SCEI: A Smart-Contract Driven Edge Intelligence Framework for IoT Systems

Chenhao Xu, Jiaqi Ge, Yong Li, Yao Deng, Longxiang Gao, *Senior Member, IEEE*, Mengshi Zhang, Yong Xiang, *Senior Member, IEEE*, and Xi Zheng, *Member, IEEE*

***Abstract*—Federated learning (FL) enables collaborative training of a shared model on edge devices while maintaining data privacy. FL is effective when dealing with independent and identically distributed (IID) datasets, but struggles with non-iid datasets. Various personalized approaches have been proposed, but such approaches fail to handle underlying shifts in data distribution, such as data distribution skew commonly observed in real-world scenarios (e.g., driver behavior in smart transportation systems changing across time and location). Additionally, trust concerns among unacquainted devices and security concerns with the centralized aggregator pose additional challenges. To address these challenges, this paper presents a dynamically optimized personal deep learning scheme based on blockchain and federated learning. Specifically, the innovative smart contract implemented in the blockchain allows distributed edge devices to reach a consensus on the optimal weights of personalized models. Experimental evaluations using multiple models and real-world datasets demonstrate that the proposed scheme achieves higher accuracy and faster convergence compared to traditional federated and personalized learning approaches.**

***Index Terms*—Blockchain, deep learning, distributed learning, federated learning, IoT, personalized model, smart contract.**

I. Introduction

COMPANIES are often hesitant to share their business data with others due to commercial competition and privacy concerns. As a result, an approach that enables collaborators to collectively train a shared model while retaining complete control over their local data becomes crucial to meet various real-world demands. For example, in smart hospital systems, accurate recognition of medical test results, such as blood tests, X-ray tests, and ultrasound tests, heavily relies on image-based classifiers [1]. However, the limited amount of local data collected from patients and cases at an individual clinic is inadequate for training a deep-learning model with the desired level of accuracy. Besides, privacy concerns hinder the sharing of disease data across multiple hospitals. Another example is a smart farming system that employs a vision-based plant monitoring system for plant growth management [2]. However, privacy concerns pose a challenge as farmers are unwilling to share plant growth data with competitors, resulting in a relatively small amount of local data available for training models on a single farm.

Federated learning (FL) improves model accuracy while preserving privacy by aggregating local gradient updates from training nodes without sharing the local data [3], [4]. However, the global model trained by FL performs well primarily on local data that follows an independent and identically distributed (iid) pattern, which is rarely encountered in real-world scenarios. The presence of non-iid data, such as diverse patients and cases across different hospitals, poses a challenge to traditional FL: Optimizing the local model on one node will cause an accuracy decline on other nodes [5] due to inconsistent data distributions among the local training data. Furthermore, the centralized aggregator in traditional FL is vulnerable to attacks. For instance, even if the local models are encrypted, an attacker who gains access to the centralized aggregation server can exploit information embedded in gradients to reconstruct the original training data through Deep Leakage from Gradients (DLG) attacks [6]. In addition, the edge devices in FL are unable to trust each other due to commercial competition.

Some researchers proposed schemes aiming to train personalized models in FL [7], [8], where the objective is to train models that focus on a subset of the global data distribution [9]. However, such personalized models tend to overfit local data and have difficulty handling data from other nodes that exhibit variances in data distribution (i.e. the data skew problem). For example, a hospital trains a personalized model that accurately predicts the condition of local patients. However, when faced with patients transferred from another hospital, the performance of the personalized model significantly deteriorates. As a result, an orchestrator is required to maintain a balance between the personalized models and the global model for each node to deal with data skew issues.

To improve the security and the credibility of FL, several schemes combining blockchain with FL have been

Manuscript received 12 April 2022; revised 15 June 2023; accepted 27 June 2023. Date of publication 30 June 2023; date of current version 4 April 2024. This work was supported in part by Taishan Scholars Program under Grant TSQN202211214 and in part by Shandong Excellent Young Scientists Fund Program (Overseas) under Grant 2023HWYQ-113. Recommended for acceptance by D. Pompili. *(Corresponding authors: Longxiang Gao; Xi Zheng.)*

Chenhao Xu is with the School of Information Technology, Deakin University, Geelong, VIC 3220, Australia (e-mail: chenhao.xu@deakin.edu.au).

