B^{2}DVCS-FL: A Blockchain-Based Decentralized Verifiable Client Selection Algorithm for Federated Learning

Yuxin Jin*
yuxinjin8955@gmail.com
Shenzhen University
Shenzhen, China

ABSTRACT

Centralized Federated Learning (FL) relies on a central server, which is responsible for model aggregation, to drive the learning process. However, if the central server fails or behaves maliciously, the entire FL process will be interrupted. To address this issue, we integrate FL with blockchain technology to build a decentralized system, ensuring equal opportunities for each client to participate in model aggregation. To prevent the inclusion of underperforming local models, clients must submit local model metrics, and only those that meet specific criteria are allowed to participate in the aggregation. However, dishonest clients might attempt to gain an advantage by submitting inaccurately calculated metrics. To deal with this, we use Zero-Knowledge Proof (ZKP) technology combined with blockchain's tamper-resistant nature to enhance oversight and accountability in model selection processes.

Based on these challenges and solutions, we propose a Blockchain-Based Decentralized Verifiable Client Selection Federated Learning (B^{2}DVCS-FL) algorithm. This algorithm adopts a decentralized architecture in which non-committee clients use ZKP to compute local model performance metrics, enabling selection by committee members while enhancing trust in the selected models. Furthermore, the algorithm incorporates a committee-based consensus protocol to improve consensus efficiency, which is further strengthened by the properties of ZKP.

We conduct numerous experiments to validate the effectiveness and feasibility of the algorithm using two benchmarks. The first benchmark involves all clients with no noise, while the second involves random client selection for aggregation, with some clients introducing varying levels of noise to simulate malicious behaviors. The results show that our algorithm, through effective client selection, can maintain satisfactory global model performance. The practical use of ZKP not only ensures efficient performance, but also significantly reduces communication overhead during consensus. Compared to the random selection method, our algorithm continues to uphold acceptable global model performance even as the proportion of malicious clients increases.

*Yuxin Jin, email:yuxinjin8955@gmail.com

ISAICS '24, December 20-22, 2024, Changsha, China

ACM ISBN 978-1-4503-XXXX-X/18/06
https://doi.org/XXXXXXX.XXXXXXX

CCS CONCEPTS

• **Computing methodologies** → **Multi-agent systems**.

KEYWORDS

Federated Learning, Zero-Knowledge Proof, Blockchain, Committee-based Consensus, Selection Mechanism

ACM Reference Format:
Yuxin Jin. 2024. B^{2}DVCS-FL: A Blockchain-Based Decentralized Verifiable Client Selection Algorithm for Federated Learning. In *Proceedings of International Symposium on AI and Cybersecurity, December 20-22, 2024, Changsha, China (ISAICS '24).* ACM, New York, NY, USA, 10 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Machine Learning (ML) requires substantial data for optimal performance. However, acquiring these training samples is challenging due to privacy concerns, leading data owners to be reluctant to contribute their precious data. To address this issue, Google introduced the concept of Federated Learning (FL) [24]. FL breaks down data silos and maximizes the utility of data to reflect collective intelligence while protecting individual privacy.

In the FL paradigm, each participating client trains local models using their own data and uploads the updated model parameters (or gradients) to the central server for aggregation into a global model according to predefined rules. This iterative process continues until the global model achieves convergence or the process reaches a pre-set iteration threshold [2]. The entire FL framework relies on honest cooperation from all participating clients.

However, malicious clients may resort to unethical tactics and compromise the system by deliberately submitting local models that deviate from expected norms or injecting noise into the training samples. Or, malicious clients may falsify model performance metrics to ensure their local models participate in the model aggregation process. These malicious behaviors not only undermine the trust foundation in FL, but also significantly diminish the performance of the global model, weakening its accuracy and reliability. As a result, honest clients are less willing to participate in learning processes due to inaccurate predictions.

In addition, malicious models can destabilize the model training process. This can cause increased fluctuations or even divergence, preventing the model from converging to a stable solution. These actions not only increase training computational costs but also render the training process unpredictable, thereby heightening risks in model development.

Other than clients potentially engaging in malicious behaviors, the central server may also encounter unexpected issues. As in distributed systems, e.g. FL, single point of failure (SPOF) refers to a critical component failure that disrupts the operation of the entire system. In Centralized FL (CFL) architectures, there is always a central server that plays a crucial role in coordinating updated models from multiple participating clients [20], [17], [15], [1]. If the central server encounters problems due to hardware failure, network interruptions, or other unforeseen problems, the entire FL process may be severely disrupted, potentially leading to the termination of training activities. This is particularly dangerous in practical distributed applications as it can affect the continuous availability of the system and ultimately lead to serious incidents. Blockchain, as an effective alternative to a central server, is frequently employed in the design of FL systems to address SPOF and related issues [3], [31]. Without control by a central server, FL systems gain greater scalability and autonomy.

In this paper, we aim to optimize the architecture of FL and the consensus mechanism. By integrating blockchain technology into FL, we achieve a transition from the traditional centralized FL architecture to a decentralized one. This transition not only empowers all local training clients with equal status, ensuring equitable participation in the global model aggregation process, but also facilitates agreement among clients through an efficient consensus mechanism design. In the aggregation process, we no longer rely on simplistic full aggregation or random aggregation strategies, but instead require participating clients to submit their models' performance metrics calculated by ZKP technology. The use of ZKP enhances trust among clients and, due to its ability to generate small-sized proof files, significantly reduces communication overhead by avoiding the transfer of model parameters. However, the computational cost of generating and verifying proofs may vary depending on the specific implementation and the problem size. Leveraging the transparency and immutability of blockchain, this innovation provides a more dependable and trustworthy environment for FL, ensuring the fairness of model training and the efficacy of the results.

Our contributions:

- This paper introduces a blockchain-based decentralized verifiable client selection approach for FL, ensuring equal status for all system clients and enabling their participation in model aggregation operations. This design effectively mitigates single-point control.
- ZKP is integrated for lightweight proof document verification, facilitating efficient performance-based selection of local models. In our committee-based consensus algorithm, the committee votes on the ZKP of local model metrics and selects only the validated models with the best performance, rather than relying on broadcasting all local models for consensus. This approach significantly reduces communication overhead during the consensus process.
- Experimental validation confirms that our proposed algorithm, B^{2}DVCS-FL, improves the final global model's accuracy by 2.15%. Additionally, experiments demonstrate the lightweight nature of ZKP proof documents and verification time, establishing a foundation for the efficiency of the committee-based consensus mechanism.

2 BACKGROUND

This section provides background knowledge of Federated Learning (FL), Zero-Knowledge Proof (ZKP) and blockchain technologies. These foundational concepts are crucial for understanding the subsequent discussions and applications in this paper.

