SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

August 2025, Vol. 68, Iss. 8, 180503:1–180503:16
https://doi.org/10.1007/s11432-025-4471-7

Special Topic: Quantum Information

Quantum-enhanced blockchain federated learning via quantum Byzantine agreement

Hao-Wen LIU[1,2], Zhi-Ping LIU[1,2], Hua-Lei YIN[2,1,3*] & Zeng-Bing CHEN[1*]

[1]*National Laboratory of Solid State Microstructures and School of Physics, Collaborative Innovation Center of Advanced Microstructures, Nanjing University, Nanjing 210093, China*
[2]*School of Physics and Beijing Key Laboratory of Opto-Electronic Functional Materials and Micro-Nano Devices, Key Laboratory of Quantum State Construction and Manipulation (Ministry of Education), Renmin University of China, Beijing 100872, China*
[3]*Beijing Academy of Quantum Information Sciences, Beijing 100193, China*

Received 27 February 2025/Revised 30 April 2025/Accepted 3 June 2025/Published online 9 July 2025

Abstract Federated learning is a powerful machine learning framework that enables collaborative model training while preserving data privacy. Blockchain-based federated learning has emerged as a decentralized solution to enhance system verifiability and security. However, existing blockchain-based federated learning approaches suffer from limited fault tolerance and high communication overhead, posing significant challenges for large-scale deployment. To address these issues, we propose a quantum-enhanced blockchain federated learning framework. It integrates quantum Byzantine agreement, enabling consensus even when nearly 50% of clients are malicious, significantly improving fault tolerance. Additionally, we leverage matrix product operators for model compression, reducing communication overhead by up to 90% while maintaining model accuracy. We design a Byzantine-resilient aggregation algorithm that effectively mitigates adversarial attacks and enhances privacy protection. Experimental results on two benchmark datasets show that even with 40% of clients being malicious, the proposed method maintains strong performance, significantly surpassing traditional approaches in terms of effectiveness, robustness, and security.

Keywords federated learning, tensor network, quantum Byzantine agreement, differential privacy, blockchain

Citation Liu H-W, Liu Z-P, Yin H-L, et al. Quantum-enhanced blockchain federated learning via quantum Byzantine agreement. Sci China Inf Sci, 2025, 68(8): 180503, https://doi.org/10.1007/s11432-025-4471-7

1 Introduction

Federated learning is a distributed machine learning paradigm designed to address privacy concerns [1]. This approach has been widely adopted in various domains, including healthcare [2], vehicular networks [3], and the Internet of Things [4]. In federated learning, clients collaboratively train a shared model by periodically downloading the global model from a central server and uploading locally computed updates based on their private datasets. This decentralized learning mechanism ensures that raw data remain on local devices, thereby preserving user privacy while enabling effective model training.

In federated learning model aggregation, federated averaging (FedAvg) [1] remains the dominant approach, but its reliance on unverified client updates makes it vulnerable to adversarial threats, such as model poisoning attacks [5–7]. Malicious participants can manipulate model updates during training, compromising the integrity and performance of the global model while undermining trust among collaborating entities. Blockchain-based federated learning has been explored as a promising solution to enhance security and trust [8, 9]. By leveraging blockchain's decentralized nature, the training process becomes more transparent and verifiable [10]. However, existing blockchain consensus mechanisms, such as practical Byzantine fault tolerance (PBFT) [11], can tolerate only up to one-third of malicious nodes, limiting their fault tolerance. Furthermore, the substantial storage and communication overhead of blockchain present scalability challenges in large-scale federated learning applications [12, 13].

To address these limitations, quantum-driven innovations are urgently needed for federated learning [14–26]. The rapid advancement of quantum technology [27, 28] presents new opportunities for federated learning, with various quantum techniques expected to significantly enhance its performance,

* Corresponding author (email: hlyin@ruc.edu.cn, zbchen@nju.edu.cn)

resilience, and trustworthiness. For instance, quantum consensus algorithms, such as quantum Byzantine agreement protocols, can significantly improve blockchain system resilience against adversarial attacks, ensuring robust and secure operations even in highly compromised environments [29–32]. Additionally, quantum-inspired compression techniques can drastically reduce blockchain's storage and computational overhead, optimizing data transmission efficiency [33, 34]. However, despite their promising capabilities, these quantum consensus algorithms and compression techniques have not yet been effectively integrated into federated learning frameworks.

To fill this gap, we propose a quantum-enhanced blockchain federated learning framework to address the fundamental limitations of existing blockchain-based federated learning systems. First, to tackle the Byzantine consensus problem while providing information-theoretic security guarantees, we incorporate a novel quantum Byzantine agreement (QBA) protocol [31]. This protocol leverages quantum cryptographic techniques to improve fault tolerance from 1/3 to 1/2, significantly enhancing system security and stability, even in adversarial environments with a high proportion of malicious nodes. Furthermore, to mitigate the substantial communication overhead of federated learning, we introduce quantum-inspired model compression techniques, specifically utilizing the matrix product operator (MPO) approach [33]. This technique enables efficient model compression while maintaining training accuracy, reducing both communication costs for model uploads and downloads.

The key contributions of this work are summarized as follows.

• We introduce QBA into federated learning, enabling honest participants to reach consensus even with up to 50% of nodes being malicious. This significantly enhances fault tolerance compared to PBFT, which requires at least $3f + 1$ nodes to tolerate f Byzantine faults.

• We employ an MPO-based compression technique, achieving over a 90% reduction in model size while effectively reducing communication costs and maintaining training accuracy. To the best of our knowledge, this marks the first application of MPO in federated learning.

• We develop a blockchain-enhanced federated learning framework that remains resilient against high proportions of malicious clients while preserving model accuracy. Additionally, we incorporate post-quantum cryptography, differential privacy, and QBA to strengthen security and privacy. Experimental results demonstrate that even with 40% of clients being malicious, our framework maintains strong performance.

This work is organized as follows. We provide some relevant background for this paper in Section 2. In Section 3, we show the assumptions and threat model in our framework. The detailed framework is shown in Section 4. The experimental results are presented in Section 5. Section 6 concludes the paper.

2 Preliminaries

2.1 Federated learning

Federated learning enables clients to collaboratively train a shared global model while preserving the privacy of their local data. In a typical federated learning setup, a central server coordinates N clients, each owning a local dataset D_i, where $i = 1, 2, \ldots, N$. The training process is conducted iteratively over multiple communication rounds.

During round t, each client i downloads the global model parameters $\boldsymbol{w}_{t-1}$ from the central server. The client then performs local training on its respective dataset using stochastic gradient descent (SGD), obtaining an updated local model $\boldsymbol{w}_t^i$. The model update, defined as the difference between the new and previous model parameters, is given by

$$\boldsymbol{d}_i^t = \boldsymbol{w}_t^i - \boldsymbol{w}_{t-1}. \tag{1}$$

After local training, each client transmits its update $\boldsymbol{d}_i^t$ back to the server. Upon receiving updates from all participating clients, the server aggregates them using the FedAvg algorithm, formulated as

$$\boldsymbol{w}_t = \boldsymbol{w}_{t-1} + \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{d}_i^t. \tag{2}$$

This aggregation mechanism ensures that the global model is updated to reflect the collective contributions of all clients. The process repeats for multiple communication rounds until a predefined convergence criterion is met or the maximum number of iterations is reached.

2.2 Differential privacy

Differential privacy is a mathematical framework designed to protect data privacy while still enabling meaningful data analysis [35]. It achieves this by introducing random noise to mask individual data points, ensuring that the inclusion or exclusion of any single data point does not significantly affect the results of statistical analyses. This property makes it difficult for an attacker, even with access to the entire dataset, to infer private information about any specific user.

The privacy guarantee of differential privacy is controlled by a parameter ϵ. A smaller ϵ value corresponds to a higher level of added noise, thereby enhancing privacy protection at the cost of reduced utility [36]. Formally, differential privacy is defined as follows. A randomized mechanism $F : \mathcal{D} \to \mathcal{R}$ is said to be (ϵ, δ)-differentially private if, for any two neighboring datasets $D, D' \in \mathcal{D}$ differing in at most one sample and for any subset of outputs $R \subset \mathcal{R}$, the following holds:

$$P(F(D) \in R) \leqslant \mathrm{e}^{\epsilon} P(F(D') \in R) + \delta, \tag{3}$$

where ϵ is the privacy budget, controlling the level of privacy protection, and δ represents the probability of the condition failing. A smaller ϵ enforces stronger privacy by adding more noise but may reduce the accuracy of data analysis. The parameter δ accounts for the probability that ϵ-differential privacy is not strictly satisfied.