Jiaqi Ge is with Jilin University, Changchun 130012, China (e-mail: gejq18@mails.jlu.edu.cn).

Yong Li is with the School of Information Technology, Changchun University of Technology, Changchun 130012, China (e-mail: liyong@ccut.edu.cn).

Yao Deng and Xi Zheng are with Macquarie University, Macquarie Park, NSW 2109, Australia (e-mail: yao.deng@hdr.mq.edu.au; james.zheng@mq.edu.au).

Longxiang Gao is with Shandong Computer Science Center, Qilu University of Technology, Jinan 250316, China (e-mail: gaolx@sdas.org).

Mengshi Zhang is with Facebook, Menlo Park, CA 94025 USA (e-mail: mengshi.zhang@utexas.edu).

Yong Xiang is with the Deakin Blockchain Innovation Lab., School of Information Technology, Deakin University, Geelong, VIC 3220, Australia (e-mail: yong.xiang@deakin.edu.au).

Digital Object Identifier 10.1109/TMC.2023.3290925

proposed [10], [11], [12]. Blockchain is regarded as an effective improvement to FL due to its features such as privacy protection, tamper resistance, and decentralized nature [13]. One particular part of blockchain, the *Smart Contract*, is a self-executing decentralized computer program that has often been underutilized. In many cases, it is treated merely as an access interface for distributed storage database [14] or as a system for authorization and authentication [15]. However, this shallow integration with blockchain fails to provide FL with significant improvements in terms of credibility and resilience against single points of failure. Besides, while the smart contract is designed to facilitate agreements among participants, its potential role in coordinating nodes to generate personalized models has not been thoroughly investigated.

To resolve the problems arising from personalized models and the shallow integration of blockchain, a smart contract-based edge intelligence framework for IoT systems (SCEI) is proposed in this paper. In SCEI, personalized models are trained based on the balanced weight of local and global models to ensure good performance on both local and skewed data. Specifically, a novel decentralized algorithm is developed on the smart contract to facilitate joint model training and dynamically adjust the weights according to local accuracies on edge nodes. In addition, a novel committee election consensus algorithm developed on the blockchain is fully utilized by the smart contract, retaining the decentralized nature while improving efficiency.

The main contributions of this paper are listed below.

- A smart-contract-based edge intelligence framework *SCEI* is developed on a novel committee election consensus algorithm, which handles non-iid challenges while tackling credibility and security concerns for personalized model learning in IoT Systems.
- A novel decentralized algorithm is developed on the smart contract for collaborative model training and dynamic weight adjusting according to local accuracies, which ensures optimal model accuracy for both local and skewed data.
- An open-sourced prototype[1] is presented with extensive experiments conducted to show SCEI has improved model accuracy over benchmarks with acceptable overhead measured by time cost for each training round.

II. Related Work

The related work is introduced from three aspects, including FL and personalized learning in IoT, blockchain-empowered FL for edge intelligence, and smart contract driven FL.

A. Federated Learning and Personalized Learning in IoT

FL was first introduced in 2017 [3] as a method to efficiently train machine learning models on edge devices [10], [16], [17]. Most FL schemes aim to obtain an optimal global model by leveraging distributed local datasets across nodes. However, the performance variance of the global model on different nodes is inevitable [7], [18], [19], which has motivated recent research to focus on personalized model training. There are four main approaches to training personalized models: (1) localized independent training, where models are trained solely on local data [20]; (2) clustering training, where models are trained within clusters that exhibit similar data distributions [21], [22]; (3) optimizing personalized models through user's context [23]; and (4) co-training of personalized models and global models [7]. Our work aligns with approach (4), which aggregates weighted global and local models to get personalized models. However, most existing learning schemes lack a perception of local datasets and an optimization for skewed local data, except for the approach proposed by Deng et al. [7]. Their work leverages aggregated models to train personalized local models, which is similar to SCEI. However, their approach relied on a centralized aggregation server during training, whereas SCEI is fully distributed. Besides, SCEI incorporates the consideration of local accuracies from edge nodes to determine the optimal balancing weight in each training round, enhancing trustworthiness and practicality when dealing with skewed data. The scheme proposed by Deng et al. [7] serves as a benchmark in empirical studies.