2.1 Federated Learning

Federated Learning (FL) [24] was originally proposed by Google to break down data silos while preserving privacy. In the current round t of FL, client C_i, $i \in [N]$, trains its local model g_i^t using its own dataset D_i, and then transmit g_i^t without exposing raw data, thereby protecting data privacy. These local models g_i^ts, $i \in [N]$, are aggregated to obtain a global model g_{global}^t. $g_{global}^t = \sum_i \frac{|D_i|}{\sum_i |D_i|} g_i^t$. The entire learning process iterates until the global model g_{global}^t converges or reaches a predefined number T of iterations.

In traditional FL algorithms, the learning processes often relies on a centralized architecture, requiring a central server S to perform the model aggregation. However, there are FL approaches that discard the need for a central server and achieve decentralized FL using Alternating Direction Method of Multipliers (ADMM) among multiple clients [13], [40]. Many studies also have explored various architectures in FL. Fedstellar [4] advances decentralized FL by supporting flexible federation setups and real-time performance monitoring. To address limitations in single-server FL, various alternative architectures have been proposed, including centralized, hierarchical, regional, and decentralized approaches, with comparative analyses of performance metrics such as latency, model evolution time, and classification accuracy [37].

2.2 Zero-Knowledge Proof

Privacy protection and trust issues often arise in multi-party collaboration scenarios, including FL. Zero-Knowledge Proof (ZKP) [9] technology is commonly introduced to enhance trust among participants.

ZKP technology typically involves two roles: the prover and the verifier. The prover aims to convince the verifier of the truth of a statement without revealing any useful information about it, while the verifier is to challenge the prover's claim and validate the proof's correctness [9], [11].

In the realm of ZKP technology, the Groth16 algorithm [10] has gained significant popularity for its efficiency. It allows for the creation of succinct proofs that are easy to verify, making it invaluable in scenarios with limited bandwidth or computational resources. Meanwhile, Groth16 supports non-interactive proof generation, eliminating the need for multiple rounds of communication between the prover and verifier.

Moreover, integrating ZKP with Machine Learning (ML) fosters trust in models. For example, the combination of ZKP and Convolutional Neural Networks (CNN) [22] enables proving to others that the prediction results on sample data are indeed computed by the model, without disclosing any information about the model itself.

2.3 Blockchain

Within the blockchain technology, decentralized ledgers have revolutionized digital transactions. A blockchain is a distributed and

immutable ledger where transactions are recorded in blocks linked together [27], [28]. Each block contains a cryptographic hash of the previous block, ensuring data integrity and tamper resistance. This structure enables transparent and secure transactions recording without the need for a central authority.

Blockchain employs consensus mechanisms to validate and agree on the state of the ledger across the network. Proof of Work (PoW) [27] and Proof of Stake (PoS) [16] are two prominent consensus algorithms. PoW involves miners solving complex mathematical puzzles to validate transactions and add blocks to the chain, while PoS relies on validators who stake their cryptocurrency to confirm transactions and secure the network. What's more, there are many other consensus protocols, such as Delegated Proof of Stake (DPoS) [36] and Practical Byzantine Fault Tolerance (PBFT) [6].

3 SYSTEM MODEL

Traditional CFL architectures rely on central servers to perform global model updates, driving the overall progress of the learning process. This section presents the system threats we aim to address in CFL, highlights the necessity of integrating blockchain architecture and ZKP technology, and introduces the system model of our proposed algorithm.

3.1 Threat Model and Related Solutions

1. Participation of Low-Quality Models: Selecting high-quality local models for aggregation is a matter worth attention. If the central server does not perform a filtering operation and instead aggregates local models of subpar quality, the global model will deviate from the expected results, thereby disturbing the entire learning process.

To enhance the efficiency and accuracy of FL, it is essential to establish a robust quality assessment mechanism and select the best local models for aggregation. In our proposed algorithm, each client is required to submit a metric indicating its model performance to facilitate the selection process.

2. Trust Issues among Participating Entities: In the current mechanism, each client submits a model metric to reflect the performance of its local model, aiming to rank higher and be selected for the global model update. However, some clients may act dishonestly by submitting false metrics to increase their chances of selection.

To prevent such dishonest behavior, we apply zero-knowledge proofs (ZKP) to the metric calculation process. By generating proofs for the metric calculation, ZKP enhances trust among participating entities regarding the authenticity of the model metrics, effectively reducing the likelihood of false submissions.

3. Single Point of Dependency: In CFL, the entire learning process is overly reliant on the coordination and functioning of the central server, creating a single point of dependency. If the central server fails or exhibits potentially dishonest behaviors, such as collusion with malicious clients, the entire FL process can be severely disrupted. Clients must trust the server's fairness and honesty to participate in the learning process, yet the transparency of such a system remains relatively low.

The decentralized nature of blockchain can effectively reduce over-reliance on a single entity within the system, ensuring equal standing for all participants. Therefore, introducing blockchain as the underlying architecture for FL offers a promising solution to enhance trust and resilience. Additionally, to improve the speed of achieving consensus within the system, we have also designed a committee-based consensus mechanism.

3.2 Decentralized System Model

We propose an algorithm called Blockchain-Based Decentralized Verifiable Client Selection Federated Learning (B^{2}DVCS-FL) to address the issues mentioned above. In our proposed algorithm, it is assumed that there are N clients C_i each with their private local training dataset D_i, $i \in \{1, 2, \cdots, N\}$. We consider the scenario where all devices participate, with T training rounds in total. In the t-th round, each client C_i trains a local model g_i^t using the stochastic gradient descent (SGD) method, $g_i^t = g_{global}^{t-1} - \eta_i \sum_{k=0}^{|D_i|} \nabla F(g_{global}^{t-1}; x_{i,k}, y_{i,k})$, where η_i is the learning rate, and $(x_{i,k}, y_{i,k})$ is the k-th sample in the client C_i's local dataset D_i. The committee member list is $L_{committee}^t$. Clients in $L_{committee}^t$ can aggregate high-performance local models to obtain a better global model, $g_{global}^t = \frac{1}{|D|} \sum_j |D_j| g_j^t$, where C_j is the j-th client that is allowed to participate in the aggregation, and $|D| = \sum_j |D_j|$, $|D_j|$ is the amount of data contributed by dataset D_j during the training. Uniformly, the subscript i or j represents the client to which it belongs, and the superscript t indicates the round of training.

3.3 Security Analysis Based on Zero-Knowledge Proof

Each client computes the cosine distance between its local model and the public benchmark model as its metric. To ensure the correctness of this computation, we employ Zero-Knowledge Proofs (ZKP) to verify that clients adhere to the prescribed computation rules while also protecting their privacy. The zero-knowledge, completeness, and soundness properties of our algorithm rely on the corresponding attributes of the ZKP protocol.