To integrate differential privacy into model training, differentially private stochastic gradient descent [37, 38] modifies the standard stochastic gradient descent algorithm by incorporating privacy-preserving mechanisms. Instead of directly averaging the gradients of a batch of samples, it first computes per-sample gradients, clips them to a predefined threshold, aggregates them into a batch gradient, and then adds Gaussian noise to the aggregated gradient. This process ensures that individual contributions remain obscured, thereby safeguarding the privacy of individual data points in the training dataset.

2.3 Tensor network

For high-order tensors, the number of elements grows exponentially with increasing dimensions, leading to the well-known "Curse of Dimensionality" [39]. Tensor networks provide a powerful computational framework widely used in physics, particularly in quantum many-body systems [40–42], and have found applications in fields such as machine learning [43] and computational biology [44]. These networks offer a structured approach to representing and manipulating large, complex quantum states by decomposing them into simpler, interconnected components called tensors.

Vectors and matrices can be regarded as special cases of tensors: a rank-0 tensor is a scalar, a rank-1 tensor is a vector, and a rank-2 tensor is a matrix. More generally, a rank-K tensor can be represented as $\mathcal{T}_{i_1,i_2,\ldots,i_K}$. Tensor networks can be visualized using diagrammatic representations, illustrating how tensors are contracted into a single structure.

The MPO is an efficient method for factorizing a high-dimensional tensor into several smaller, localized tensors [45]. Given a tensor M with $2n$ indices $[I_1, I_2, \ldots, I_n, J_1, J_2, \ldots, J_n]$, its MPO decomposition is expressed as

$$\mathrm{MPO}(M) = \sum_{d_1,d_2,\ldots,d_{n-1}} A^{j_1 d_1}_{i_1 d_0} A^{j_2 d_2}_{i_2 d_1} \cdots A^{j_n d_n}_{i_n d_{n-1}}, \tag{4}$$

where each tensor $A^{j_k d_k}_{i_k d_{k-1}}$ is a rank-4 tensor with dimensions $D_{k-1} \times I_k \times J_k \times D_k$. Here, the indices i_k and j_k correspond to the physical dimensions I_k and J_k, respectively, while D_{k-1} and D_k represent the bond dimensions that connect adjacent tensors. These bond dimensions, also known as MPO bond indices [46], establish interconnections between different layers in the tensor network.

As shown in Figure 1, each tensor in the decomposition is visually represented, with connecting legs indicating shared indices between tensors. By truncating the bond dimensions, one can obtain a low-rank approximation of the original tensor, significantly reducing computational complexity while preserving essential data features [33].

2.4 Quantum Byzantine agreement

The Byzantine agreement remains a fundamental challenge in distributed systems [47]. Classical Byzantine consensus protocols like PBFT exhibit inherent fault-tolerance limitations. Specifically, these protocols require a minimum of $3f + 1$ nodes to tolerate f Byzantine faults, rendering them inapplicable in

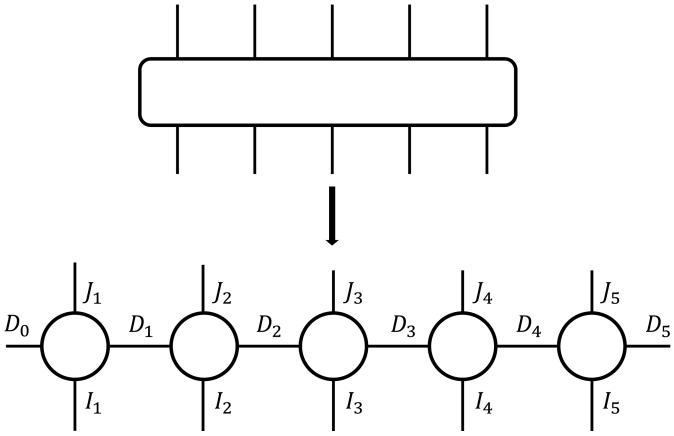

Figure 1 Diagram of MPO decomposition, illustrating the process of factorizing a large tensor into a product of five rank-4 tensors. Each tensor is graphically represented as a circle with four connecting legs.

scenarios with 3 participants where one of them is malicious. Furthermore, their reliance on classical cryptographic assumptions makes them vulnerable to quantum attacks. QBA overcomes these constraints by employing quantum cryptographic techniques to enhance both security and fault tolerance. Recent studies [31,32] demonstrate that QBA can surpass the classical $\frac{1}{3}$ fault-tolerance bound, achieving thresholds approaching $\frac{1}{2}$.

QBA provides two critical improvements compared to traditional Byzantine protocols. First, it achieves significantly higher fault tolerance through quantum digital signatures, extending the threshold to nearly $\frac{1}{2}$. This breakthrough ensures system consistency even when malicious nodes constitute almost half of the network. Second, the protocol enhances security via quantum digital signatures that guarantee anti-counterfeiting and non-repudiation properties. These features prevent message tampering or forgery by malicious clients, substantially strengthening the consensus process's resilience and reliability. The complete QBA protocol specification is detailed in Appendix A.

3 System design

3.1 Model assumptions

To ensure the robustness and efficiency of our proposed federated learning system, we establish the following assumptions.

Network structure. All clients form a fully connected network, allowing direct communication between any two participants. Each client not only serves as a model trainer but also functions as a blockchain node, responsible for maintaining blockchain data and actively participating in the consensus process. We assume that all clients possess sufficient quantum resources to engage in the consensus protocol effectively.

Clients. Participants in the federated learning framework exhibit diverse behaviors. Some clients are classified as honest-but-curious, meaning they adhere to the federated learning protocol while potentially attempting to infer private data. Conversely, certain clients may act maliciously, seeking to disrupt the system by submitting incorrect model updates or launching attacks during the consensus process.

Data distribution. The data held by clients are assumed to be non-independent and non-identically distributed (non-IID), reflecting real-world scenarios where variations arise due to differences in data collection methods, environments, or client behaviors [48,49]. This heterogeneity presents challenges for model convergence, necessitating the development of robust aggregation mechanisms.

3.2 Threat model

Our federated learning system is exposed to the following primary threats.

Data leakage. Malicious clients may attempt to infer private data from other participants by analyzing model updates, potentially exposing sensitive information about their training data.

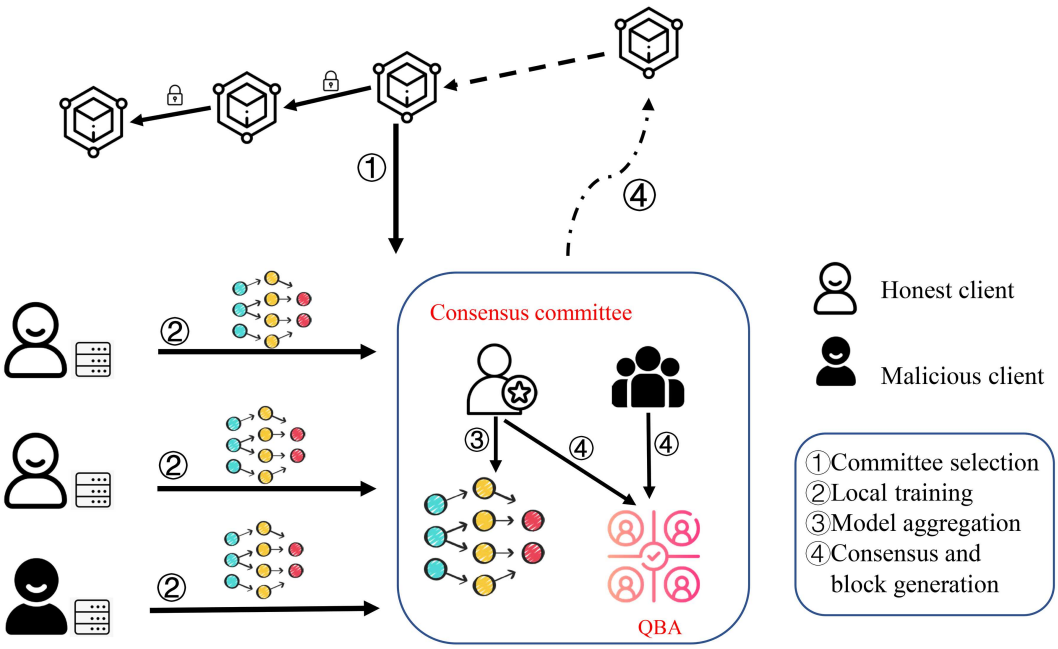

Figure 2 (Color online) Detailed workflow of the quantum-enhanced blockchain federated learning framework for each training round. After initialization, the system iterates through the four illustrated steps in a continuous loop until the predefined number of training rounds is reached. Some icons used in this figure are gratefully acknowledged from www.flaticon.com.

