wise collusion analysis on PoRWV consensus

strating its robustness and reliability against a wide range of security threats. This formal verification provides strong evidence of the framework's ability to maintain security properties under various threat models.

5.3.1 *Security Guarantees*

a) Privacy Against Honest-but-Curious Server

In ZK-RBFL, client updates are encrypted using CKKS HE, whose security relies on the hardness of the Ring Learning With Errors (RLWE) problem [43] and achieves semantic security. As a result, the server only observes encrypted model updates and aggregation outputs, and learns no meaningful information about individual training samples beyond what is inherently revealed by the final global model. We emphasize that the above privacy guarantees are assumption-based and aligned with widely adopted system-level security arguments in HE-based FL.

b) On the security of single-step training attestation

The one-step SGD proof is generated in response to a computationally unpredictable challenge derived from public round parameters and fresh randomness. This makes it computationally infeasible for a client to anticipate which training step will be verified. As a result, a client that does not participate in the training honestly cannot fabricate valid intermediate values or gradients consistent with the prescribed learning procedure. The space of possible training states and gradients is sufficiently large to prevent precomputation or replay attacks in practice. This ensures that the proof serves as a strong cryptographic deterrent against lazy or fabricated client updates.

c) Adversarial Analysis of PoRWV

PoRWV is designed as a lightweight, application-layer FL-centric consensus mechanism for coordination in a permissioned environment, rather than a fully permissionless blockchain consensus. Client identities are authenticated using X.509 certificates and continuously verified through

TABLE 7: Ablation study on the impact of ZKP and PoRWV under varying numbers of false clients

False Clients	Scenario-based Anlaysis											
	ZKP ✗, PoRWV ✗			ZKP ✗, PoRWV ✓			ZKP ✓, PoRWV ✗			ZKP ✓, PoRWV ✓		
	Yes	No	Decision	Yes	No	Decision	Yes	No	Decision	Yes	No	Decision
0	10	0	Yes	54	0	Yes	10	0	Yes	54	0	Yes
1	9	1	Yes	54	10	Yes	9	1	Yes	54	1	Yes
2	8	2	Yes	48	20	Yes	8	2	Yes	53	2	Yes
3	7	3	Yes	32	30	Yes	7	3	Yes	52	3	Yes
4	6	4	Yes	31	20	Yes	6	4	Yes	51	4	Yes
5	5	5	No	18	50	No	5	5	No	50	5	Yes
6	4	6	No	8	60	No	4	6	No	40	6	Yes
7	3	7	No	7	61	No	3	7	No	30	7	Yes
8	2	8	No	6	62	No	2	8	No	20	16	Yes
9	1	9	No	1	67	No	1	9	No	10	45	No
10	0	10	No	0	68	No	0	10	No	0	36	No
Decision Flip Rate (%)	50			50			50			20		

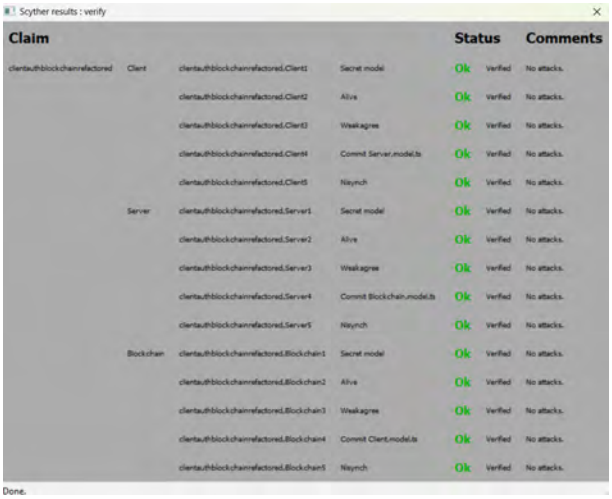

Fig. 7: Scyther security analysis

a token-based authentication mechanism employing short-lived access tokens and refresh tokens, limiting Sybil attacks. Reputation scores are derived from ZKP-verified model performance and are dynamically updated using smoothing and decay factors to prevent single-round manipulation, while persistent adversarial behavior leads to reputation decay and reduced long-term influence.

While colluding high-reputation clients could influence voting, PoRWV remains correct as long as a majority of reputation-weighted voting power is held by honest clients. This is an assumption consistent with widely adopted consensus mechanisms. Under this condition, and assuming a minimum level of active client participation per round, PoRWV preserves safety and liveness.

5.4 Discussion

The feature comparison between existing work and the proposed ZK-RBFL framework is presented in Table 8. We also compared the features offered by the proposed FL-centric PoRWV with those of popular blockchain consensus mechanisms. The distinctions are summarized in Table 9.

Furthermore, from Figure 6d, we observe that Mathivanan et al. (2024) [118] and Rasool et al. (2022) [119] reported comparatively higher accuracy than our approach; however, their methods rely solely on centralized learning, offering no privacy or security guarantees. Under the same evaluation setting, our framework outperformed Khan et al. (2022) [120] and achieved competitive results with Lamrani et al. (2022) [121] and Vidyarthi et al. (2022) [122]. In FL settings, our framework outperformed Islam et al. (2023) [123] and achieved comparable accuracy to Albalawi et al. (2024) [124] and Ghanta et al. (2025) [76].