1. Zero-Knowledge: The ZKP protocol safeguards client privacy by preventing the disclosure of any specific information about the local model. While verifying that each client's metric computation conforms to the rules, the protocol ensures that privacy protection requirements are met.

2. Completeness: The completeness property of the ZKP protocol guarantees that honest clients who submit metrics in accordance with the rules will successfully pass verification. This ensures that honest clients are not unfairly excluded from the metric ranking due to verification failure, thus supporting fair participation.

3. Soundness: In the case of malicious clients, if they submit forged or non-compliant metrics, their proofs will fail to pass the verification. Consequently, the protocol effectively filters out clients that submit false metrics or do not compute the metric according to the specified rules, enhancing the overall reliability of the system.

4 ALGORITHM DESIGN

This section presents the key modules of our proposed algorithm, including the Decentralized Architecture Based on Blockchain, the Selection Mechanism, and the Committee-based Consensus Mechanism. It then provides an overview of the algorithm's process.

4.1 Decentralized Architecture Based on Blockchain

Due to the single-point of dependency inherent in traditional CFL algorithms aforementioned, we consider integrating blockchain technology into FL to form a decentralized architecture. Each client has an equitable opportunity to take part in the aggregation operation. The equality embodied by decentralization enhances the fairness of the system and weakens control over a single point. In contrast, centralized FL appears less fair in terms of the positions of the server and the clients.

With the integration of blockchain technology, each piece of local model information, which waits to be selected and verified, is stored as a "transaction" in a transaction pool. Each transaction records the necessary information about a local model, including the client name, the hash of the local model, the performance metrics of the local model, and the proof of metric computation. Each committee client will independently verify the transactions in the transaction pool and vote for those that meet the verification and selection criteria. If a transaction receives more votes than a certain threshold and its corresponding metric is sufficiently prominent, it will be chosen to aggregate for the global model. Subsequently, the committee generates a new block and stores it on the chain.

To accommodate the diversity of tasks and the potential variability in client trustworthiness across tasks, each task is recorded on its own blockchain. This task-specific approach simplifies the system's storage structure and decouples records between tasks, thus reducing overall system complexity.

Here, we use a single task as an example. Each block's data structure consists of a block header and a block body. The block header contains the following key fields that facilitate effective block management:

block_id: As the unique identifier of a block within the blockchain, it is used to mark the position of the block in the chain for indexing and querying purposes.

pre_hash: Record the hash value of the previous block, forming a chain structure in conjunction with the preceding blocks.

timestamp: Record the generation time of the current block.

task_name: Record the name of the current training task.

epoch: Record the training round of the current task.

The block header also needs to include other necessary fields. In addition, the block body contains the following key fields, mainly recording updates of local and global models in FL processes:

bench_model: Record the benchmark model, used for comparison with local models to calculate model metrics.

model_info: Record necessary information about local models, including client name, model hash, performance metric, and proof (see Fig. 1 for details). Specifically, **client_name** identifies the client, **hash** stores the model hash, **performance_metric** indicates model performance, and **proof** verifies the accuracy of performance metric calculation.

global_model: Record the global model obtained in the current round of the current task. This global model is aggregated from the local models trained by non-committee clients voted on by the committee.

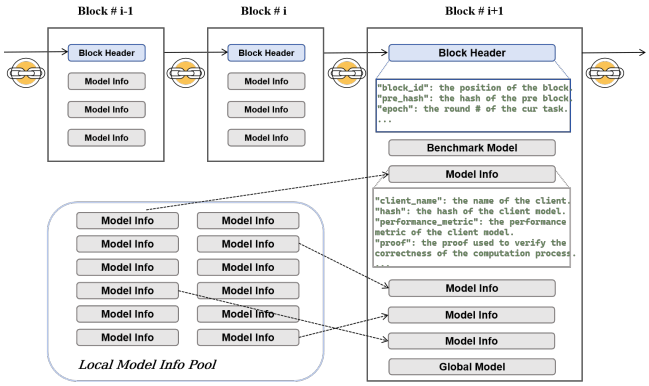

Figure 1: Data Structure of a Block in the Blockchain-Based Decentralized FL System

committee: Record the committee clients of the current round.

This kind of block's data structure allows legitimate clients to quickly verify whether a particular local model truly meets the metrics it has submitted, thereby increasing the transparency of the entire system. The block's data structure is displayed in Fig. 1.

Blockchain technology, often recognized as a distributed ledger, records updates of local and global models in FL. Data within the blockchain is structured as a series of blocks linked sequentially, forming an ever-expanding chain. Each subsequent block is generated based on the immediately preceding block. That is, each block contains the hash value of its immediate predecessor. Due to the collision resistance and sensitivity of hash calculations, any modification to a block will lead to a change in that block's hash value, thereby disrupting the integrity of the entire chain. This characteristic endows blockchain with a high degree of tamper resistance.

4.2 Selection Mechanism

Regarding the first point of the ***Subsection 3.1***, one of the challenges during model aggregation is to ensure the consolidation of high-quality local models while excluding low-quality ones. Thus, the system requires participating clients to submit performance metrics of their local models, utilizing cosine similarity [21], [26], [5] as the preferred metric. Cosine similarity measures the angle between the vectors representing the local model and the benchmark model, with values ranging from -1 to 1. Higher values, close to 1, indicate strong alignment with the benchmark, suggesting that the local model effectively captures the target patterns and is of high quality. Conversely, lower similarity values suggest greater deviation from the benchmark, potentially indicating lower quality. This similarity score is stored in the transaction data structure for each model, serving as a key reference for aggregation.

Of course, alternative metrics such as cross-entropy [8], [25] or Kullback-Leibler (KL) divergence [34], [39] can also be applied, offering flexibility for model selection under different evaluation criteria. Cross-entropy and KL divergence both serve to measure differences between probability distributions.

Without any form of verification, simply calculating performance metrics for local models is unlikely to convince other participants of the results' integrity. This lack of assurance can lead to doubts about the accuracy and honesty of reported metrics, potentially affecting the overall reliability of the model aggregation process. Therefore, we use ZKP technology during metric computation to strengthen trust within the system, ensuring that performance metrics are accurate and reducing the risk of fraudulent submissions.