Model attacks. Adversarial clients may seek to compromise the training process by submitting manipulated model updates, including Bit-Flipping attacks and Label-Flipping attacks. Such adversarial behavior can degrade the accuracy and reliability of the aggregated global model.

Consensus manipulation. During the blockchain consensus process, malicious entities may inject false information or cast conflicting votes with the intent of disrupting consensus. These actions can undermine the integrity of the federated learning workflow and degrade overall system performance.

Quantum threat. The emergence of quantum algorithms presents a significant threat to conventional cryptographic systems. Encryption schemes commonly employed in blockchain-based federated learning, such as ECDSA and RSA, are highly vulnerable to quantum attacks [50–52]. A successful quantum attack could compromise security by enabling identity forgery and the manipulation of blockchain records, undermining the integrity and trustworthiness of the system.

4 System model

System overview. In this section, we present the general workflow of our proposed framework. Our protocol involves N clients, among which f may be malicious. The training process begins with the task publisher initializing the system by releasing the learning task along with a genesis block that contains the initial model parameters. After initialization, the system proceeds with multiple training rounds, each comprising four key steps, as illustrated in Figure 2.

(1) Committee selection. At the beginning of each training round, a consensus committee is selected from the participants to oversee the generation of a new block.

(2) Local training. Clients perform local training using model parameters derived from the most recent block. Upon completing their updates, they submit their locally trained models to the consensus committee for aggregation.

(3) Model aggregation. The leader aggregates the submitted model updates from the participants using a predefined aggregation algorithm to generate new global model parameters.

(4) Consensus and block generation. A consensus message is disseminated among committee members via QBA. A block that secures valid signatures from the majority of the committee members is considered legitimate and is subsequently broadcast to all blockchain participants.

Each block in the blockchain comprises several essential components, including global model updates, participant IDs contributing to the global model, committee member signatures, the previous block's hash

Algorithm 1 Stake-based committee selection algorithm.

Require: Hash of the latest block h, stake values $s(i)$ for each client i.
Ensure: Elected committee members 'committee'.

Client execution:
Sort all clients by their stake values $s(i)$ to generate the candidate list $C = \{c_1, c_2, \ldots, c_M\}$;
Compute the weight w_i for each candidate using (5);
Initialize an empty committee;
while len(committee) $< k$ **do**
  Update the block hash: $h \leftarrow \text{hash}(h)$;
  Generate a pseudorandom value $v = \frac{h}{2^{\text{len}(h)}} \in (0, 1]$;
  for each candidate $c_i \in C$ **do**
    if $w_{i-1} < v \leqslant w_i$ and $c_i \notin$ committee **then**
      Add c_i to committee;
    end if
  end for
end while
return committee.

linking it to its predecessor, and the hash of the current block. This structured design ensures model integrity and traceability throughout the training process. To mitigate quantum threats, our system integrates post-quantum cryptographic techniques. Specifically, we employ Blake3 as the blockchain's hashing algorithm and Dilithium [53] for hash-based identity authentication and digital signatures, providing robust resistance against quantum attacks.

4.1 Initialization

Before training begins, the task publisher releases the genesis block along with relevant information for model training. Each client generates its own public-private key pair for signing model updates and broadcasts the generated public key to the network. Each pair of clients performs key agreement through the quantum key distribution protocol to generate keys for the QBA protocol.

4.2 Committee selection

To enhance the efficiency, security, and fairness of federated learning, we propose a dynamic client role allocation protocol. In each training round, a committee consisting of N_c clients is elected. One member of the committee is designated as the leader, responsible for model aggregation and the generation of new blocks. The remaining committee members participate in the consensus process, overseeing the leader's aggregation to ensure correctness and integrity.

The committee election process is based on the stake held by each client. As shown in Algorithm 1, in each round, the top $2N_c$ clients ranked by stake are eligible for selection. If multiple clients have the same stake and this tie results in more than $2N_c$ clients being eligible, all tied clients will be included in the election process. Suppose there are a total of M clients, defined as $c_1, c_2, \ldots, c_M$, sorted by their IDs. The election weight of client i in the current round is computed as

$$w_i = \frac{\sum_{j=1}^{i} s(j)}{\sum_{j=1}^{M} s(j)}, \tag{5}$$

where $s(j)$ represents the stake of the j-th client. The computed weight w_i falls within the range $[0, 1]$, and we assume that $w_0 = 0$. A hash operation is applied to the hash of the latest block to generate a pseudorandom number v within the interval $(0, 1]$. The client i satisfying $w_{i-1} < v \leqslant w_i$ is selected as the leader. This process is iteratively executed using hash operations to select additional committee members. The first selected client will serve as the leader for this round.

During the system initialization phase, all clients begin with equal stake values. As training progresses, the system dynamically adjusts each client's stake based on their contributions: whenever a client's model is selected for global aggregation and recorded on the blockchain, their stake value increases. Since the blockchain maintains a transparent and immutable record of each aggregation, client contributions can be reliably tracked, eliminating the possibility of tampering or falsifying stake values. Honest clients will gradually accumulate higher stake values as their contributions increase, thereby gaining a dominant position in the committee.

4.3 Local training

In this work, we propose using a compressed model based on matrix product operator decomposition to reduce communication costs. Consider the weight parameter $\boldsymbol{w}$ of a linear layer with input size N_{in} and output size N_{out}. We first reshape $\boldsymbol{w}$ into a $2n$-dimensional tensor with shape $[I_1, I_2, \ldots, I_n, J_1, J_2, \ldots, J_n]$, where $\prod_{i=1}^{n} I_i = N_{\text{in}}$ and $\prod_{i=1}^{n} J_i = N_{\text{out}}$. By setting the MPO bond dimensions as $D_0 = D_n = 1$ and $D_1 = \cdots = D_{n-1} = \chi$, we perform MPO decomposition on this tensor to obtain its compressed representation, as shown in (4). This transformation allows the original linear layer to be replaced with an MPO layer, significantly reducing the number of trainable parameters.

Given an input $\boldsymbol{x} \in \mathbb{R}^{N_{\text{in}}}$, we reshape it into a tensor $\boldsymbol{x}_{I_1,I_2,\ldots,I_n}$ with shape $[I_1, I_2, \ldots, I_n]$. This reshaped input is then contracted with the MPO tensor to produce an intermediate output $\boldsymbol{y}_{J_1,J_2,\ldots,J_n}$ of shape $[J_1, J_2, \ldots, J_n]$. When compressing the linear layer into an MPO layer, the bias term is retained due to its relatively small number of parameters. Moreover, retaining the bias can help maintain or even improve model performance. After contraction, the bias is added to the output tensor, which is then passed to the subsequent layers. The parameters of the compressed MPO model are trained using standard optimization procedures via backpropagation. For convenience, let $\mathcal{T}^{J_1,J_2,\ldots,J_n}_{I_1,I_2,\ldots,I_n}(\chi)$ denote such an MPO layer.

The total number of trainable parameters in an MPO layer is given by

$$N_{\text{MPO}} = \prod_{k=1}^{n-1} I_k J_k \chi^2 + (I_0 J_0 + I_n J_n)\chi + N_{\text{out}}. \tag{6}$$

Compared to the original parameter size of $N_{\text{ori}} = (N_{\text{in}}+1) \times N_{\text{out}}$, this formulation enables efficient model compression. For instance, the original LeNet-5 model contains over 60000 parameters. By compressing the last three layers, we can obtain a reduced model with fewer than 5000 parameters, thereby significantly lowering communication costs.

To safeguard data privacy, each client incorporates differential privacy during training. Specifically, Gaussian noise is added to ensure (ϵ, δ)-differential privacy [54]. After completing the training process, the client signs the updated model parameters with its private key and submits them to the consensus committee for the current training round.

4.4 Model aggregation

The committee aggregates the model updates submitted by participating clients only after verifying their cryptographic signatures. To evaluate the quality of each local update, we employ a comprehensive metric that incorporates two critical factors: (1) the degree of alignment with the global update direction derived from the previous aggregation round and (2) the level of consistency with updates contributed by other clients in the current round [55, 56].