Since we employed a simple ZKP-compatible LeNet architecture as the global model, the accuracy is slightly lower compared to state-of-the-art architectures. Nevertheless, the proposed framework offers a favorable trade-off by enabling zero-knowledge proofs for both model training participation and reported accuracy, thereby ensuring cryptographic evidence of honest training and reported accuracy.

TABLE 8: Feature comparison between existing works and the ZK-RBFL framework

Ref.	Year	ZKP	HE	BC	RM	AM	ZAV	SM	MD
[51]	2025	✗	✗	✗	✗	✗	✗	✗	✓
[52]	2025	✗	✗	✗	✗	✗	✗	✓	✗
[53]	2025	✗	✗	✗	✗	✗	✗	✓	✗
[54]	2023	✗	✗	✗	✓	✗	✗	✓	✓
[55]	2023	✗	✗	✗	✗	✗	✗	✗	✗
[68]	2022	✗	✓	✗	✓	✗	✗	✓	✓
[74]	2024	✗	✓	✗	✗	✗	✗	✓	✓
[75]	2025	✗	✓	✗	✗	✗	✗	✓	✓
[76]	2025	✗	✓	✗	✓	✗	✗	✓	✓
[77]	2024	✗	✗	✓	✗	✗	✗	✓	✓
[78]	2024	✗	✗	✓	✗	✗	✗	✓	✗
[79]	2022	✗	✗	✓	✗	✗	✗	✗	✓
[80]	2024	✗	✗	✓	✗	✗	✗	✗	✓
[81]	2023	✗	✗	✓	✗	✗	✗	✗	✓
[82]	2020	✗	✗	✓	✗	✗	✗	✗	✗
[83]	2020	✗	✗	✓	✗	✗	✗	✗	✗
[84]	2022	✗	✗	✓	✗	✗	✗	✗	✗
[85]	2023	✗	✗	✓	✗	✓	✗	✓	✗
[86]	2023	✗	✓	✓	✓	✓	✗	✓	✗
[87]	2024	✗	✓	✓	✓	✓	✗	✗	✓
[88]	2023	✓	✗	✓	✗	✓	✗	✗	✗
[89]	2024	✓	✓	✓	✗	✗	✗	✓	✗
[90]	2023	✓	✗	✓	✗	✗	✗	✗	✗
[91]	2025	✓	✗	✓	✗	✓	✗	✗	✓
[92]	2023	✓	✓	✗	✗	✗	✗	✓	✗
[93]	2025	✓	✗	✗	✗	✗	✗	✗	✗
ZK-RBFL		✓	✓	✓	✓	✓	✓	✓	✓

BC=Blockchain, RM=Reputation Management, SM=Scalability Management
AM=Authentication Management, ZAV=ZKP-based Accuracy Verifiability
MD=Medical Data

TABLE 9: Comparison of PoRWV with Existing Consensus Mechanisms

Feature	Consensus Mechanism						
	PoW	**PoS**	**DPoS**	**PoA**	**PoR**	**PBFT**	**PoRWV (Proposed)**
Energy Efficient	No	Yes	Yes	Yes	Yes	Yes	**Yes**
Scalability	No	Yes	Yes	Yes	Yes	Yes	**Yes**
Reputation-Based	No	No	limited	No	Yes	No	**Yes (dynamic reputation tiers)**
Voting Mechanism	No	No	Yes	No	Yes	Yes	**Yes (weighted voting)**
Validator Selection	Mining	Stake	Delegate	Authority	Reputation	Byzantine	**Yes (Reputation based voting)**
Byzantine Fault Tolerant	No	limited	limited	No	No	Yes	**Yes (penalizes invalid votes)**
Authentication	No	No	limited	Yes	No	limited	**Yes (X.509 + token)**
Model/Transaction Verification	No	No	No	No	No	No	**Yes (ZKP + signature)**
Tamper Detection	limited	limited	limited	limited	Yes	Yes	**Yes (hash + signature)**
Penalty for Malicious Actors	No	No	Yes	No	limited	limited	**Yes (vote weight reduced)**
Designed for FL	No	No	No	No	No	No	**Yes**

6 Conclusion

This paper presented ZK-RBFL, a novel and secure privacy-preserving BFL framework that integrates token-based authentication, ZKP, and reputation-based aggregation to ensure trust, accountability, and a decentralized learning environment. The proposed PoRWV is an FL-centric consensus mechanism tailored to meet the FL requirements. We evaluated the framework across various metrics such as model performance, consensus reliability, and simulated attack scenarios. The results demonstrate that the proposed PoRWV consensus consistently achieved a higher success rate than traditional consensus mechanisms. For ZKP compatibility, we consider a lightweight LeNet model and achieve a reliable 94.22% accuracy, offering a practical trade-off by enabling cryptographically verifiable client-side training participation and validation claims. We have optimized the framework by distributing the verification responsibilities among the clients, rather than relying solely on the server.

While this work focuses on homogeneous data settings, future work will explore extending the proposed ZK-RBFL framework to heterogeneous Non-IID data environments. Further, we plan to customize the global model to meet individual client requirements, thereby advancing into the domain of personalized FL. To encourage honest client participation throughout the FL process, we aim to introduce incentives for the clients.

References

[1] D. Talati, "Ai (artificial intelligence) in daily life," *Authorea Preprints*, 2024.