Algorithm 1 Blockchain-Based Decentralized Verifiable Client Selection Federated Learning (B^{2}DVCS-FL)

Input: N clients C_i with their local dataset D_i,$\forall$ i $\in [N]$
Output: A global model g^T_{global}

1: The Blockchain System launches a new task T, initializes the model g^0_{global}, and initializes the committee list $L^1_{committee}$.
2: **for** iteration $t = 1, 2, ..., T$ **do**
3: **for all** client $C_i \in L^t_{committee}$ in parallel **do**
4: client C_i trains local model g^t_i based on D_i.
5: **end for**
6: $g^t_{benchmark} = \sum_{i=1}^{N} \frac{|D_i|}{\sum_{i=1}^{N}|D_i|} g^t_i$ as the benchmark model generated by the committee.
7: **for all** client $C_j \notin L^t_{committee}$ in parallel **do**
8: client C_j trains local model g^t_j based on D_j.
9: client C_j computes the metric cos^t_j between g^t_j and $g^t_{benchmark}$ using **ZKP** and generates the proof π^t_j.
10: **end for**
11: **for all** client $C_i \in L^t_{committee}$ in parallel **do**
12: client C_i verifies each proof π^t_j.
13: **if** π^t_j passes the verification of C_i **then**
14: C_i votes for the corresponding cos^t_j of π^t_j.
15: **end if**
16: **end for**
17: The committee sorts the verified cos^t_j values and selects the top k.
18: The Blockchain System updates next-round committee list $L^{t+1}_{committee}$ according to the voting results.
19: $g^t_{global} = \sum_{k=1}^{N} \frac{|D_k|}{\sum_{k=1}^{N}|D_k|} g^t_k$ for $C_k \in L^{t+1}_{committee}$.
20: **end for**
21: return g^T_{global}

4.3 Committee-based Consensus Mechanism

This subsection provides an in-depth exploration of the committee, a component previously mentioned, by elucidating its pivotal functions and its update process.

In the realm of FL combined with blockchain, the consensus mechanism is typically designed around a permissioned chain consensus algorithm. In the consensus process, all clients are predetermined. Only authenticated clients are allowed to join FL, thus ensuring the credibility of clients' identities. The consensus mechanism often involves the committee election [23], [38], [18] to guarantee efficient and secure transaction processing and recording. The elected committee plays a pivotal role, responsible for verifying transactions, proposing blocks, and handling disputes.

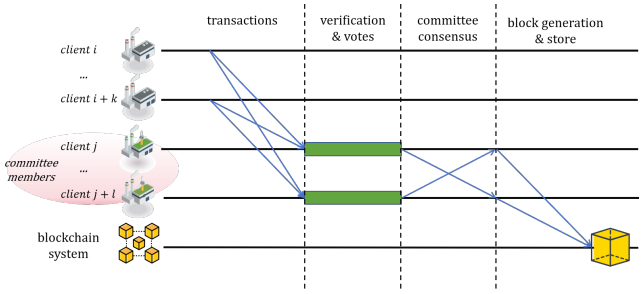

Figure 2: Flowchart of the Consensus-Reaching Process of Committee Clients

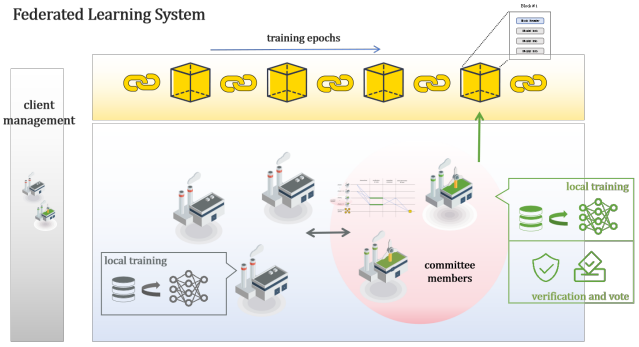

Figure 3: Overview of the Algorithm B^{2}DVCS-FL

We design a consensus algorithm specifically for our actual scenario, as illustrated in Fig. 2. During each training round, committee clients update their local models with their local datasets. The aggregated model from the committee acts as a benchmark to evaluate the updated local models from non-committee. Each non-committee client uses ZKP to calculate the performance metrics of its local model in comparison to the benchmark model, obtaining the proof document π.

Committee clients receive performance metrics and proof documents from non-committee clients. For those performance metrics that have proof documents passing a sufficient number of verifications, the committee collects the corresponding top-performance local models to form a global model for the current round, while generating a block. Clients who contribute to the global model in the current round will serve as the committee for the next round. The committee election process ensures that committee clients are ineligible for consecutive terms, thereby effectively prevents centralization and enhancing the security of the FL system. What's more, such a consensus mechanism has a certain degree of fault tolerance and security, ensuring the normal progress of the system network even if some clients (even some committee clients) fail or act maliciously.

The regular turnover of committee members ensures that the benchmark model is trained on a diverse set of sample datasets in each new round. This ongoing variation in the training data enhances the benchmark model's adaptability and generalization capabilities.

4.4 Algorithm Process Review

This subsection presents the overall process of B^{2}DVCS-FL, as depicted in Fig. 3 and Algorithm 1.

After being properly prepared, the system releases a training task and provides an initial model g^0_{global}. Clients prepare their own training data and resources. The system allows authorized clients C_i, $i \in [N]$ to join the training process and then the system initializes the initial committee $L^1_{committee}$. The process of local training and model aggregation will repeat until the global model converges or reaches the specified number T of iterations.

In the t-th round, $t = 1, 2, ..., T$, the operations are outlined as follows:

(1) Each committee client C_i (in $L^t_{committee}$) trains a local model g^t_i using their own local data, and these local models are aggregated to form a benchmark model $g^t_{benchmark}$.
(2) Each non-committee client C_j (not in $L^t_{committee}$) also utilizes their own data to update their own local models g^t_j.
(3) Each non-committee client C_j uses ZKP to compute the metrics cos^t_j of its local model g^t_j in comparison to $g^t_{benchmark}$ and then generates the proof π^t_j.
(4) Each committee client C_i (in $L^t_{committee}$) collects metrics and proofs. If C_i verifies a proof π^t_j, C_i will vote for π^t_j. If a proof π^t_j receives votes from more than half of the $L^t_{committee}$, the corresponding metric cos^t_j will be added to the list for ranking.
(5) The committee aggregates the top-k ranked local models to form the global model g^t_{global}. The clients C_k corresponding to the aggregated local models are selected as the new committee $L^{t+1}_{committee}$. $g^t_{global} = \sum_k \frac{|D_k|}{\sum_k |D_k|} g^t_k$.

5 EXPERIMENTS AND ANALYSIS

Based on the aforementioned design of our algorithm, we conduct the following experiments and analysis to assess the effectiveness of each module in B^{2}DVCS-FL, as follows.

5.1 Model Performance

We utilize the MNIST image dataset for model training and performance evaluation. To meet the requirements for subsequent ZKP processes, we preprocess the dataset by resizing and center cropping the images. This image preprocessing step is crucial for training models with a parameter count at the order of 10^3. In the FL settings, we configure the system with 20 participating clients.

To assess the effectiveness of our proposed algorithm, we first assume that all participating clients are honest and all their local datasets are free from noise, thus creating a best-case scenario for FL as a baseline for comparison. The global model is aggregated by randomly selecting 5 local models. This experimental setup aims to provide a clear understanding of the basic FL algorithm's performance in the absence of adversarial influences and data imperfections, serving as Baseline 1.