Suppose that in round t, the model update provided by client i is defined as $\boldsymbol{d}_i^t$. The first factor is quantified as

$$v_1(i) = \cos(\boldsymbol{d}_i^t, \overline{\boldsymbol{d}^{t-1}}) = \frac{\boldsymbol{d}_i^t \cdot \overline{\boldsymbol{d}^{t-1}}}{\|\boldsymbol{d}_i^t\| \|\overline{\boldsymbol{d}^{t-1}}\|}, \tag{7}$$

which measures the alignment between the gradient update $\boldsymbol{d}_i^t$ and the global update direction $\overline{\boldsymbol{d}^{t-1}}$ from the previous round.

For the second factor, we define

$$v_2(i) = \sum_{k \to i} \|\boldsymbol{d}_i^t - \boldsymbol{d}_k^t\|, \tag{8}$$

where $k \to i$ represents the set of the $N - f - 1$ closest model updates to i [56]. Honest model updates typically exhibit relatively small values for this metric, indicating their consistency with other reliable models.

The final weight of each model is determined by combining these two factors:

$$v(i) = \tilde{v}_1(i) - \tilde{v}_2(i), \tag{9}$$

Algorithm 2 Aggregation algorithm.

Require: Received local updates $\{\boldsymbol{d}_1^t, \boldsymbol{d}_2^t, \ldots, \boldsymbol{d}_N^t\}$ at round t, fault tolerance f, stake weights $s(i)$ for each client i, and the previous global direction $\overline{\boldsymbol{d}^{t-1}}$.
Ensure: Global aggregated model $\overline{\boldsymbol{d}^t}$.
1: **for** each update $\boldsymbol{d}_i^t$ **do**
2: $v_1(i) \leftarrow \cos(\boldsymbol{d}_i^t, \overline{\boldsymbol{d}^{t-1}}) = \frac{\boldsymbol{d}_i^t \cdot \overline{\boldsymbol{d}^{t-1}}}{\|\boldsymbol{d}_i^t\| \|\overline{\boldsymbol{d}^{t-1}}\|}$;
3: $v_2(i) \leftarrow \sum_{k \to i} \|\boldsymbol{d}_i^t - \boldsymbol{d}_k^t\|$;
4: **end for**
5: Calculate the value of each update using (9);
6: Select $\mathcal{S} = \{i \mid v(i)$ is among the largest $N - f - 2\}$;
7: Compute the aggregated model as a weighted average:

$$\overline{\boldsymbol{d}^t} \leftarrow \frac{\sum_{i \in \mathcal{S}} s(i) \boldsymbol{d}_i^t}{\sum_{i \in S} s(i)};$$

8: **return** $\overline{\boldsymbol{d}^t}$.

where $\tilde{v}_1(i) = \frac{v_1(i) - \mu_{v_1}}{\sigma_{v_1}}$ and $\tilde{v}_2(i) = \frac{v_2(i) - \mu_{v_2}}{\sigma_{v_2}}$ are the normalized values of the two metrics, with $\mu_{v_1}, \sigma_{v_1}, \mu_{v_2}, \sigma_{v_2}$ denoting the means and standard deviations across all candidates.

After computing $v(i)$ for each client's update, we select the $N - f - 2$ updates with the highest evaluation scores to form the aggregation set $\mathcal{S}$. This strategy ensures that both the alignment of each client's update with the global direction and the consistency among client updates are considered, thereby enhancing the model's robustness in Byzantine settings.

As shown in Algorithm 2, the final aggregation of the selected updates is performed using a stake-based weighted scheme. Let $s(i)$ denote the stake of client i. The aggregated global update $\overline{\boldsymbol{d}^t}$ is then computed as

$$\overline{\boldsymbol{d}^t} = \frac{\sum_{i \in \mathcal{S}} s(i) \boldsymbol{d}_i^t}{\sum_{i \in \mathcal{S}} s(i)}. \tag{10}$$

4.5 Consensus and block generation

The QBA is primarily composed of two phases: the broadcasting phase and the gathering phase. In a committee comprising a total of N_c clients, with at most f_c malicious actors, the broadcasting phase consists of f_c rounds of multicasts, during which committee members communicate multiple times using a quantum digital signature to exchange messages. In the gathering phase, committee members analyze the received messages to extract the consensus information intended to be broadcast by the leader or to determine if the broadcast has failed.

The leader is responsible for generating the consensus information based on the aggregation results. This consensus information includes the IDs of the selected clients and the hash of the aggregated update. The leader then initiates a consensus process on this information using the QBA protocol.

As shown in Algorithm 3, committee members can verify the consensus information provided by the leader using the updated parameters they have received. Here, we assume that all members of the committee receive consistent model updates. If the verification is successful, the member signs the message with their private key and broadcasts it. Conversely, if the QBA fails, the member takes no action, effectively rejecting the current round of consensus.

Eventually, the leader collects all signatures, and if more than half of the committee members provide valid signatures, the leader can issue a new candidate block, recording these signatures in the block. Once all clients verify the block, they update their local model weights and proceed to the next round of training. If a block fails to gain sufficient approval votes or if no valid block is received within a certain timeframe, honest clients will penalize the committee members for the current round by reducing their stake. Subsequently, a new committee will be selected, and the aggregation protocol will be reinitiated.

5 Results

5.1 Simulation setup

We implement our framework using Python 3.11, leveraging PyTorch for model training. To simulate communication between clients, we employ gRPC. We assume that clients have prepared the necessary

Algorithm 3 Consensus and block generation algorithm.

Require: Aggregated message 'msg', committee members 'committee', communication channel QBA.
Ensure: New block.

Leader executes:
Broadcast 'msg' to committee members via QBA;
Wait and collect signatures σ_i from committee members;
if $|\{\sigma_i\}| > \frac{N_c}{2}$ **then**
  Package all $\{\sigma_i\}$ into a new block;
  Broadcast the new block to all clients;
end if
Committee member executes:
for each committee member i **do**
  Generate msg_i based on Algorithm 2;
  if $\text{msg}_i == \text{msg}_{\text{rec}}$ **then**
    $\sigma_i \leftarrow \text{Sign}(\text{msg}_i, i)$;
    Send back σ_i;
  end if
end for
Client executes:
for each $i \in$ clients **do**
  if valid signatures $> \frac{N_c}{2}$ **then**
    $s(j) \leftarrow s(j) + 1$ for each j participating in model aggregation;
    Accept new block;
  else
    $s(j) \leftarrow s(j) - 1$ for j in consensus committee;
    Perform committee reconstruction;
  end if
end for

keys for QBA through a quantum key distribution protocol [57, 58]. Our experiments are conducted on an RTX 2080 Ti GPU.

Datasets. We utilize two widely recognized datasets in the field of federated learning: MNIST [59], a benchmark dataset for handwritten digit classification, and Fashion-MNIST [60], which consists of grayscale images of various clothing items. These datasets provide a comprehensive evaluation framework for assessing model performance across diverse visual recognition tasks.

To evaluate model accuracy in a non-IID setting, we simulate a scenario involving 50 clients by partitioning the data according to the methodology outlined in LotteryFL [61]. Each client is assigned data from all 10 classes, with 100 samples per class. The data is distributed in an unbalanced manner, maintaining an imbalance degree of 0.75. This partitioning strategy effectively captures realistic heterogeneity, closely resembling the conditions found in practical federated learning environments.

Models. To evaluate the generalizability and performance of our framework, we utilize two distinct model architectures. For the MNIST dataset, we use a fully connected neural network (FCNN), which is a simple two-layer architecture with a hidden layer containing 256 neurons. In contrast, for the Fashion-MNIST dataset, we adopt a more complex convolutional neural network (CNN) architecture that comprises two convolutional layers followed by three fully connected layers.

Malicious behavior. During the training process, malicious clients may engage in one of the following types of adversarial behavior.

• **Label-Flipping attack.** This attack involves intentionally altering the labels of training samples. Specifically, in the MNIST and Fashion-MNIST datasets, samples originally labeled as class i are maliciously reassigned to class $9 - i$.

• **Bit-Flipping attack.** In this attack, clients upload gradients with inverted signs, thereby transmitting updates that are deliberately opposed to the direction of correct gradient descent.

• **Gaussian attack.** Malicious clients add stochastic perturbations to their gradient updates by injecting Gaussian noise. In this study, the noise is sampled from a normal distribution with zero mean and a standard deviation of 0.1.

Additionally, malicious clients may target the consensus phase of model aggregation by casting opposing votes or generating incorrect aggregation results if they are selected as committee members or leaders.

5.2 Evaluation results

Compression efficiency. In this section, we evaluate the impact of the proposed model compression technique on test accuracy. We apply the MPO structure to compress the fully connected layers of the two models mentioned earlier, as these layers typically account for the majority of the model parameters.