[2] G. Linden, B. Smith, and J. York, "Amazon. com recommendations: Item-to-item collaborative filtering," *IEEE Internet computing*, vol. 7, no. 1, pp. 76–80, 2003.

[3] S. S. Thete, R. P. Jare, M. Jungare, G. Bhagat, S. Durgule, and V. Borate, "Netflix recommendation system by genre categories using machine learning," in *2025 3rd International Conference on Device Intelligence, Computing and Communication Technologies (DICCT)*. IEEE, 2025, pp. 196–201.

[4] P. Covington, J. Adams, and E. Sargin, "Deep neural networks for youtube recommendations," in *Proceedings of the 10th ACM conference on recommender systems*, 2016, pp. 191–198.

[5] P. P. Ray, "Chatgpt: A comprehensive review on background, applications, key challenges, bias, ethics, limitations and future scope," *Internet of Things and Cyber-Physical Systems*, vol. 3, pp. 121–154, 2023.

[6] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[7] R. Zhang, H. Du, Y. Liu, D. Niyato, J. Kang, S. Sun, X. Shen, and H. V. Poor, "Interactive ai with retrieval-augmented generation for next generation networking," *IEEE Network*, vol. 38, no. 6, pp. 414–424, 2024.

[8] Z. Yang, E. Khatibi, N. Nagesh, M. Abbasian, I. Azimi, R. Jain, and A. M. Rahmani, "Chatdiet: Empowering personalized nutrition-oriented food recommender chatbots through an llm-augmented framework," *Smart Health*, vol. 32, p. 100465, 2024.

[9] S. Rasal, "Llm harmony: Multi-agent communication for problem solving," *arXiv preprint arXiv:2401.01312*, 2024.

[10] S. Asundi, "Microsoft co-pilot's role in augmenting decision intelligence for executives," *International Journal of Advanced Research in Computer Science*, vol. 14, no. 2278, pp. 10–17 148, 2025.

[11] H. Luo, Y. Yan, Y. Bian, W. Feng, R. Zhang, Y. Liu, J. Wang, G. Sun, D. Niyato, H. Yu *et al.*, "Ai reasoning for wireless communications and networking: A survey and perspectives," *arXiv preprint arXiv:2509.09193*, 2025.

[12] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.

[13] R. S. Antunes, C. André da Costa, A. Küderle, I. A. Yari, and B. Eskofier, "Federated learning for healthcare: Systematic review and architecture proposal," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 13, no. 4, pp. 1–23, 2022.

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

[15] P. M. Mammen, "Federated learning: Opportunities and challenges," *arXiv preprint arXiv:2101.05428*, 2021.

[16] M. A. Alsheikh, S. Lin, D. Niyato, and H.-P. Tan, "Machine learning in wireless sensor networks: Algorithms, strategies, and applications," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 4, pp. 1996–2018, 2014.

[17] T. Nallamothu, P. Surapaneni, and S. Bojjagani, "Ciciov-ml: Detecting iov cyber risks with machine learning and boosting techniques," in *Integrating Advanced Technologies for Enhanced Security and Efficiency*. Springer, 2025, pp. 195–205.

[18] S. Nazir and M. Kaleem, "Federated learning for medical image analysis with deep neural networks," *Diagnostics*, vol. 13, no. 9, p. 1532, 2023.

[19] P. Chitrapu, M. K. Morampudi, and H. K. Kalluri, "Robust face recognition using deep learning and ensemble classification," *IEEE Access*, 2025.

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

[21] F. D. Protection, "General data protection regulation (gdpr)," *Intersoft Consulting, Accessed in October*, vol. 24, no. 1, 2018.

[22] G. J. Annas, "Hipaa regulations—a new era of medical-record privacy?" pp. 1486–1490, 2003.

[23] W. Wang, X. Tang, Y. Wang, Y. Lin, T. Zhang, M. Shen, D. Niyato, and L. Zhu, "Label inference attacks against federated unlearning," *arXiv preprint arXiv:2508.06789*, 2025.

[24] J. Konečnỳ, H. B. McMahan, D. Ramage, and P. Richtárik, "Federated optimization: Distributed machine learning for on-device intelligence," *arXiv preprint arXiv:1610.02527*, 2016.

[25] J. Konečnỳ, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," *arXiv preprint arXiv:1610.05492*, 2016.

[26] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks

from decentralized data," in *Artificial intelligence and statistics*. PMLR, 2017, pp. 1273–1282.

[27] Y. Liu, J. James, J. Kang, D. Niyato, and S. Zhang, "Privacy-preserving traffic flow prediction: A federated learning approach," *IEEE Internet of Things Journal*, vol. 7, no. 8, pp. 7751–7763, 2020.

[28] P. Zhao, Y. Huang, J. Gao, L. Xing, H. Wu, and H. Ma, "Federated learning-based collaborative authentication protocol for shared data in social iov," *IEEE Sensors Journal*, vol. 22, no. 7, pp. 7385–7398, 2022.

[29] Z. Wang, M. Song, Z. Zhang, Y. Song, Q. Wang, and H. Qi, "Beyond inferring class representatives: User-level privacy leakage from federated learning," in *IEEE INFOCOM 2019-IEEE conference on computer communications*. IEEE, 2019, pp. 2512–2520.