B. Blockchain Empowered Federated Learning for Edge Intelligence

The integration of public (Proof-of-Work based) blockchain with FL was initially proposed in [24]. So far, blockchain has been utilized to (1) select reliable participating nodes for FL [25], (2) securely and reliably store local and global models [12], [15], (3) reduce the risk of a single point of failure from the centralized aggregate server [11], [17], [26], [27], [28], [29], and (4) resist poisoning attacks by detecting and penalizing attackers [11], [17], [30]. However, these lines of research focus solely on reaching an optimal global model accuracy without considering local model accuracy or data skew, limiting their application for most real-world scenarios (e.g., Smart Hospitals). Furthermore, the high computational load of blockchain reduces the feasibility of FL. To address these limitations, several asynchronous federated learning schemes have been proposed to improve efficiency by reducing the waiting time for aggregation [10]. However, due to the aggregation of outdated models from slow nodes, the improved efficiency usually comes at the expense of model accuracy, as evidenced by experimental results. This paper develops a decentralized algorithm executed on the smart contract to balance the weight of local and global models, leading to improved efficiency and performance when dealing with skewed data.

C. Smart Contract Driven Federated Learning

The smart contract, a distributed program in blockchain, possesses several features such as self-verification, self-execution, and tamper-resistance. These attributes enable the smart contract to execute various distributed operations on the blockchain at a minimal cost [31]. The smart contract has been utilized to 1) incentive nodes to participate in FL [14], 2) record and verify cached data on edge devices [32], 3) manage edge training in IoT [33], and 4) facilitate transaction authentication and user verification [12], [34]. However, these lines of research have

[1]The Github link is https://github.com/xuchenhao001/EASC.

TABLE I
SYMBOLS AND MEANINGS

Symbol	Meaning
$k,\ K$	the node number and the total number of nodes
$t,\ T$	the round number of federated optimization and its upper limit
$r,\ R$	the negotiation round number and its upper limit
w_G	the global model
w_k	the local model of node k
w_{kr}	the personalized local model of node k in negotiation round r
w_{kp}	the personalized local model of node k
α	the local model weight
α_l, α_u	the lower and upper bounds of the local model weight
α_{r*}^t	the optimal local model weight in training round t
A_{kr}	the test accuracy of node k in negotiation round r
$\bar{A}_r$	the average test accuracy in negotiation round r
γ	the ratio of committee size to the number of nodes
C	the identity set of the committee
C_L^t	the committee leader identity in training round t
H	the MD5 hash function

not fully explored the potential of the smart contract with the underlying consensus algorithm during the FL training process. In SCEI, blockchain and smart contracts are tightly integrated into the distributed learning process to achieve decentralized edge intelligence. This tight integration enables SCEI to strike a remarkable balance between model accuracy, scalability, credibility, and security.

III. PROPOSED SOLUTION

SCEI is explained from four aspects, including the system architecture and workflow (Section III-A), the smart contract coordinator and model weight balancing (Section III-B), the committee election and security (Section III-C), and the asynchronous solution (Section III-D). The relevant parameter symbols used in this section are listed in Table I.

A. System Architecture and Workflow

SCEI is built upon a consortium blockchain, with core component peers deployed on edge nodes, forming the backbone of the system. These peers execute the smart contract, maintain a shared ledger, and establish a peer-to-peer communication network among themselves, as shown in Fig. 1. To meet the requirements of storage capacity and computational power, the peers are strategically positioned on edge nodes and are responsible for collecting data from IoT devices within the same organization. For example, in a smart hospital system, peers deployed on edge nodes within medical imaging departments across hospitals join FL and collectively train models using healthcare data acquired from medical devices to categorize X-ray test results. Similarly, in a multi-farm smart farming system, peers running on edge nodes gather image data from IoT devices and utilize it to train models that predict the growth of specific crops.