Furthermore, under otherwise identical experimental conditions, we add minor noise to some clients to simulate inevitable data collection imperfections. Additionally, we add more pronounced noise to some clients to simulate malicious behaviors initiated by

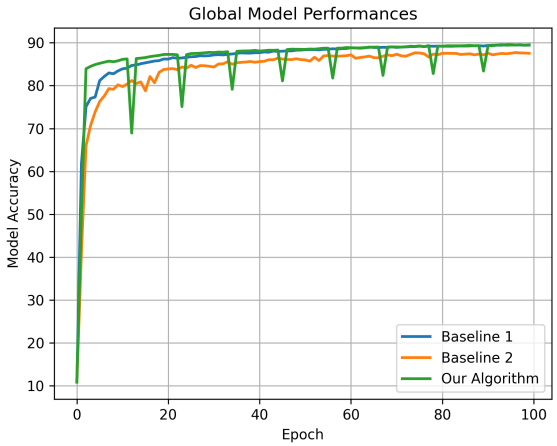

Figure 4: Global Model Performance Comparison in FL: Baseline 1, Baseline 2, and Proposed Algorithm

adversarial clients. The remaining clients are left without any noise. Similarly, the global model is aggregated by randomly selecting 5 local models. The results of FL based on these conditions serve as Baseline 2.

Finally, under conditions consistent with experimental settings in Baseline 2 (where some clients simulate malicious behaviors with pronounced noise similar to Baseline 2, and some have minor noise and the remainder no noise), we validate the effectiveness of our proposed algorithm in local model selection and consensus processes. Experimental results for these three scenarios are shown in the Fig. 4.

Based on the results depicted in the figure, under our experimental conditions, in the entirely ideal scenario free from noise and malicious behaviors, the global model achieves approximately 90% accuracy. However, under the second condition mentioned, the optimal performance of the global model decreases to around 87.85%. Evidently, the malicious behavior of adding noise to local datasets impacts the overall performance of the global model. Our algorithm, despite occasional periodic drops, achieves consistent model performance comparable to the ideal case. This underscores the effectiveness and necessity of local model selection mechanism. The occasional periodic declines in performance may be attributed to the consensus process.

5.2 ZKP Performance Validation

To accomplish the task of ZKP, we utilize version 2.0.0 of Circom to design and generate metric circuits. Circom offers an intuitive and efficient toolkit to describe the constraints and computational logic of the ZKP system using circuit files. Subsequently, we employ snarkJS, which is based on the Groth16 protocol for the bn128 elliptic curve, to generate and verify proofs, ensuring both security and efficiency. The experiments are conducted on an Ubuntu 20.04.6 LTS subsystem within the Windows operating system, with 15GB of system memory. The computer is equipped with an Intel 12th Generation Core i7-12700T processor.

ZKP technology plays a vital role in enhancing trust among clients, and the Groth16 protocol stands out for its efficient proof generation and verification mechanisms. Theoretically, the size of the circuits generated by Circom increases linearly with the size of

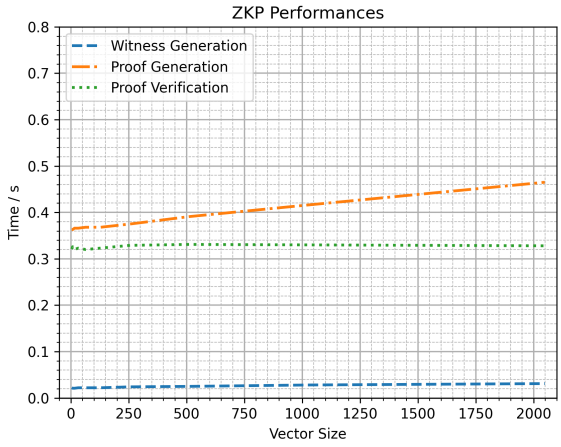

Figure 5: Impact of Vector Size on ZKP Performance: Witness Generation, Proof Generation and Proof Verification

the input data, which directly affects the time taken by the Groth16 protocol to generate witnesses and proofs. This linearity facilitates the prediction and planning of system resources. Moreover, the proof size of the Groth16 protocol is small and fixed, with verification time typically in the millisecond range, providing a solid technical foundation for real-time verification. Compared to traditional machine learning models, the Groth16 protocol significantly reduces the burden of data transfer while ensuring the lightweight nature of the proofs.

In our experiments, the theoretical advantages of the Groth16 protocol are fully validated and the experimental results are shown in Fig. 5 and Fig. 6. It is found that as the scale of the input data grows linearly, the time to generate witnesses and proofs also exhibits a linear growth trend. Specifically, the time to generate witnesses increases linearly from 20 to 30 milliseconds, while the time to generate proofs increases linearly from approximately 360 to 460 milliseconds. This finding indicates that the time growth for proof generation is predictable, providing valuable insights for performance evaluation and resource allocation in practical applications.

Furthermore, the verification time of the proofs remains around 330 milliseconds in the experiments, and the size of the proof documents is consistently around 805 bytes. These results not only correspond with the constant time for the proof verification process and the fixed size of the proofs in the Groth16 protocol, but also further demonstrate the efficiency and reliability of the Groth16 protocol in practical applications. This efficiency means that in scenarios requiring rapid verification and lightweight data exchange, the Groth16 protocol can offer significant performance advantages.

5.3 Communication Overhead in Consensus Protocols

Due to the traditional FL algorithm being centralized, it does not provide a consensus algorithm for comparison. Therefore, we utilize a broadcast consensus for comparative analysis with our committee-based consensus algorithm using ZKP, particularly focusing on the communication load.

Under the assumption that FL involves N fully connected clients, each client trains a local model with a uniform structure. The size of the model is denoted as S_{model}. We particularly focus on the communication overhead introduced by different consensus protocols, quantifying the amount of data transmitted during the communication process directly by document sizes. The voting information, whether consisting of a few characters or a more complex dictionary data structure, is relatively negligible in size compared to the models. Therefore, simply, we directly apply the size of the transmitted models as a direct measure of the communication overload.

In the broadcast consensus, after updating their local models in the t-th round, each client C_i sends its local model g_i^t to the remaining $N-1$ clients in the FL topology. The other $N-1$ clients can then evaluate and vote on the local model g_i^t from C_i. Each client performs this operation, sending its local model out and voting on other local models. With N clients in total, there are $N \cdot (N-1)$ model transmissions in total. In the best case, each local model only needs to be approved by a majority of the clients. Therefore, each local model should be sent to at least $\lceil N/2 \rceil$ clients. As a result, the total number of model transmissions in the best case is $N \cdot \lceil N/2 \rceil$. On average, the broadcast consensus requires $O(N^2)$ model transmissions.