[30] C. Dwork, A. Roth *et al.*, "The algorithmic foundations of differential privacy," *Foundations and trends® in theoretical computer science*, vol. 9, no. 3–4, pp. 211–407, 2014.

[31] R. Canetti, Y. Lindell, R. Ostrovsky, and A. Sahai, "Universally composable two-party and multi-party secure computation," in *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, 2002, pp. 494–503.

[32] S. T. Arasteh, A. Ziller, C. Kuhl, M. Makowski, S. Nebelung, R. Braren, D. Rueckert, D. Truhn, and G. Kaissis, "Private, fair and accurate: Training large-scale, privacy-preserving ai models in medical imaging," *arXiv preprint arXiv:2302.01622*, 2023.

[33] G. A. Kaissis, M. R. Makowski, D. Rückert, and R. F. Braren, "Secure, privacy-preserving and federated machine learning in medical imaging," *Nature Machine Intelligence*, vol. 2, no. 6, pp. 305–311, 2020.

[34] C. Li, Z. Xing, J. Liu, G. Russello, Z. Li, Y. Wu, M. Li, and M. R. Asghar, "Integrating zero-knowledge proofs into federated learning: a path to on-chain verifiable and privacy-preserving federated learning frameworks," *International Journal of Web Information Systems*, vol. 21, no. 3, pp. 275–297, 2025.

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

[36] M. J. Sheller, B. Edwards, G. A. Reina, J. Martin, S. Pati, A. Kotrotsou, M. Milchenko, W. Xu, D. Marcus, R. R. Colen, and S. Bakas, "Federated learning in medicine: facilitating multi-institutional collaborations without sharing patient data," *Scientific reports*, vol. 10, no. 1, p. 12598, 2020.

[37] C. Zhang, S. Li, J. Xia, W. Wang, F. Yan, and Y. Liu, "{BatchCrypt}: Efficient homomorphic encryption for {Cross-Silo} federated learning," in *2020 USENIX annual technical conference (USENIX ATC 20)*, 2020, pp. 493–506.

[38] N. Wang, J. Zhang, J. Huang, W. Ou, W. Han, and Q. Zhang, "Telemedicine data secure sharing scheme based on heterogeneous federated learning," *Cybersecurity*, vol. 7, no. 1, p. 56, 2024.

[39] D. Usynin, A. Ziller, M. Makowski, R. Braren, D. Rueckert, B. Glocker, G. Kaissis, and J. Passerat-Palmbach, "Adversarial interference and its mitigations in privacy-preserving collaborative machine learning," *Nature Machine Intelligence*, vol. 3, no. 9, pp. 749–758, 2021.

[40] M. Fredrikson, S. Jha, and T. Ristenpart, "Model inversion attacks that exploit confidence information and basic countermeasures," in *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, 2015, pp. 1322–1333.

[41] S. Bojjagani, D. Brabin, K. Kumar, N. K. Sharma, and U. Batta, "Secure privacy-enhanced fast authentication and key management for iomt-enabled smart healthcare systems," *Computing*, vol. 106, no. 7, pp. 2427–2458, 2024.

[42] P. Surapaneni, S. Bojjagani, and M. K. Khan, "Dynamic-trust: Blockchain-enhanced trust for secure vehicle transitions in intelligent transport systems," *IEEE Transactions on Intelligent Transportation Systems*, 2025.

[43] J. H. Cheon, A. Kim, M. Kim, and Y. Song, "Homomorphic encryption for arithmetic of approximate numbers," in *International conference on the theory and application of cryptology and information security*. Springer, 2017, pp. 409–437.

[44] D. Kang, T. Hashimoto, I. Stoica, and Y. Sun, "Scaling up trustless dnn inference with zero-knowledge proofs," *arXiv preprint arXiv:2210.08674*, 2022.

[45] X. Tang, M. Li, M. Shen, J. Kang, L. Zhu, Z. Liu, G. Yang, D. Niyato, and R. H. Deng, "Roby: A byzantine-robust and privacy-preserving serverless federated learning framework," *IEEE Transactions on Information Forensics and Security*, 2025.

[46] S. Bowe, J. Grigg, and D. Hopwood, "Recursive proof composition without a trusted setup," *Cryptology ePrint Archive*, 2019.

[47] T. Liu, X. Xie, and Y. Zhang, "Zkcnn: Zero knowledge proofs for convolutional neural network predictions and accuracy," in *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, 2021, pp. 2968–2985.

[48] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," *Available at SSRN 3440802*, 2008.

[49] C. T. Nguyen, D. T. Hoang, D. N. Nguyen, D. Niyato, H. T. Nguyen, and E. Dutkiewicz, "Proof-of-stake consensus mechanisms for future blockchain networks: fundamentals, applications and opportunities," *IEEE access*, vol. 7, pp. 85 727–85 745, 2019.

[50] P. Surapaneni, S. Bojjagani, and A. K. Maurya, "Handover-authentication scheme for internet of vehicles (iov) using blockchain and hybrid computing," *IEEE Access*, 2024.

[51] S. Ghanta, A. Thiriveedhi, P. Boyapati, and A. K. Pradhan, "Federated transfer learning for chest x-ray classification: An explainable and generative ai framework with reliability assessment," *SN Computer Science*, vol. 6, no. 7, p. 795, 2025.