Table 1 Detailed information on MPO-based model compression.

Model	Original layer	N_{ori}	MPO layer	N_{MPO}	Compression ratio $\eta = N_{\text{MPO}}/N_{\text{ori}}$ (%)
FCNN	Linear(784, 256)	200960	$\mathcal{T}^{4,4,4,4}_{4,7,7,4}(8)$	4096	2.0
	Linear(256, 10)	2570	$\mathcal{T}^{1,1,10,1}_{4,4,4,4}(4)$	746	29.0
CNN	Linear(400, 120)	48120	$\mathcal{T}^{2,2,5,3,2}_{2,2,5,10,2}(4)$	1096	2.3
	Linear(120, 84)	10164	$\mathcal{T}^{1,2,3,7,2}_{2,2,5,3,2}(4)$	748	7.4
	Linear(84, 10)	850	$\mathcal{T}^{1,1,2,5,1}_{1,2,3,7,2}(2)$	188	22.1

Table 2 Accuracy (%) comparison of two models in different settings.

Dataset	Scenario	Original	MPO
MNIST	Centralized	98.15	97.92
	Distributed	97.87	97.47
Fashion-MNIST	Centralized	89.20	88.92
	Distributed	89.18	88.55

Since the convolutional layers have relatively fewer parameters, we retain the original convolutional layers in the CNN model. Table 1 presents the compression results for the fully connected layers across different model architectures. Here, the compression ratio is defined as the number of parameters in the compressed MPO layer divided by the number of parameters in the original uncompressed layer. For example, in the FCNN model, a linear layer of size $[784, 256]$ is replaced by an MPO layer, $\mathcal{T}^{4,4,4,4}_{4,7,7,4}(8)$, which contains only 4096 parameters, achieving a compression ratio of 2.0%. Similarly, in the CNN model, the largest linear layer is compressed into an MPO layer $\mathcal{T}^{2,2,5,3,2}_{2,2,5,10,2}(4)$, achieving a compression ratio of 2.3%. Overall, the MPO structure reduces the total number of model parameters by 97.6% for the FCNN model and 92.5% for the CNN model, significantly decreasing the communication overhead among clients. The MPO framework also allows flexible tuning of the MPO layer parameters based on specific requirements, enabling a balance between compression efficiency and model accuracy. For large convolutional layers, MPO layers can be directly applied to replace them, or alternative tensor network methods can be used for parameter compression [62].

Table 2 compares the performance of the original and MPO-compressed models on the MNIST and Fashion-MNIST datasets under both centralized and distributed training scenarios. Despite the significant reduction in model parameters, the MPO-compressed models demonstrate only a slight decrease in accuracy. In the centralized scenario, the compressed model achieves an accuracy of 97.92% on MNIST, compared to 98.15% for the original model, reflecting a decrease of just 0.23%. A similar trend is observed for the Fashion-MNIST dataset, where the MPO-compressed model attains an accuracy of 88.92%, slightly lower than the 89.20% achieved by the original model. Under the distributed scenario, where the data distribution is non-IID across 50 clients, the performance of the MPO-compressed models remains competitive. For MNIST, the compressed model achieves an accuracy of 97.47%, only 0.40% lower than the original model. Similarly, on Fashion-MNIST, the compressed model attains an accuracy of 88.55%, reflecting a decrease of 0.63% compared to the original model. These results highlight the potential of MPO-based tensor networks as an effective compression technique for federated learning frameworks. By reducing communication burdens and computational costs, MPO compression facilitates the deployment of resource-efficient machine learning models on edge devices while maintaining high levels of accuracy.

Byzantine robustness. To evaluate the robustness of our model against Byzantine attacks, we test its accuracy under varying proportions of malicious clients. Specifically, the experiment involves a total of 50 clients, with the number of malicious clients gradually increasing from 10 to 20. The data allocated to each client is non-IID, and the configuration is consistent with that of the previous experiment.

We set the learning rate to 0.05 for the MNIST dataset and 0.02 for the Fashion-MNIST dataset. The optimizer used is SGD with a momentum of 0.9 and a weight decay of 1×10^{-4}. The batch size is set to 50. Each client trains for 1 epoch per round, with a total of 200 communication rounds for MNIST and 300 rounds for Fashion-MNIST. We compare our proposed method with several commonly used Byzantine-robust algorithms, including FedAvg, Trimmed-Mean, Median, and Krum. In all experiments, the compressed model based on the MPO is utilized. Notably, differential privacy mechanisms are not introduced during training to isolate and analyze the robustness of the compressed model under different attack scenarios.

Table 3 Average test accuracy (%) of the model under varying proportions of malicious clients. Compared to all other Byzantine-resilient algorithms, our model demonstrates the highest accuracy across both datasets. The best results are in bold.

Dataset	Method	Label-Flipping attack			Bit-Flipping attack			Gaussian attack		
		Malicious number			Malicious number			Malicious number		
		10	15	20	10	15	20	10	15	20
MNIST	FedAvg	88.09	76.67	58.57	95.60	92.94	75.25	95.60	92.73	92.42
	Trimmed-Mean	95.54	87.64	78.67	96.08	93.41	80.90	97.05	96.72	95.77
	Median	96.34	96.48	90.11	95.98	93.78	84.67	96.90	96.73	96.72
	Krum	78.60	82.72	88.72	73.35	85.26	78.70	78.58	79.96	81.15
	This work	**97.21**	**96.79**	**96.52**	**96.44**	**95.16**	**94.84**	**97.26**	**97.39**	**96.74**
Fashion-MNIST	FedAvg	80.59	64.56	52.32	84.18	78.69	62.13	83.00	80.21	73.78
	Trimmed-Mean	84.76	78.80	58.01	84.67	81.03	73.87	87.19	77.33	61.18
	Median	86.52	84.78	81.92	84.44	81.34	74.62	**87.30**	86.21	86.70
	Krum	76.78	77.67	73.85	68.52	64.83	70.11	76.25	76.38	77.17
	This work	**86.65**	**86.61**	**86.46**	**85.62**	**84.51**	**83.99**	87.22	**86.83**	**86.94**

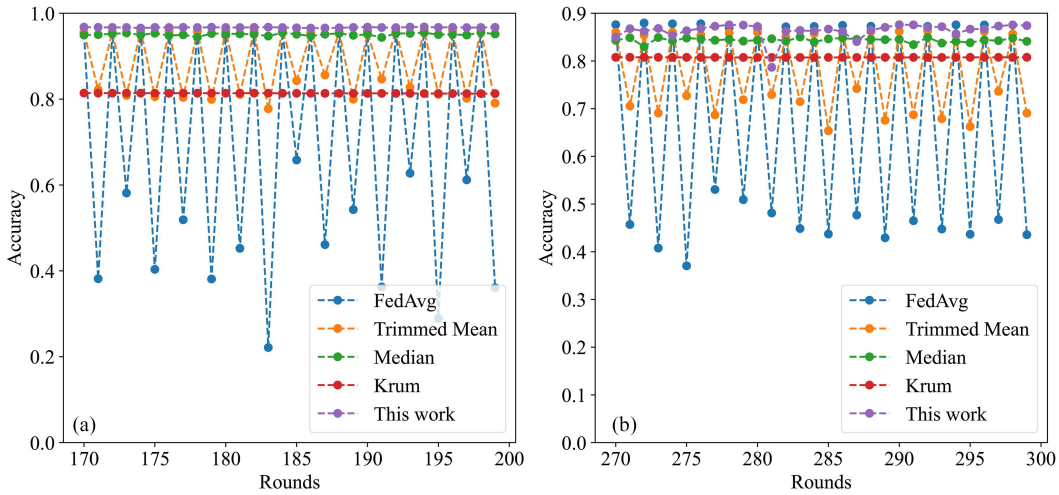

Figure 3 (Color online) Accuracy curves over the last 30 rounds under a 30% label-flipping attack for different methods: (a) MNIST; (b) Fashion MNIST. Our method demonstrates superior stability and robustness compared to the baselines.

As shown in Table 3, our proposed model demonstrates strong robustness against various types of adversarial attacks, as reflected by the average accuracy over the final 20 training rounds. Even when the proportion of malicious clients is relatively high, our approach consistently maintains high accuracy, significantly outperforming existing aggregation methods. Under Label-Flipping and Gaussian attacks, the model's accuracy declines by less than one percentage point as the number of malicious clients increases. Even under the more severe Bit-Flipping attack, the accuracy is reduced by only approximately two percentage points, further highlighting the robustness of our approach. In contrast, conventional approaches such as FedAvg, Median, and Trimmed-Mean perform reasonably well when the number of malicious clients is low but suffer a substantial decline in accuracy as the number of adversaries increases, with some methods even failing entirely under certain attack scenarios.