Each edge node in the consortium blockchain undergoes verification of its identity by other nodes. A new edge node

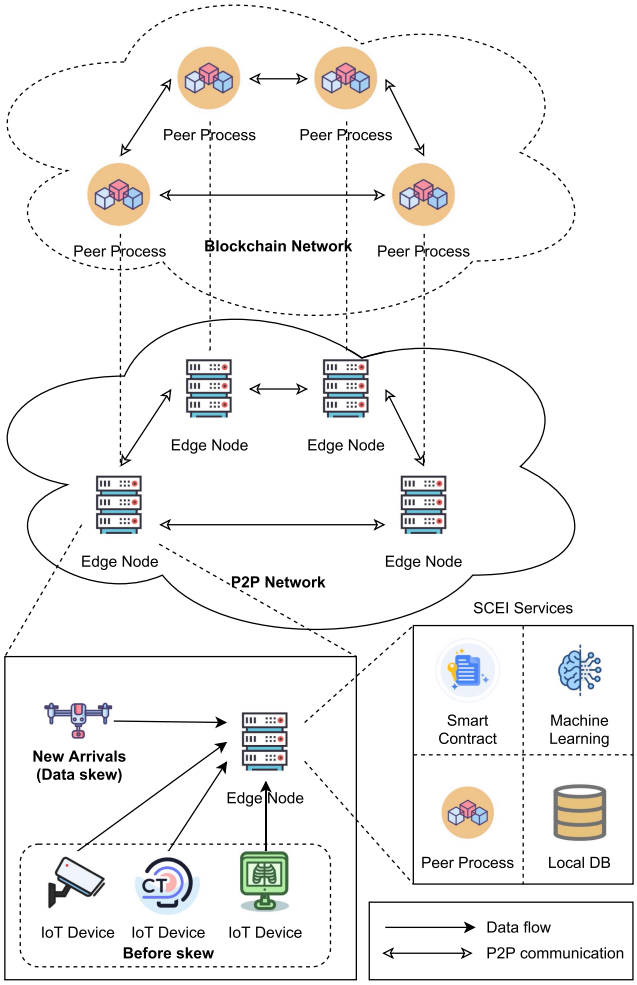


Fig. 1. The system architecture.

can only join the network if the majority of participants agree, which aligns with real-world scenarios. Upon joining SCEI, edge nodes are assigned sequential identities ranging from 1 to K. Besides, in SCEI, the smart contracts running on edge nodes are uniform and govern the process of distributed model training. To improve scalability, all original models in SCEI are stored in the local database of the edge nodes, with only MD5 hash values uploaded to the shared ledger. For other nodes to access the original model, the owner of the model must generate a model download address and upload it to the shared ledger.

During each training round, a committee comprising a predetermined fraction (γ) of nodes is selected based on the hash value of the global model from the previous round. This committee consists of a leader and several members. Section III-C explains the process of committee election. The committee utilizes the Raft consensus algorithm [35] to synchronize the original models. Compared to Paxos [36] and other consensus algorithms, Raft has demonstrated superiority due to its simpler and more comprehensible processes, adequate completeness for practical system requirements, proven safety properties, and comparable efficiency [35]. Specifically, Raft employs a more effective form of leadership than other consensus algorithms by restricting