[52] Y. Li, F. Li, S. Yang, and Y. Wang, "Bgefl: Enabling communication-efficient federated learning via bandit gradient estimation in resource-constrained networks," *IEEE Transactions on Networking*, vol. 33, no. 5, pp. 2410–2425, 2025.

[53] Y. Li, S. Liu, Y. Meng, S. Qi, Z. Qu, F. Li, and Y. Wang, "Toward collaborative intelligence for meta-computing-driven iiot based on vertical federated learning with fast convergence," *IEEE Internet of Things Journal*, vol. 12, no. 10, pp. 13 806–13 816, 2025.

[54] M. Panigrahi, S. Bharti, and A. Sharma, "A reputation-aware hierarchical aggregation framework for federated learning," *Computers and Electrical Engineering*, vol. 111, p. 108900, 2023.

[55] D. Kang and C. W. Ahn, "Ga approach to optimize training client set in federated learning," *IEEE Access*, vol. 11, pp. 85 489–85 500, 2023.

[56] C. Dwork, "Differential privacy," in *International colloquium on automata, languages, and programming*. Springer, 2006, pp. 1–12.

[57] K. B. Frikken, "Secure multiparty computation," in *Algorithms and theory of computation handbook: Special topics and techniques*, 2010, pp. 14–14.

[58] C. Gentry, "A fully homomorphic encryption scheme-stanford university, ph. d. thesis, 2009," 2009.

[59] W. Zhu, P. Kairouz, B. McMahan, H. Sun, and W. Li, "Federated heavy hitters discovery with differential privacy," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2020, pp. 3837–3847.

[60] K. Wei, J. Li, M. Ding, C. Ma, H. H. Yang, F. Farokhi, S. Jin, T. Q. Quek, and H. V. Poor, "Federated learning with differential privacy: Algorithms and performance analysis," *IEEE transactions on information forensics and security*, vol. 15, pp. 3454–3469, 2020.

[61] S. Truex, L. Liu, K.-H. Chow, M. E. Gursoy, and W. Wei, "Ldp-fed: Federated learning with local differential privacy," in *Proceedings of the third ACM international workshop on edge systems, analytics and networking*, 2020, pp. 61–66.

[62] M. Kim, O. Günlü, and R. F. Schaefer, "Federated learning with local differential privacy: Trade-offs between privacy, utility, and communication," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 2650–2654.

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

[64] S. Kadhe, N. Rajaraman, O. O. Koyluoglu, and K. Ramchandran, "Fastsecagg: Scalable secure aggregation for privacy-preserving federated learning," *arXiv preprint arXiv:2009.11248*, 2020.

[65] H. Fereidooni, S. Marchal, M. Miettinen, A. Mirhoseini, H. Möllering, T. D. Nguyen, P. Rieger, A.-R. Sadeghi, T. Schneider, H. Yalame *et al.*, "Safelearn: Secure aggregation for private federated learning," in *2021 IEEE security and privacy workshops (SPW)*. IEEE, 2021, pp. 56–62.

[66] J. Xu, N. Hong, Z. Xu, Z. Zhao, C. Wu, K. Kuang, J. Wang, M. Zhu, J. Zhou, K. Ren *et al.*, "Data-driven learning for data rights, data pricing, and privacy computing," *Engineering*, vol. 25, pp. 66–76, 2023.

[67] G. K. Mahato and S. K. Chakraborty, "A comparative review on homomorphic encryption for cloud security," *IETE journal of research*, vol. 69, no. 8, pp. 5124–5133, 2023.

[68] L. Zhang, J. Xu, P. Vijayakumar, P. K. Sharma, and U. Ghosh, "Homomorphic encryption-based privacy-preserving federated learning in iot-enabled healthcare system," *IEEE transactions on network science and engineering*, vol. 10, no. 5, pp. 2864–2880, 2022.

[69] Z. Brakerski, C. Gentry, and V. Vaikuntanathan, "(leveled) fully homomorphic encryption without bootstrapping," *ACM Transactions on Computation Theory (TOCT)*, vol. 6, no. 3, pp. 1–36, 2014.

[70] R. Geelen and F. Vercauteren, "Bootstrapping for bgv and bfv revisited," *Journal of Cryptology*, vol. 36, no. 2, p. 12, 2023.

[71] J. Fan and F. Vercauteren, "Somewhat practical fully homomorphic encryption," *Cryptology ePrint Archive*, 2012.

[72] F. Wibawa, F. O. Catak, M. Kuzlu, S. Sarp, and U. Cali, "Homomorphic encryption and federated learning based privacy-preserving cnn training: Covid-19 detection use-case," in *Proceedings of the 2022 European interdisciplinary cybersecurity conference*, 2022, pp. 85–90.

[73] Y. Pan, Z. Chao, W. He, Y. Jing, L. Hongjia, and W. Liming, "Fedshe: privacy preserving and efficient federated learning with adaptive segmented ckks homomorphic encryption," *Cybersecurity*, vol. 7, no. 1, p. 40, 2024.

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

[75] A. Veda Sri, M. K. Morampudi, S. Alahari, V. V. V. Boggavarapu, J. Chennu, and S. Yakkala, "Privacy-preserving federated learning with homomorphic encryption: Alzheimer's detection use-case," in *Enabling Person-Centric Healthcare Using Ambient Assistive Technology, Volume 2: Personalized and Patient-Centric Healthcare Services in AAT*. Springer, 2025, pp. 111–125.