Figure 3 illustrates the performance of various aggregation methods over the final 30 training rounds under a Label-Flipping attack launched by 15 malicious clients. It can be observed that our proposed method demonstrates stable and superior robustness on both MNIST and Fashion-MNIST, maintaining high and consistent accuracy throughout the last 30 rounds. In contrast, FedAvg and Trimmed Mean exhibit significant instability under attack, with multiple sharp drops in accuracy. Although the Krum method remains stable, its overall accuracy is relatively low, indicating a substantial performance trade-off under high attack rates. The Median method shows moderate performance on both datasets but still falls short of ours in terms of stability and effectiveness.

Experimental results demonstrate that our method experiences only minimal accuracy degradation across all attack scenarios on both the MNIST and Fashion-MNIST datasets. By jointly considering two key factors the alignment between a client update and the previous global update, as well as the

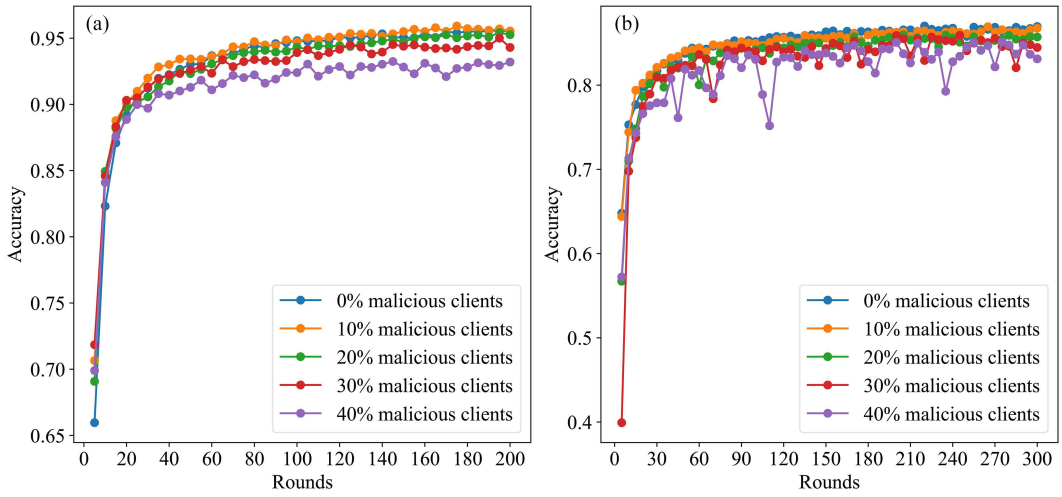

Figure 4 (Color online) Performance of the global model under differential privacy is evaluated by adding Gaussian noise during the training process to ensure privacy preservation. Our framework demonstrates strong accuracy and convergence across different proportions of malicious clients. (a) Accuracy training curves for the MNIST dataset; (b) corresponding results for the Fashion-MNIST dataset. Both highlight the model's robustness and effectiveness in maintaining high performance under varying levels of malicious interference.

consistency among clients our model effectively mitigates the influence of malicious actors through a stake-based weighted aggregation mechanism. This approach significantly enhances its ability to identify and counteract adversarial behavior.

Accuracy under differential privacy. To evaluate the robustness and convergence properties of our proposed model under differential privacy, we systematically analyze its performance with varying proportions of malicious clients. In each training round, Gaussian noise satisfying $(\varepsilon = 4, \delta = 10^{-5})$-differential privacy is added to protect the uploaded data. Malicious clients randomly select an attack strategy in each round, simulating diverse real-world adversarial scenarios. Additionally, to further assess the model's resilience to interference in distributed environments, malicious clients engage in adversarial behaviors during the consensus phase. As illustrated in Figure 4, the experimental results demonstrate that the model maintains strong convergence and robustness across different datasets, even in the presence of a high proportion of malicious clients. By introducing noise, client data privacy is effectively safeguarded. Despite increasing numbers of malicious clients and intensified differential privacy noise, the model consistently achieves high accuracy, validating the effectiveness of our approach in balancing privacy protection and model performance.

On the MNIST dataset, in the absence of malicious clients, the model attains a final accuracy of 95.59% after 200 training rounds. When the proportion of malicious clients increases to 40%, the final accuracy slightly decreases to 93.20%. A closer examination of the training process reveals that while a higher proportion of malicious clients marginally slows down convergence, the overall accuracy remains stable. Similarly, on the Fashion-MNIST dataset, the model demonstrates robust performance, achieving 86.96% accuracy in the absence of malicious clients and maintaining an accuracy of 83.68% even with 40% adversarial participants. Notably, across all scenarios, the model exhibits a consistent upward trend in accuracy during the initial training phase, followed by gradual convergence in later stages. The introduction of DP noise ensures a strong balance between data privacy protection and model performance, further demonstrating the model's ability to withstand complex attacks in distributed environments.

Time needed for consensus. In our protocol, the consensus message generated by the leader in each round comprises two key components: the hash of the aggregated model and the IDs of clients participating in the aggregation. The total size of this information is approximately 500 bits. The time required for each round of QBA depends on multiple factors, including message length, committee size, and the number of multicast rounds in the protocol.

To evaluate the efficiency and scalability of our approach, we measure the time required to complete a single round of QBA. Specifically, we conduct experiments to assess the average consensus time for message lengths of 500 and 1000 bits under different committee configurations, providing insights into the protocol's performance under varying conditions.

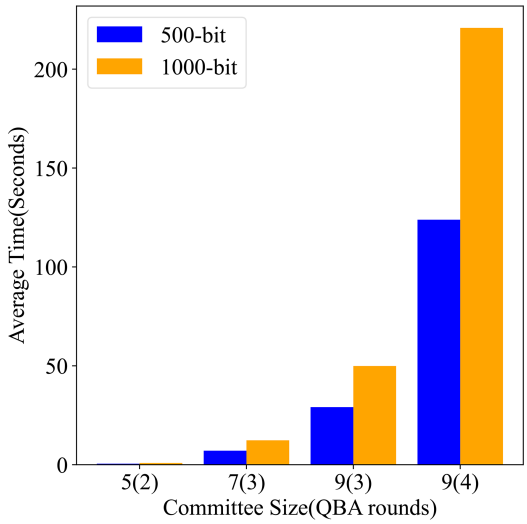

Figure 5 (Color online) Time consumption of QBA under different committee sizes. The horizontal axis represents the size of the consensus committee and the number of multicast rounds. The vertical axis indicates the average time required to complete a single QBA consensus instance.

As illustrated in Figure 5, the average time required to reach Byzantine consensus increases significantly with system size. For instance, with 5 clients and 2 malicious actors, the protocol requires an average of 0.51 s for a 500-bit message and 0.87 s for a 1000-bit message. However, as the system scales to 9 clients and 4 malicious participants, the consensus time rises sharply to 123.87 s for a 500-bit message and 220.86 s for a 1000-bit message. These results highlight the scalability challenges associated with the increased communication and computational complexity in quantum protocols. While the protocol remains efficient for smaller system sizes, the exponential growth in required quantum communication significantly extends the time needed to reach consensus in larger settings, a limitation we leave for future work to address.

6 Conclusion

In this work, we propose a quantum-enhanced blockchain federated learning framework that integrates a stake-based Byzantine-robust aggregation algorithm with the quantum Byzantine agreement protocol. This integration enables a federated learning system with high fault tolerance, maintaining high accuracy even in the presence of up to 40% malicious clients. Additionally, we employ a quantum-inspired tensor network structure to compress the model using matrix product operators, significantly reducing communication and computational costs. Experimental results demonstrate that our protocol is well-suited for distributed learning scenarios that involve adversarial clients and stringent privacy protection requirements.

However, our findings indicate that in multi-client scenarios, the quantum Byzantine agreement protocol incurs substantial computational overhead. This trade-off between efficiency and system scalability offers valuable insights into the limitations of quantum Byzantine agreement protocol in practical distributed systems and underscores the need for further optimization to enhance its applicability in large-scale deployments.

Acknowledgements This work was supported by National Natural Science Foundation of China (Grant No. 12274223), Fundamental Research Funds for the Central Universities and the Research Funds of Renmin University of China (Grant No. 24XNKJ14), and Program for Innovative Talents, Entrepreneurs in Jiangsu (Grant No. JSSCRC2021484).