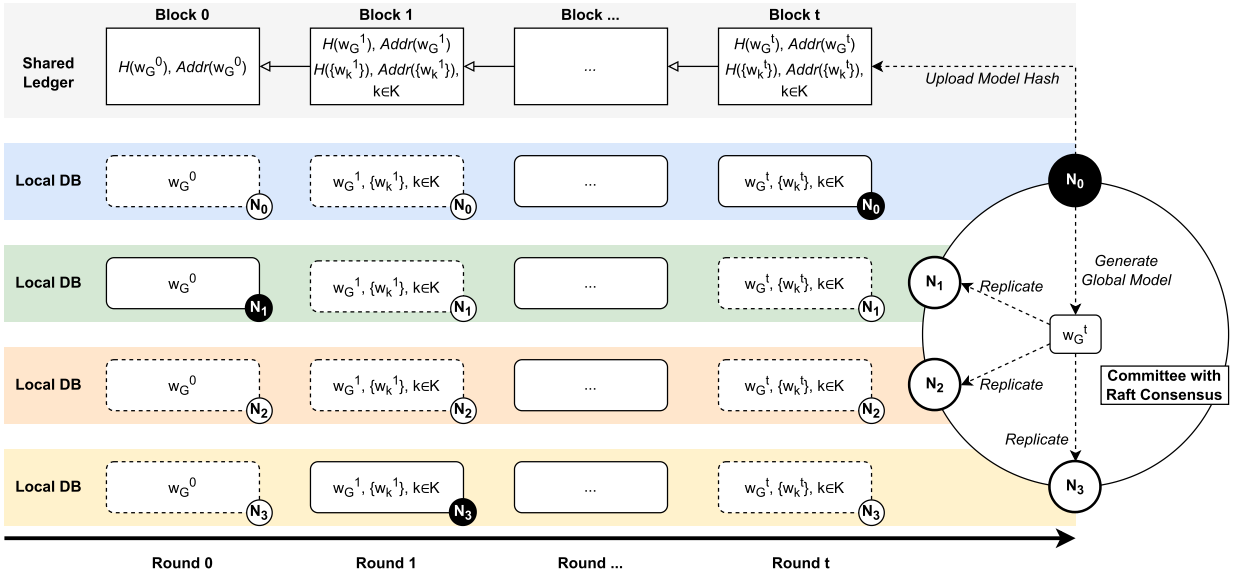


Fig. 2. The committee structure leverages the Raft consensus to replicate the global model. The committee leader, depicted in black, assumes the role of generating the global model, distributing it to committee members, and uploading the corresponding hash value to the shared ledger in each round.

the flow of log entries from the leader to other members. This simplifies the management of the replicated log and enhances the comprehensibility of Raft. Besides, Raft resolves conflicts simply and rapidly by electing leaders based on randomized timers, which introduces only a minimal additional mechanism to the heartbeats already required for other consensus algorithms.

Fig. 2 demonstrates the committee with Raft consensus and the replication process of the global model in training round t. Assume there are four edge nodes $(N_0, \ldots, N_3)$ elected as the committee for training round t, where N_0 serves as the committee leader. Although not explicitly shown in the figure, the other edge nodes are connected to the committee. Each edge node possesses its own local database for storing the original model while maintaining the shared ledger of the blockchain.

The workflow initiates with the training of an initial global model w_G^0 on node N_1, after which the hash value of the initial global model $H(w_G^0)$ and the corresponding model download address $Addr(w_G^0)$ are uploaded to the shared ledger. Specifically, assuming the IP address of the first node is IP_1, $Addr(w_G^0)$ can be interpreted as http://IP_1/$H(w_G^0)$. Following this, the first training round commences.

In each training round, such as round t, all nodes retrieve the base model w_G^{t-1} by using the download address provided in the shared ledger. Each node then utilizes its local data to train a new local model based on the base model. Afterward, the newly trained local model w_k^t from node k is uploaded to a randomly selected committee member, and replicated to all nodes within the committee by Raft. Concurrently, both the model download address $Addr(w_k^t)$ and the model hash value $H(w_k^t)$ are uploaded to the shared ledger, allowing other nodes to download and validate the original local model w_k^t. Upon receiving local models from all nodes, i.e. $\{w_k^t\}, k \in K$, the committee leader N_0 calculates a new global model w_G^t using (1). The hash value $H(w_G^t)$ and download address $Addr(w_G^t)$ are then uploaded to the shared ledger. Although the committee leader generates w_G^t, all committee members are able to verify it based on their replicated local models. Thereafter, all nodes obtain w_G^t using the download address provided in the shared ledger.

$$w_G^t = \frac{1}{K} \sum_{k=1}^{K} w_k^t \tag{1}$$

Next, nodes generate their personalized local models by proportionally integrating the global and the local model with the help of the smart contract, as detailed in Section III-B. This integration allows the personalized local models to learn features from both the local and global models. With this step, training round t is finalized. The entire training process finishes when the value of t reaches T.