[76] S. Ghanta, P. Boyapati, S. Biswas, A. K. Pradhan, and S. P. Mohanty, "Enhancing privacy-preserving brain tumor classification with adaptive reputation-aware federated learning and homomorphic encryption," *PeerJ Computer Science*, vol. 11, p. e3165, 2025.

[77] S. Rajit, Z. F. Ananna, M. M. Ehsan, N. N. Punom, and S. Siddique, "Multi-class brain tumor classification of mri image using federated learning with blockchain," in *2024 IEEE Region 10 Symposium (TENSYMP)*. IEEE, 2024, pp. 1–8.

[78] Y. Li, F. Li, S. Yang, C. Zhang, L. Zhu, and Y. Wang, "A cooperative analysis to incentivize communication-efficient federated learning," *IEEE Transactions on Mobile Computing*, vol. 23, no. 10, pp. 10 175–10 190, 2024.

[79] Y. Liu, W. Yu, Z. Ai, G. Xu, L. Zhao, and Z. Tian, "A blockchain-empowered federated learning in healthcare-based cyber physical systems," *IEEE Transactions on Network Science and Engineering*, vol. 10, no. 5, pp. 2685–2696, 2022.

[80] R. Myrzashova, S. H. Alsamhi, A. Hawbani, E. Curry, M. Guizani, and X. Wei, "Safeguarding patient data-sharing: Blockchain-enabled federated learning in medical diagnostics," *IEEE Transactions on Sustainable Computing*, vol. 10, no. 1, pp. 176–189, 2024.

[81] L. Bhatia and S. Samet, "A decentralized data evaluation framework in federated learning," *Blockchain: Research and Applications*, vol. 4, no. 4, p. 100152, 2023.

[82] Y. Qu, S. R. Pokhrel, S. Garg, L. Gao, and Y. Xiang, "A blockchained federated learning framework for cognitive computing in industry 4.0 networks," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 4, pp. 2964–2973, 2020.

[83] Y. Lu, X. Huang, K. Zhang, S. Maharjan, and Y. Zhang, "Communication-efficient federated learning and permissioned blockchain for digital twin edge networks," *IEEE Internet of Things Journal*, vol. 8, no. 4, pp. 2276–2288, 2020.

[84] A. P. Kalapaaking, I. Khalil, M. S. Rahman, M. Atiquzzaman, X. Yi, and M. Almashor, "Blockchain-based federated learning with secure aggregation in trusted execution environment for internet-of-things," *IEEE Transactions on Industrial Informatics*, vol. 19, no. 2, pp. 1703–1714, 2022.

[85] M. Fan, Z. Zhang, Z. Li, G. Sun, H. Yu, and M. Guizani, "Blockchain-based decentralized and lightweight anonymous authentication for federated learning," *IEEE Transactions on Vehicular Technology*, vol. 72, no. 9, pp. 12 075–12 086, 2023.

[86] M. Fan, K. Ji, Z. Zhang, H. Yu, and G. Sun, "Lightweight privacy and security computing for blockchained federated learning in iot," *IEEE Internet of Things Journal*, vol. 10, no. 18, pp. 16 048–16 060, 2023.

[87] G. K. Mahato, A. Banerjee, S. K. Chakraborty, and X.-Z. Gao, "Privacy preserving verifiable federated learning scheme using blockchain and homomorphic encryption," *Applied Soft Computing*, vol. 167, p. 112405, 2024.

[88] S. Ji, J. Zhang, Y. Zhang, Z. Han, and C. Ma, "Lafed: A lightweight authentication mechanism for blockchain-enabled federated learning system," *Future Generation Computer Systems*, vol. 145, pp. 56–67, 2023.

[89] O. Chakraborty and A. Boudguiga, "A decentralized federated learning using reputation," *Cryptology ePrint Archive*, 2024.

[90] Z. Xing, Z. Zhang, M. Li, J. Liu, L. Zhu, G. Russello, and M. R. Asghar, "Zero-knowledge proof-based practical federated learning on blockchain," *arXiv preprint arXiv:2304.05590*, 2023.

[91] L. Petrosino, L. Masi, F. D'Antoni, M. Merone, and L. Vollero, "A zero-knowledge proof federated learning on dlt for healthcare data," *Journal of Parallel and Distributed Computing*, vol. 196, p. 104992, 2025.

[92] B. Zhang, G. Lu, P. Qiu, X. Gui, and Y. Shi, "Advancing federated learning through verifiable computations and homomorphic encryption," *Entropy*, vol. 25, no. 11, p. 1550, 2023.

[93] X. Tang, M. Li, T. Zhang, Y. Lin, L. Zhu, C. Zhou, and Z. Liu, "zkfl: Verifiable byzantine-robust federated learning against malicious servers," *IEEE Transactions on Network Science and Engineering*, 2025.

[94] B. S. Egala, A. K. Pradhan, P. Dey, V. Badarla, and S. P. Mohanty, "Fortified-chain 2.0: Intelligent blockchain for decentralized smart healthcare system," *IEEE Internet of Things Journal*, vol. 10, no. 14, pp. 12 308–12 321, 2023.

[95] Z. Liao and S. Cheng, "Rvc: A reputation and voting based blockchain consensus mechanism for edge computing-enabled iot systems," *Journal of Network and Computer Applications*, vol. 209, p. 103510, 2023.