References

1 McMahan B, Moore E, Ramage D, et al. Communication-efficient learning of deep networks from decentralized data. In: Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, 2017. 1273–1282
2 Li J, Meng Y, Ma L, et al. A federated learning based privacy-preserving smart healthcare system. IEEE Trans Ind Inf, 2022, 18: 2021–2031
3 Manias D M, Shami A. Making a case for federated learning in the Internet of vehicles and intelligent transportation systems. IEEE Netw, 2021, 35: 88–94
4 Khan L U, Saad W, Han Z, et al. Federated learning for Internet of Things: recent advances, taxonomy, and open challenges. IEEE Commun Surv Tutorials, 2021, 23: 1759–1799

5 Fang M H, Cao X Y, Jia J Y, et al. Local model poisoning attacks to Byzantine-robust federated learning. In: Proceedings of the 29th USENIX Conference on Security Symposium, 2020. 1623–1640
6 Bagdasaryan E, Veit A, Hua Y Q, et al. How to backdoor federated learning. In: Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics, 2020. 2938–2948
7 Yu K, Rong C, Wang H, et al. Federated local causal structure learning. Sci China Inf Sci, 2025, 68: 132105
8 Zhu J, Cao J, Saxena D, et al. Blockchain-empowered federated learning: challenges, solutions, and future directions. ACM Comput Surv, 2023, 55: 240
9 Issa W, Moustafa N, Turnbull B, et al. Blockchain-based federated learning for securing Internet of Things: a comprehensive survey. ACM Comput Surv, 2023, 55: 191
10 Nakamoto S. Bitcoin: a peer-to-peer electronic cash system. 2008. http://bitcoin.org/bitcoin.pdf
11 Castro M, Liskov B. Practical Byzantine fault tolerance. In: Proceedings of the 3rd Symposium on Operating Systems Design and Implementation, 1999. 173–186
12 Shayan M, Fung C, Yoon C J M, et al. Biscotti: a blockchain system for private and secure federated learning. IEEE Trans Parallel Distrib Syst, 2021, 32: 1513–1525
13 Li Y, Chen C, Liu N, et al. A blockchain-based decentralized federated learning framework with committee consensus. IEEE Netw, 2021, 35: 234–241
14 Zhang Y C, Zhang C, Zhang C, et al. Federated learning with quantum secure aggregation. 2022. ArXiv:2207.07444
15 Li W K, Lu S R, Deng D-L. Quantum federated learning through blind quantum computing. Sci China-Phys Mech Astron, 2021, 64: 100312
16 Kaewpuang R, Xu M R, Niyato D, et al. Adaptive resource allocation in quantum key distribution (QKD) for federated learning. In: Proceedings of International Conference on Computing, Networking and Communications, 2023. 71–76
17 Wang T, Li P, Wu Y, et al. Quantum-empowered federated learning in space-air-ground integrated networks. IEEE Netw, 2024, 38: 96–103
18 Quy V K, Quy N M, Hoai T T, et al. From federated learning to quantum federated learning for space-air-ground integrated networks. 2024. ArXiv:2411.01312
19 Zhou M G, Liu Z P, Yin H L, et al. Quantum neural network for quantum neural computing. Research, 2023, 6: 0134
20 Liu Z P, Cao X Y, Liu H W, et al. Practical quantum federated learning and its experimental demonstration. 2025. ArXiv:2501.12709
21 Song Y Q, Wu Y S, Wu S Y, et al. A quantum federated learning framework for classical clients. Sci China-Phys Mech Astron, 2024, 67: 250311
22 Xia Q, Li Q. QuantumFed: a federated learning framework for collaborative quantum training. In: Proceedings of IEEE Global Communications Conference, 2021. 1–6
23 Huang R, Tan X Q, Xu Q S. Quantum federated learning with decentralized data. IEEE J Sel Top Quantum Electron, 2022, 28: 1–10
24 Shi J J, Chen T, Zhang S C, et al. Personalized quantum federated learning for privacy image classification. 2024. ArXiv:2410.02547
25 Zhao H. Non-IID quantum federated learning with one-shot communication complexity. Quantum Mach Intell, 2023, 5: 3
26 Wang J T, Lian J Y, Ye T Y. Multi-party quantum private comparison of size relationship based on one-direction quantum walks on a circle. Sci China Inf Sci, 2025, 68: 129502
27 Xie Y M, Lu Y S, Weng C X, et al. Breaking the rate-loss bound of quantum key distribution with asynchronous two-photon interference. PRX Quantum, 2022, 3: 020315
28 Li C L, Yin H L, Chen Z B. Asynchronous quantum repeater using multiple quantum memory. Rep Prog Phys, 2024, 87: 127901
29 Fitzi M, Gottesman D, Hirt M, et al. Detectable Byzantine agreement secure against faulty majorities. In: Proceedings of the 21st Annual Symposium on Principles of Distributed Computing, 2002. 118–126
30 Smania M, Elhassan A M, Tavakoli A, et al. Experimental quantum multiparty communication protocols. npj Quantum Inf, 2016, 2: 16010
31 Weng C X, Gao R Q, Bao Y, et al. Beating the fault-tolerance bound and security loopholes for byzantine agreement with a quantum solution. Research, 2023, 6: 0272
32 Jing X, Qian C, Weng C X, et al. Experimental quantum Byzantine agreement on a three-user quantum network with integrated photonics. Sci Adv, 2024, 10: eadp2877
33 Gao Z F, Cheng S, He R Q, et al. Compressing deep neural networks by matrix product operators. Phys Rev Res, 2020, 2: 023300
34 Ji X, Liu Q, Huang S, et al. Image compression and reconstruction based on quantum network. In: Proceedings of IEEE International Parallel and Distributed Processing Symposium Workshops, 2024. 1128–1135
35 Dwork C. Differential privacy. In: Proceedings of the 33rd International Conference on Automata, Languages and Programming, 2006. 1–12
36 Dwork C, McSherry F, Nissim K, et al. Calibrating noise to sensitivity in private data analysis. In: Proceedings of the 3rd Conference on Theory of Cryptography, 2006. 265–284
37 Abadi M, Chu A, Goodfellow I, et al. Deep learning with differential privacy. In: Proceedings of the ACM SIGSAC Conference on Computer and Communications Security, 2016. 308–318
38 Bassily R, Smith A, Thakurta A. Private empirical risk minimization: efficient algorithms and tight error bounds. In: Proceedings of IEEE 55th Annual Symposium on Foundations of Computer Science, 2014. 464–473
39 Bishop C M. Pattern Recognition and Machine Learning. New York: Springer, 2006
40 Biamonte J, Bergholm V. Tensor networks in a nutshell. 2017. ArXiv:1708.00006
41 Huckle T, Waldherr K, Schulte-Herbrüggen T. Computations in quantum tensor networks. Linear Algebra its Appl, 2013, 438: 750–781
42 Choo K, Carleo G, Regnault N, et al. Symmetries and many-body excitations with neural-network quantum states. Phys Rev Lett, 2018, 121: 167204
43 Panagakis Y, Kossaifi J, Chrysos G G, et al. Tensor methods in computer vision and deep learning. Proc IEEE, 2021, 109: 863–890
44 Orús R. Tensor networks for complex quantum systems. Nat Rev Phys, 2019, 1: 538–550
45 Pirvu B, Murg V, Cirac J I, et al. Matrix product operator representations. New J Phys, 2010, 12: 025012
46 Hubig C, McCulloch I P, Schollwöck U. Generic construction of efficient matrix product operators. Phys Rev B, 2017, 95: 035129
47 Lamport L, Shostak R, Pease M. The Byzantine generals problem. ACM Trans Program Lang Syst, 1982, 4: 382–401
48 Zhao Y, Li M, Lai L Z, et al. Federated learning with non-IID data. 2018. ArXiv:1806.00582
49 Zhang Z, Zhang Y, Guo D, et al. Communication-efficient federated continual learning for distributed learning system with non-IID data. Sci China Inf Sci, 2023, 66: 122102
50 Aggarwal D, Brennen G K, Lee T, et al. Quantum attacks on Bitcoin, and how to protect against them. 2017. ArXiv:1710.10377