In the consensus algorithm we propose, in the current FL round t, assuming there are x committee clients and y non-committee clients, such that the total is $x+y=N$. At the beginning of round t, we assume there is a leader C_{leader} within the committee. C_{leader} needs to aggregate the local models from committee clients to form a benchmark model $g_{benchmark}^t$, which involves $x-1$ model transmissions. Afterward, the committee sends $g_{benchmark}^t$ to y non-committee clients, resulting in y model transmissions. These y non-committee clients need to send their ZKP proofs to x committee clients for voting. The ZKP proof documents are computed by each non-committee client based on the cosine similarity between their local model and the benchmark model. The size of each proof document is fixed and denoted as S_{proof}. So, the communication load for proof documents is $x \cdot y \cdot S_{proof}$. Subsequently, according to the committee's voting results, non-committee clients, whose local model performance metrics rank in the top λ proportion, are required to transmit their local models to the committee for aggregation to obtain the global model g_{global}^t for round t. The updated global model g_{global}^t needs to be disseminated to all clients, resulting in model communications on the order of $O(N)$ transmissions. It should be noted that, in broadcast consensus, each client aggregates directly upon receiving the voting results from other clients, and the fastest client will produce the new block.

So, the total communication overhead of our consensus algorithm is $(x-1+y+y\cdot\lambda+N-1)\cdot S_{Model} + xy \cdot S_{proof} = (2N+y\cdot\lambda-2)\cdot S_{Model} + y(N-y)\cdot S_{proof}$.

We compare the communication overhead between broadcast consensus and our proposed consensus algorithm to assess the reduction in the communication overhead by our algorithm.

$$ratio = \frac{(2\cdot N + y\cdot\lambda - 2)\cdot S_{Model} + y\cdot(N-y)\cdot S_{proof}}{N^2 \cdot S_{Model}} \quad (1)$$

$$= \frac{(2\cdot N + y\cdot\lambda - 2)\cdot S_{Model}}{N^2 \cdot S_{Model}} + \frac{y\cdot(N-y)\cdot S_{proof}}{N^2 \cdot S_{Model}} \quad (2)$$

$$\leq \frac{2\cdot N + y\cdot\lambda}{N^2} + \frac{S_{proof}}{4\cdot S_{Model}} \leq \frac{3}{N} + \frac{S_{proof}}{4\cdot S_{Model}} \quad (3)$$

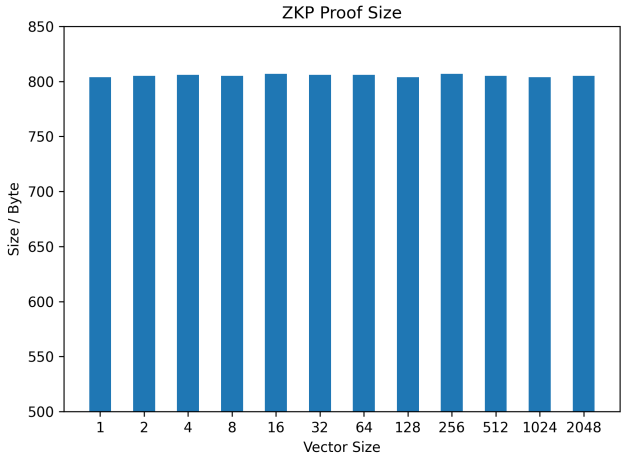

Figure 6: Impact of Vector Size on ZKP Proof Size

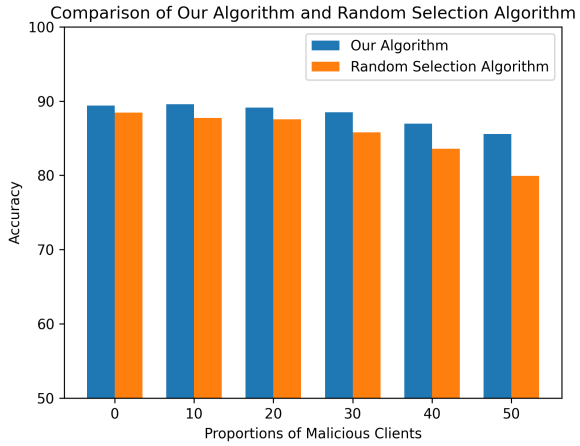

Figure 7: Effect of Malicious Client Proportion on Global Model Accuracy: Comparison of Our Algorithm and Random Selection

S_{proof} is nearly fixed. Therefore, the upper bound ratio of communication overhead between the two consensus algorithms is $O(\frac{1}{N} + \frac{1}{S_{model}})$. It is evident that the consensus algorithm is influenced by the scale of FL clients and model size. If the model size is large, then our consensus algorithm optimizes the communication overhead from the original broadcast consensus $O(N^2)$ to $O(N)$.

5.4 Malicious Client Tolerance Test

We conduct FL experiments with varying proportions of malicious clients to evaluate the accuracy of the global model. Following consistent experimental settings, we add minor noise to simulate unavoidable data collection imperfections and more pronounced noise to simulate malicious behaviors among clients, while some data remains free from noise. Different proportions of malicious clients are sequentially set for each comparative experiment with identical noise settings.

Comparative analysis in Fig. 7 reveals that our algorithm demonstrates notably improved performance compared to random selection. As the proportion of malicious clients increases, our algorithm maintains satisfactory accuracy of the global model. In contrast, under random selection, the global model's accuracy declines to below 80% when the proportion of malicious clients reaches 50%.

These findings underscore the effectiveness of our approach in mitigating the impact of malicious behaviors, ensuring robust model performance in federated learning scenarios. The experimental results further demonstrate the efficacy of our algorithm.

6 RELATED WORKS

In this section, we review the advanced approaches to FL: Blockchain-based FL (Subsection 6.1) and ZKP-based FL (Subsection 6.2). These methods not only expand the boundaries of traditional FL but also explore new possibilities in safeguarding data privacy and enhancing trust.

6.1 Federated Learning Based on Blockchain

Blockchain technology, with its unique decentralized characteristics, provides a new approach to solving the problems of centralization. It allows multiple participants to jointly maintain an immutable data ledger without a central authority. The application of this technology not only enhances the transparency and security of the system but also provides a viable alternative to the central server for FL [3], [31].

In the PBFL algorithm [26], the Solver trains a baseline model on a small clean dataset to identify Byzantine clients. PBFL employs CKKS [7] for secure aggregation of model updates. Additionally, it integrates a blockchain framework to ensure transparency in the aggregation process and enforce regulations, enhancing the overall robustness against malicious attacks. BFLC [18] embeds FL into the blockchain platform for storing model update information, while also designing an efficient consensus algorithm to avoid redundant and cumbersome consensus calculations.