[96] S. Biswas, K. Sharif, Z. Latif, M. J. Alenazi, A. K. Pradhan, and A. K. Bairagi, "Blockchain controlled trustworthy federated learning platform for smart homes," *IET Communications*, vol. 18, no. 20, pp. 1840–1852, 2024.

[97] L. Feng, Y. Zhao, S. Guo, X. Qiu, W. Li, and P. Yu, "Blockchain-based asynchronous federated learning for internet of things," *IEEE Trans. Comput*, vol. 71, no. 5, pp. 1092–1103, 2021.

[98] Y. Li, Y. Yu, C. Lou, N. Guizani, and L. Wang, "Decentralized public key infrastructures atop blockchain," *IEEE Network*, vol. 34, no. 6, pp. 133–139, 2020.

[99] J. Won, A. Singla, E. Bertino, and G. Bollella, "Decentralized public key infrastructure for internet-of-things," in *MILCOM 2018-2018 IEEE Military Communications Conference (MILCOM)*. IEEE, 2018, pp. 907–913.

[100] P. S. Barreto, B. Libert, N. McCullagh, and J.-J. Quisquater, "Efficient and provably-secure identity-based signatures and signcryption from bilinear maps," in *International conference on the theory and application of cryptology and information security*. Springer, 2005, pp. 515–532.

[101] G. Xu, J. Dong, C. Ma, J. Liu, and U. G. O. Cliff, "A certificateless signcryption mechanism based on blockchain for edge computing," *IEEE Internet of Things Journal*, vol. 10, no. 14, pp. 11 960–11 974, 2022.

[102] S. Shen, H. Wang, and Y. Zhao, "Identity-based authenticated encryption with identity confidentiality," *Theoretical Computer Science*, vol. 901, pp. 1–18, 2022.

[103] Y. Zhao, "Identity-based authenticated encryption with identity confidentiality," in *European Symposium on Research in Computer Security*. Springer, 2020, pp. 633–653.

[104] W. Wang, F. H. Memon, Z. Lian, Z. Yin, T. R. Gadekallu, Q.-V. Pham, K. Dev, and C. Su, "Secure-enhanced federated learning for ai-empowered electric vehicle energy prediction," *IEEE Consumer Electronics Magazine*, vol. 12, no. 2, pp. 27–34, 2021.

[105] Y. Fan, K. Ma, L. Zhang, X. Lei, G. Xu, and G. Tan, "Validcnn: A large-scale cnn predictive integrity verification scheme based on zk-snark," *IEEE Transactions on Dependable and Secure Computing*, vol. 21, no. 6, pp. 5185–5195, 2024.

[106] J. Zhang, Z. Fang, Y. Zhang, and D. Song, "Zero knowledge proofs for decision tree predictions and accuracy," in *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, 2020, pp. 2039–2053.

[107] L. Zhao, Q. Wang, C. Wang, Q. Li, C. Shen, and B. Feng, "Veriml: Enabling integrity assurances and fair payments for machine

learning as a service," *IEEE Transactions on Parallel and Distributed Systems*, vol. 32, no. 10, pp. 2524–2540, 2021.
[108] C. Niu, F. Wu, S. Tang, S. Ma, and G. Chen, "Toward verifiable and privacy preserving machine learning prediction," *IEEE Transactions on Dependable and Secure Computing*, vol. 19, no. 3, pp. 1703–1721, 2020.
[109] Z. Ghodsi, T. Gu, and S. Garg, "Safetynets: Verifiable execution of deep neural networks on an untrusted cloud," *Advances in Neural Information Processing Systems*, vol. 30, 2017.
[110] S. Lee, H. Ko, J. Kim, and H. Oh, "vcnn: Verifiable convolutional neural network based on zk-snarks," *IEEE Transactions on Dependable and Secure Computing*, vol. 21, no. 4, pp. 4254–4270, 2024.
[111] "Ezkl," https://github.com/zkonduit/ezkl, 2025.
[112] Z. Peng, T. Wang, C. Zhao, G. Liao, Z. Lin, Y. Liu, B. Cao, L. Shi, Q. Yang, and S. Zhang, "A survey of zero-knowledge proof based verifiable machine learning," *arXiv preprint arXiv:2502.18535*, 2025.
[113] T. Xie, T. Lu, Z. Fang, S. Wang, Z. Zhang, Y. Jia, D. Song, and J. Zhang, "zkpytorch: A hierarchical optimized compiler for zero-knowledge machine learning," *Cryptology ePrint Archive*, 2025.
[114] M. Nickparvar, "Brain tumor mri dataset," https://www.kaggle.com/dsv/2645886, 2021, dOI:10.34740/KAGGLE/DSV/2645886.
[115] S. Bhuvaji, A. Kadam, P. Bhumkar, S. Dedge, and S. Kanchan, "Brain tumor classification (mri)," 2025. [Online]. Available: https://www.kaggle.com/dsv/12745533
[116] J. Cheng, "Brain tumor dataset," https://figshare.com/articles/dataset/brain_tumor_dataset/151242 2017, https://doi.org/10.6084/m9.figshare.1512427.
[117] A. Hamada, "Br35h :: Brain tumor detection 2020," 2020. [Online]. Available: https://www.kaggle.com/datasets/ahmedhamada0/brain-tumor-detection?select=no
[118] S. K. Mathivanan, S. Sonaimuthu, S. Murugesan, H. Rajadurai, B. D. Shivahare, and M. A. Shah, "Employing deep learning and transfer learning for accurate brain tumor detection," *Scientific Reports*, vol. 14, no. 1, p. 7232, 2024.
[119] M. Rasool, N. A. Ismail, W. Boulila, A. Ammar, H. Samma, W. M. Yafooz, and A.-H. M. Emara, "A hybrid deep learning model for brain tumour classification," *Entropy*, vol. 24, no. 6, p. 799, 2022.
[120] A. H. Khan, S. Abbas, M. A. Khan, U. Farooq, W. A. Khan, S. Y. Siddiqui, and A. Ahmad, "Intelligent model for brain tumor identification using deep learning," *Applied Computational Intelligence and Soft Computing*, vol. 2022, no. 1, p. 8104054, 2022.
[121] D. Lamrani, B. Cherradi, O. El Gannour, M. A. Bouqentar, and L. Bahatti, "Brain tumor detection using mri images and convolutional neural network," *International Journal of Advanced Computer Science and Applications*, vol. 13, no. 7, 2022.
[122] A. Vidyarthi, R. Agarwal, D. Gupta, R. Sharma, D. Draheim, and P. Tiwari, "Machine learning assisted methodology for multiclass classification of malignant brain tumors," *IEEE Access*, vol. 10, pp. 50 624–50 640, 2022.
[123] M. Islam, M. T. Reza, M. Kaosar, and M. Z. Parvez, "Effectiveness of federated learning and cnn ensemble architectures for identifying brain tumors using mri images," *Neural Processing Letters*, vol. 55, no. 4, pp. 3779–3809, 2023.
[124] E. Albalawi, M. TR, A. Thakur, V. V. Kumar, M. Gupta, S. B. Khan, and A. Almusharraf, "Integrated approach of federated learning with transfer learning for classification and diagnosis of brain tumor," *BMC Medical Imaging*, vol. 24, no. 1, p. 110, 2024.