51 Gouzien É, Sangouard N. Factoring 2048-bit RSA integers in 177 days with 13436 qubits and a multimode memory. Phys Rev Lett, 2021, 127: 140503
52 Hong C L, Pei Z, Wang Q D, et al. Quantum attack on RSA by D-Wave advantage: a first break of 80-bit RSA. Sci China Inf Sci, 2025, 68: 129501
53 Ducas L, Kiltz E, Lepoint T, et al. Crystals-dilithium: a lattice-based digital signature scheme. IACR Trans Cryptogr Hardw Embed Syst, 2018, 2018: 238–268
54 Jiang Y, Tong X D, Liu Z Y, et al. Efficient federated unlearning with adaptive differential privacy preservation. 2024. ArXiv:2411.11044
55 Cao X Y, Fang M H, Liu J, et al. FLTrust: Byzantine-robust federated learning via trust bootstrapping. 2020. ArXiv:2012.13995
56 Blanchard P, Mhamdi E M E, Guerraoui R, et al. Machine learning with adversaries: Byzantine tolerant gradient descent. In: Proceedings of the 31st International Conference on Neural Information Processing Systems, 2017. 118–128
57 Yin H L, Liu P, Dai W W, et al. Experimental composable security decoy-state quantum key distribution using time-phase encoding. Opt Express, 2020, 28: 29479–29485
58 Zhou L, Lin J, Xie Y M, et al. Experimental quantum communication overcomes the rate-loss limit without global phase tracking. Phys Rev Lett, 2023, 130: 250801
59 Lecun Y, Bottou L, Bengio Y, et al. Gradient-based learning applied to document recognition. Proc IEEE, 1998, 86: 2278–2324
60 Xiao H, Rasul K, Vollgraf R. Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms. 2017. ArXiv:1708.07747
61 Li A, Sun J W, Wang B H, et al. LotteryFL: personalized and communication-efficient federated learning with lottery ticket hypothesis on non-IID datasets. 2020. ArXiv:2008.03371
62 Xie K, Liu C, Wang X, et al. Neural network compression based on tensor ring decomposition. IEEE Trans Neural Netw Learn Syst, 2025, 36: 5388–5402

Appendix A Quantum Byzantine agreement

In this section, we present the brief procedure of the quantum Byzantine agreement [31]. First, we present the protocol process of the three-party quantum digital signature, followed by an overview of the quantum Byzantine agreement protocol.

Appendix A.1 Quantum digital signature

Quantum digital signature (QDS)[1)2)3)] achieves its quantum advantage by utilizing a highly secure method based on principles of quantum mechanics, providing robust authentication resistant to quantum attacks. The QDS protocol typically involves three parties: the signer (Alice), the receiver (Bob), and the verifier (Charlie). In this protocol, Alice signs an m-bit document denoted as Doc, which Bob receives, and Charlie verifies to ensure its authenticity and integrity. The QDS process follows several crucial steps to secure the message and prevent both forgery and repudiation.

Pre-distribution state. Alice, Bob, and Charlie generate and share asymmetric quantum keys to ensure secure communication. They each have two key bit strings $\{X_a, X_b, X_c\}$ with n bits and $\{Y_a, Y_b, Y_c\}$ with $2n$ bits, where $X_a = X_b \oplus X_c$ and $Y_a = Y_b \oplus Y_c$, respectively.

Signing by Alice. Alice uses a quantum random number generator to produce a unique n-bit sequence p_a, where the polynomial representation of this sequence is an irreducible polynomial. She then uses the bit string X_a and p_a to generate a random linear feedback shift register-based Toeplitz matrix H_{nm}[4)]. Next, she computes an n-bit hash value of the m-bit document by calculating $\text{Hash} = H_{nm} \cdot \text{Doc}$. Following this, she constructs the $2n$-bit digest $\text{Dig} = (\text{Hash} \| p_a)$ using the hash value and the irreducible polynomial. Finally, she masks the digest with the bit string Y_a to create the signature: $\text{Sig} = \text{Dig} \oplus Y_a$, and sends the message $\{\text{Sig}, \text{Doc}\}$ to Bob.

Verification by Bob. Bob sends the received message $\{\text{Sig}, \text{Doc}\}$ along with his message bits $\{X_b, Y_b\}$ to Charlie. Charlie then sends his key bit strings X_c, Y_c back to Bob. Bob can reconstruct the keys that Alice owns using $K_{X_b} = X_b \oplus X_c$ and $K_{Y_b} = Y_b \oplus Y_c$. Bob utilizes K_{Y_b} to obtain the digest message received from Alice. Using p_a, K_{X_b}, and Doc, he repeats the operations that Alice performed to obtain a $2n$-bit actual digest. If the two digests match, he informs Charlie of the result; if not, he rejects the signature and aborts the protocol.

Verification by Charlie. If Bob announces his acceptance of the result, Charlie initiates the verification procedure. Similar to Bob, she reconstructs the key bit strings using $K_{X_c} = X_b \oplus X_c$ and $K_{Y_c} = Y_b \oplus Y_c$. Charlie then uses the keys she received to reconstruct the digest and checks if it matches the digest sent from Bob. If the two digests are identical, she accepts the signature.

Appendix A.2 Procedure of the quantum Byzantine agreement

The quantum Byzantine agreement mainly consists of two phases. The first phase is the broadcasting phase, where messages are passed among the clients through QDS. The second phase is the gathering phase, where committee members collect the messages received and derive the consensus message. Suppose there are N total members in the committee and f represents the number of malicious clients.

Broadcasting phase. The broadcasting phase consists of f rounds of multicast, corresponding to the number of malicious members. In the first multicast round, the leader acts as primary, and sequentially selects committee members i and j to act as forwarder and verifier, respectively, performing three-party QDS operations. During this round, a total

1) Yin H L, Fu Y, Li C L, et al. Experimental quantum secure network with digital signatures and encryption. Natl Sci Rev, 2023, 10: nwac228.
2) Li B H, Xie Y M, Cao X Y, et al. One-time universal hashing quantum digital signatures without perfect keys. Phys Rev Applied, 2023, 20: 044011.
3) Cao X Y, Li B H, Wang Y, et al. Experimental quantum e-commerce. Sci Adv, 2024, 10: eadk3258.
4) Krawczyk H. Lfsr-based hashing and authentication. In: Proceedings of the 14th Annual International Cryptology Conference on Advances in Cryptology, 1994. 129–139.

of A_{N-1}^2 QDS operations are performed. If the signature protocol is successfully verified, the forwarder and verifier record the received information from the primary.

In subsequent multicast rounds, each forwarder initiates its own multicast process, acting as the primary node and conducting QDS operations with those nodes that have not acted as primary in the message propagation path to propagate the message received from the primary node in the previous round. During this phase, the forwarder performs a consistency check to verify that the message received from the current primary matches the message received from the primary at the previous depth $d-1$. This iterative process continues for f rounds.

To achieve a higher fault tolerance limit, our protocol requires an increased number of three-party QDS operations. Specifically, for a system with N clients and f malicious clients, the total number of QDS operations is given by

$$N_{\mathrm{QDS}} = \sum_{m=0}^{f-1} A_{N-1}^{2+m}, \tag{A1}$$

where $A_a^b = \frac{a!}{(a-b)!}$ represents the number of b-permutations of a. This approach ensures robust consensus even in the presence of up to f malicious clients.

Gathering phase. The gathering phase begins once the broadcasting phase is complete. The derivation of messages is performed in reverse order. Assume that in the last multicast round, client j receives a list of messages from the primary path

$$j_{p_1} \to j_{p_2} \to \cdots \to j_{p_f},$$

defined as $L_{j_{p_1},\ldots,j_{p_f}}$. The consensus message for this round is computed using the majority function:

$$m_{j_{p_1},\ldots,j_{p_f}} = \mathrm{majority}(L_{j_{p_1},\ldots,j_{p_f}}), \tag{A2}$$

where the majority function selects the content that appears most frequently in the list.

For client j, different primary sequences may result in different messages. Here, we consider sequences where the first $f-1$ primary nodes are identical. In the f-th round, node j receives a series of messages from different primaries, denoted as $m_{j_{p_1},\ldots,j_k}$, where j_k iterates over all nodes satisfying $j_k \neq j$ and $j_k \notin \{j_{p_1}, j_{p_2}, \ldots, j_{p_{f-1}}\}$. Additionally, client j records the message he sends when acting as the primary in the f round as $m_{j_{p_1},\ldots,j_{p_{f-1}},j}$. These messages form the list:

$$L_{j_{p_1},\ldots,j_{p_{f-1}}} = \{\, m_{j_{p_1},\ldots,j_{p_{f-1}},j_k} \mid j_k \in J \setminus \{j_{p_1}, \ldots, j_{p_{f-1}}\} \,\}, \tag{A3}$$

where J represents the set of all committee members.

This process is applied recursively until a final list $L_{j_{p_1}}$ is obtained. The final QBA message is then determined by

$$m = \mathrm{majority}(L_{j_{p_1}}). \tag{A4}$$