Many blockchain-based FL algorithms [23], [38] are designed to use differential privacy to protect local models, use blockchain to store updated models, and include a consensus algorithm based on the quality inspection of training. In the consensus protocols of these FL algorithms, committee clients receive the local models from non-committee clients. Committee clients then use their own local datasets as verification sets to assess the prediction accuracy of each local model. Subsequently, the accuracy serves as a criterion for screening the next round of committee clients. As committee clients change, so do the corresponding verification sets, contrasting with [26], which uses a fixed verification dataset.

The consensus algorithms [18][23][38] diverge from broadcast consensus like PoW in Bitcoin. Instead, they focus on submitting information about local models solely to committee clients for screening and verification. This approach notably reduces communication load and achieves efficient consensus. However, the decreased client participation poses significant challenges for consensus. Balancing communication load and system security has thus become a critical concern in designing blockchain-based FL algorithms.

6.2 Federated Learning Based on Zero-Knowledge Proof

In the FL architectures, trust issue among clients (and with the server) is one of the core challenges. Dishonest behaviors can occur at the various stages of FL, affecting the performance of global models. To enhance the trust, researchers explore the integration of ZKP, a cutting-edge cryptographic technology, into FL.

Clients utilize ZKP technology to ensure the integrity and accuracy of the training process [35][12][32][30][29], demonstrating their commitment to using the necessary computing resources. Similarly, if the server employs ZKP to complete the model aggregation process [33], it demonstrates honesty and fairness to clients, further enhancing trust in the server. In addition, applying ZKP and blockchain in FL can verify the identities of the participating clients in the training process [19][14].

In summary, ZKP provides an effective solution to trust issues in FL, fostering a more secure, transparent, and trustworthy distributed learning environment.

7 CONCLUSION AND DISCUSSION

Conclusion: To address the inherent concerns of centralization in FL, we propose B^{2}DVCS-FL to reduce single-point dependency and enhance trust. The B^{2}DVCS-FL algorithm utilizes blockchain architecture to ensure equal opportunities for clients to contribute to the model aggregation process, while incentivizing clients to provide high-performance models. However, achieving consensus among clients can be challenging, especially when dealing with large models. If applying broadcast consensus, there will be significant communication overhead. Thus, we design a committee-based consensus protocol. The committee, composed of trusted clients, selects the local models which to aggregate based on model metrics submitted by clients each round.

Furthermore, by integrating ZKP, we enhance the committee's trust in model metrics. The lightweight proof documents generated by ZKP can further reduce communication overhead during consensus. The generation and verification of proof documents maintaining millisecond-level efficiency further demonstrate the rationale of our proposed algorithm.

We conduct experiments to validate the effectiveness of our algorithm and demonstrate significant improvements in global model performance through selective aggregation of local models based on performance metrics. We use Circom and snarkjs to verify the properties of ZKP. Under our experimental conditions, the time taken for proof generation and verification aligns closely with theoretical expectations. The proof documents maintain a consistently small size compared to the local model size. With the committee clients representing only a small fraction of all clients, communication overhead significantly decreases during consensus. Furthermore, as the proportion of malicious clients increases, our proposed algorithm indeed outperforms conventional random selection in achieving better global model performance.

Discussion: This paper focuses on applying ZKP to the calculation of local model performance metrics for selection in decentralized FL systems. While enhancing the trust among clients, the features of ZKP and the selection mechanism also reduce the transmission load.

Additionally, one might question how our algorithm ensures that the submitted models meet the claimed performance metrics. This issue has been discussed in the previous sections. Due to the immutable nature of the blockchain, if a local client C_i correctly computes the performance metrics but submits a local model $\widehat{g}_i^t$ (rather than g_i^t) that does not meet the stated performance, attempting to interfere with the learning process, the model $\widehat{g}_i^t$, once aggregated into the final global model, will be permanently recorded on the blockchain. Legitimate clients, when verifying the blocks, will discover this malicious behavior, severely impacting trust in the client C_i. If the client C_i attempts to alter its submitted data after aggregation, C_i would need to alter all subsequent blocks, which would consume a significant amount of resources.

This mechanism not only enhances security and reliability but also strengthens the monitoring of client behaviors, ensuring the fairness and effectiveness of the overall system.

REFERENCES

[1] Sawsan AbdulRahman, Hanine Tout, Hakima Ould-Slimane, Azzam Mourad, Chamseddine Talhi, and Mohsen Guizani. 2020. A survey on federated learning: The journey from centralized to distributed on-site learning and beyond. *IEEE Internet of Things Journal* 8, 7 (2020), 5476–5497.

[2] Mohammed Aledhari, Rehma Razzak, Reza M Parizi, and Fahad Saeed. 2020. Federated learning: A survey on enabling technologies, protocols, and applications. *IEEE Access* 8 (2020), 140699–140725.

[3] Mansoor Ali, Hadis Karimipour, and Muhammad Tariq. 2021. Integration of blockchain and federated learning for Internet of Things: Recent advances and future challenges. *Computers & Security* 108 (2021), 102355.

[4] Enrique Tomás Martínez Beltrán, Ángel Luis Perales Gómez, Chao Feng, Pedro Miguel Sánchez Sánchez, Sergio López Bernal, Gérôme Bovet, Manuel Gil Pérez, Gregorio Martínez Pérez, and Alberto Huertas Celdrán. 2024. Fedstellar: A platform for decentralized federated learning. *Expert Systems with Applications* 242 (2024), 122861.

[5] Xiaoyu Cao, Minghong Fang, Jia Liu, and Neil Zhenqiang Gong. 2020. Fltrust: Byzantine-robust federated learning via trust bootstrapping. *arXiv preprint arXiv:2012.13995* (2020).

[6] Miguel Castro, Barbara Liskov, et al. 1999. Practical byzantine fault tolerance. In *OsDI*, Vol. 99. 173–186.

[7] Jung Hee Cheon, Andrey Kim, Miran Kim, and Yongsoo Song. 2017. Homomorphic encryption for arithmetic of approximate numbers. In *Advances in Cryptology–ASIACRYPT 2017: 23rd International Conference on the Theory and Applications of Cryptology and Information Security, Hong Kong, China, December 3-7, 2017, Proceedings, Part I 23*. Springer, 409–437.

[8] Xiuwen Fang and Mang Ye. 2022. Robust federated learning with noisy and heterogeneous clients. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 10072–10081.

[9] Shafi Goldwasser, Silvio Micali, and Chales Rackoff. 2019. The knowledge complexity of interactive proof-systems. In *Providing sound foundations for cryptography: On the work of shafi goldwasser and silvio micali*. 203–225.

[10] Jens Groth. 2016. On the size of pairing-based non-interactive arguments. In *Advances in Cryptology–EUROCRYPT 2016: 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part II 35*. Springer, 305–326.

[11] Jahid Hasan. 2019. Overview and applications of zero knowledge proof (ZKP). *International Journal of Computer Science and Network* 8, 5 (2019), 2277–5420.