Swetha Ghanta received her B. Tech. degree and M. Tech. degree in Computer Science and Engineering from RVR & JC College of Engineering, Guntur, AP, India. She is currently pursuing her PhD degree in the Department of Computer Science and Engineering at SRM University-AP, Amaravati, India. Her research interests include deep learning, federated learning, enhanced privacy, and security approaches. She is currently working on medical image analysis using Blockchain Federated Learning.

Ashok Kumar Pradhan is an Associate Professor in the Department of Computer Science and Engineering, School of Engineering and Applied Sciences, SRM University AP, India. He received the M.Tech. degree in Computer Science and Engineering from NIT Rourkela in 2010 and the Ph.D. degree from NIT Durgapur in 2015. His research interests include optical communication and networks, IoT, blockchain, cybersecurity and privacy, machine learning, and cloud/edge computing. He has published over 35 papers in reputed journals and conferences, edited two books, contributed four book chapters, and holds one patent. He received the SERB Research Grant (TAR/2019/000286) in 2019 and serves as a reviewer for leading IEEE, Springer, and Elsevier journals.

Prasanthi Boyapati She is an Assistant Professor in the Department of Computer Science and Engineering at SRM University AP, India. She received the Ph.D. degree in Image Processing from Acharya Nagarjuna University, India, in 2019. Her research interests include medical image processing, machine learning, deep learning, and big data analytics. She has over 14 years of academic and research experience and has published widely in SCI and Scopus-indexed journals. Dr. Boyapati is a member of the ACM and IAENG and received the Best Woman Academician Award in 2020.

Sujit Biswas (Senior Member, IEEE) He is a Senior Lecturer (Associate Professor) in Cybersecurity in the Department of Computer Science at City, University of London (City St. George's), U.K. He received the Ph.D. degree in Computer Science and Technology from the Beijing Institute of Technology, China. His research interests include blockchain, federated learning, distributed consensus, and privacy-preserving AI. Dr. Biswas has published in leading journals, including the IEEE Internet of Things Journal, IEEE Transactions on Big Data, IEEE Transactions on Services Computing, IEEE Transactions on Network and Service Management, and ACM Computing Surveys. He is actively involved in industry–academia collaborative projects on trustworthy AI and blockchain-enabled systems.

Saraju P Mohanty (Senior Member, IEEE) received the B.E. degree in Electrical Engineering from the Orissa University of Agriculture and Technology, Bhubaneswar, India, in 1995, the M.E. degree in Systems Science and Automation from the Indian Institute of Science, Bengaluru, India, in 1999, and the Ph.D. degree in Computer Science and Engineering from the University of South Florida, Tampa, FL, USA, in 2003. He is a Professor with the University of North Texas, USA. His research interests include smart electronic systems, hardware-assisted security, and AI-integrated cyber-physical systems. He has authored over 550 research articles, five books, and ten patents, with an h-index of 58 and more than 15,000 citations. Dr. Mohanty is a recipient of 19 Best Paper Awards, the IEEE Consumer Electronics Society Outstanding Service Award (2020), and the Fulbright Specialist Award (2021). He has served as Editor-in-Chief of the IEEE Consumer Electronics Magazine (2016–2021) and currently serves on the editorial boards of several IEEE and ACM journals.