Uploading hash values rather than original models to the shared ledger ensures the scalability of SCEI. In addition, the global models from earlier rounds are removable after a new global model has been successfully replicated to all nodes, which helps free up storage space and further improves scalability.

B. The Smart Contract Coordinator and Model Weight Balancing

As shown in Fig. 3, both the smart contract and the shared ledger are involved in the training process. The smart contract is responsible for 1) local model aggregation and 2) local model weight negotiation. Throughout the training process, the shared ledger records important information such as the hash values of models and the optimal local model weight.

To achieve improved personalized local models, a hyperparameter α that balances local and global models is introduced and is defined as (2). In (2), the variables represent the following: t corresponds to the current training round, w_k^t represents the

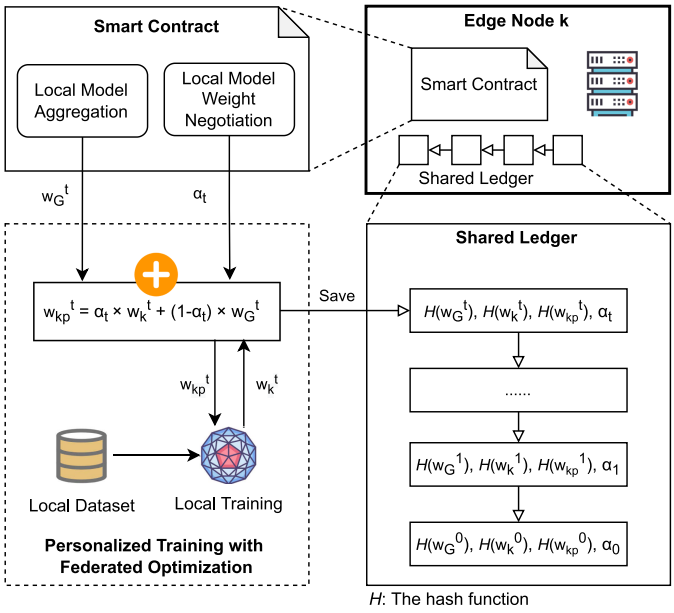


Fig. 3. The integration of machine learning and smart contract.

local model on node k, w_G^t denotes the global model obtained through FedAvg [3], and w_{kp}^t represents the personalized local model. The hyperparameter α plays a significant role in achieving a balance between the global and local models, particularly in handling non-iid datasets with the data distribution skew. When $\alpha = 0.0$, the local model is disregarded, and SCEI is equivalent to FedAvg. Conversely, when $\alpha = 1.0$, the personalized local model in SCEI becomes independent and loses its generalization capabilities.

$$w_{kp}^t = \alpha \times w_k^t + (1-\alpha) \times w_G^t. \tag{2}$$

The optimal value of α cannot be predicted in advance due to the uncertainty of data distribution among the nodes [7]. In SCEI, the smart contract enforces each node to test multiple α values during each training round to determine the best one. As shown in Algorithm 1, the corporation of the smart contract with the training procedure on node k is illustrated in lines 5 to 11 and 21 to 30. Specifically, in training round t, the smart contract uploads the hash value of the global model $H(w_G^t)$ to the shared ledger, triggering node k to download the global model w_G^t from the committee leader C_L^t, as indicated in lines 5 and 21. Suppose R denotes the number of rounds to negotiate α, α_l and α_u represent the lower and upper bounds of α, and $step$ represents the increment of α in each negotiation round, which is calculated by (3) and implemented in line 22.