[12] Jonathan Heiss, Elias Grünewald, Stefan Tai, Nikolas Haimerl, and Stefan Schulte. 2022. Advancing blockchain-based federated learning through verifiable off-chain computations. In *2022 IEEE International Conference on Blockchain (Blockchain)*. IEEE, 194–201.

[13] Beomyeol Jeon, SM Ferdous, Muntasir Raihan Rahman, and Anwar Walid. 2021. Privacy-preserving decentralized aggregation for federated learning. In *IEEE INFOCOM 2021-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE, 1–6.

[14] Shan Ji, Jiale Zhang, Yongjing Zhang, Zhaoyang Han, and Chuan Ma. 2023. LAFED: A lightweight authentication mechanism for blockchain-enabled federated learning system. *Future Generation Computer Systems* 145 (2023), 56–67.

[15] Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. 2020. Scaffold: Stochastic controlled averaging for federated learning. In *International conference on machine learning*. PMLR, 5132–5143.

[16] Sunny King and Scott Nadal. 2012. Ppcoin: Peer-to-peer crypto-currency with proof-of-stake. *self-published paper, August* 19, 1 (2012).

[17] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. 2020. Federated optimization in heterogeneous networks. *Proceedings of Machine learning and systems* 2 (2020), 429–450.

[18] Yuzheng Li, Chuan Chen, Nan Liu, Huawei Huang, Zibin Zheng, and Qiang Yan. 2020. A blockchain-based decentralized federated learning framework with committee consensus. *IEEE Network* 35, 1 (2020), 234–241.

[19] Yijing Li, Xiaofeng Tao, Xuefei Zhang, Junjie Liu, and Jin Xu. 2021. Privacy-preserved federated learning for autonomous driving. *IEEE Transactions on Intelligent Transportation Systems* 23, 7 (2021), 8423–8434.

[20] Zengpeng Li, Vishal Sharma, and Saraju P Mohanty. 2020. Preserving data privacy via federated learning: Challenges and solutions. *IEEE Consumer Electronics Magazine* 9, 3 (2020), 8–16.

[21] Yihuai Liang, Yan Li, and Byeong-Seok Shin. 2023. Auditable federated learning with byzantine robustness. *IEEE Transactions on Computational Social Systems* (2023).

[22] Tianyi Liu, Xiang Xie, and Yupeng Zhang. 2021. Zkcnn: Zero knowledge proofs for convolutional neural network predictions and accuracy. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*. 2968–2985.

[23] Yunlong Lu, Xiaohong Huang, Yueyue Dai, Sabita Maharjan, and Yan Zhang. 2019. Blockchain and federated learning for privacy-preserved data sharing in industrial IoT. *IEEE Transactions on Industrial Informatics* 16, 6 (2019), 4177–4186.

[24] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*. PMLR, 1273–1282.

[25] Jiaxu Miao, Zongxin Yang, Leilei Fan, and Yi Yang. 2023. Fedseg: Class-heterogeneous federated learning for semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 8042–8052.

[26] Yinbin Miao, Ziteng Liu, Hongwei Li, Kim-Kwang Raymond Choo, and Robert H Deng. 2022. Privacy-preserving Byzantine-robust federated learning via blockchain systems. *IEEE Transactions on Information Forensics and Security* 17 (2022), 2848–2861.

[27] Satoshi Nakamoto. 2008. Bitcoin: A peer-to-peer electronic cash system. (2008).

[28] Christopher Natoli, Jiangshan Yu, Vincent Gramoli, and Paulo Esteves-Verissimo. 2019. Deconstructing blockchains: A comprehensive survey on consensus, membership and structure. *arXiv preprint arXiv:1908.08316* (2019).

[29] Timon Rückel, Johannes Sedlmeir, and Peter Hofmann. 2022. Fairness, integrity, and privacy in a scalable blockchain-based federated learning system. *Computer Networks* 202 (2022), 108621.

[30] Abla Smahi, Hui Li, Yong Yang, Xin Yang, Ping Lu, Yong Zhong, and Caifu Liu. 2023. BV-ICVs: A privacy-preserving and verifiable federated learning framework for V2X environments using blockchain and zkSNARKs. *Journal of King Saud University-Computer and Information Sciences* 35, 6 (2023), 101542.

[31] Ke Wang, Chien-Ming Chen, Zuodong Liang, Mohammad Mehedi Hassan, Giuseppe ML Sarne, Lidia Fotia, and Giancarlo Fortino. 2021. A trusted consensus fusion scheme for decentralized collaborated learning in massive IoT domain. *Information Fusion* 72 (2021), 100–109.

[32] Lingling Wang, Xueqin Zhao, Zhongkai Lu, Lin Wang, and Shouxun Zhang. 2023. Enhancing privacy preservation and trustworthiness for decentralized federated learning. *Information Sciences* 628 (2023), 449–468.

[33] Zhipeng Wang, Nanqing Dong, Jiahao Sun, William Knottenbelt, and Yike Guo. 2024. zkfl: Zero-knowledge proof-based gradient aggregation for federated learning. *IEEE Transactions on Big Data* (2024).

[34] Zhijie Xie and Shenghui Song. 2023. FedKL: Tackling data heterogeneity in federated reinforcement learning by penalizing KL divergence. *IEEE Journal on Selected Areas in Communications* 41, 4 (2023), 1227–1242.

[35] Zhibo Xing, Zijian Zhang, Meng Li, Jiamou Liu, Liehuang Zhu, Giovanni Russello, and Muhammad Rizwan Asghar. 2023. Zero-knowledge proof-based practical federated learning on blockchain. *arXiv preprint arXiv:2304.05590* (2023).

[36] Guangxia Xu, Yong Liu, and Prince Waqas Khan. 2019. Improvement of the DPoS consensus mechanism in blockchain based on vague sets. *IEEE Transactions on Industrial Informatics* 16, 6 (2019), 4252–4259.

[37] Hongyi Zhang, Jan Bosch, and Helena Holmström Olsson. 2020. Federated learning systems: Architecture alternatives. In *2020 27th Asia-Pacific Software Engineering Conference (APSEC)*. IEEE, 385–394.

[38] Shuxin Zhang and Jinghua Zhu. 2023. Privacy protection federated learning framework based on blockchain and committee consensus in IoT devices. In *2023 IEEE 47th Annual Computers, Software, and Applications Conference (COMPSAC)*. IEEE, 627–636.

[39] Xu Zhang, Yinchuan Li, Wenpeng Li, Kaiyang Guo, and Yunfeng Shao. 2022. Personalized federated learning via variational bayesian inference. In *International Conference on Machine Learning*. PMLR, 26293–26310.

[40] Shenglong Zhou and Geoffrey Ye Li. 2021. Communication-efficient ADMM-based federated learning. *arXiv preprint arXiv:2110.15318* (2021).