$$step = \frac{\alpha_u - \alpha_l}{R} \tag{3}$$

Next, node k generates multiple temporary personalized local models w_{kr}^t with different α values for R negotiation rounds and tests the accuracy of w_{kr}^t using the local test dataset, as shown in lines 23 to 27. Following this, in line 28, node k uploads the local test accuracy set $\{A_{kr}^t\}$ and the corresponding α set $\{\alpha_r\}$ to the shared ledger, which signals the smart contract to stop waiting in line 6. For each negotiation round, the smart contract averages $\{A_{kr}^t\}, k \in K$ and generates the average test accuracy $\bar{A}_r^t$, as illustrated in lines 7 to 9. The smart contract then employs one of two strategies to select the optimal α: (A) maximizing the average of the test accuracies, or (B) minimizing the variance of the test accuracies. If Strategy A is chosen, the smart contract selects the negotiation round r that yields the maximum $\bar{A}_r^t$, disregarding any relatively low test accuracies, as shown in lines 10 to 11. On the contrary, if Strategy B is adopted, the smart contract selects r that exhibits the minimum variance of $\{A_{kr}^t\}, k \in K$. Subsequently, the optimal local model weight α_{r*} is determined and uploaded to the shared ledger. As a result, node k calculates the optimal personalized local model w_{kp}^t for training round t, as shown in lines 29-30. If the required number of optimization rounds is not reached, node k initiates a new round of training based on w_{kp}^t, as shown in line 16.

Strategy A is tailored for a static personalized objective, where each node possesses a fixed personalized learning objective. For example, in a smart farming system, the personalized learning goal of one farm is to predict the growth of wheat, whereas the personalized learning goal of another farm is to predict the growth of beans [37]. Strategy B caters to a dynamic personalized objective, where nodes continuously learn and adapt to new data arrivals. For example, the learning goal of a smart hospital system is to recognize COVID-19 cases, but it changes over time as new viral strains emerge [38]. Therefore, the personalized learning objective of the hospital shifts, leading to dynamic updates in the personalized model based on gradients learned from models in other hospitals that have detected the new variants.

C. Committee Election and Security

The committee leader is responsible for aggregating local models and generating the global model in each training round, while the committee members are tasked with verifying the aggregation results. Ensuring the security of SCEI requires that the committee for each training round is distinct and unpredictable. Therefore, a new committee is elected for each round, based on the MD5 hash value of the most recent global model generated in the preceding round.

Specifically, the MD5 hash function denoted by H is utilized to compute the hash value of base64-encoded models. Suppose t represents the current training round number, and k represents the number of nodes in SCEI. The hash value of the most recent global model generated in the preceding training round is $H(w_G^{t-1})$. As shown in lines 17 to 18 of Algorithm 1, the identity of the committee leader, denoted as C_L^t, is determined by (4).

$$C_L^t \equiv H(w_G^{t-1}) (\text{mod}\tilde{} K). \tag{4}$$

Assuming γ represents the ratio of the number of nodes in the committee to the total number of nodes in SCEI, the identity set of committee members C^t in training round t is defined in (5).

$$\{C_L^t + 1, C_L^t + 2, \ldots, C_L^t + [\gamma K]\}. \tag{5}$$

The identification of the committee leader is verifiable for all nodes due to the unified global model in each training round. Besides, the unpredictability of the hash value for the

Algorithm 1: SCEI.

procedure SMARTCONTRACT
  initialize the global model w_G^0
  wait for local updates $\{w_k^t\}$, $k \in K$
  $w_G^t \leftarrow$ FedAvg($\{w_k^t\}$)
  upload $H(w_G^t)$ to the shared ledger
  wait for $\{A_{kr}^t\}$ and $\{\alpha_r\}$ ($r \in R$, $k \in K$)
  for $r = 1, 2, \ldots, R$ **do** *// R rounds of negotiation*
    $\bar{A}_r^t \leftarrow \sum_{k=1}^{K} A_{kr}^t / K$
  end for
  $r* \leftarrow \arg\max_{r \in R}(\{\bar{A}_r^t\})$
  upload α_{r*} to the shared ledger
end procedure

procedure LOCALTRAININGk ▷ *Run Parallelly*
  $w_{kp}^0 \leftarrow w_G^0$ *// initiate the personalized local model*
  for $t = 1, 2, \ldots, T$ **do** *// T rounds of optimization*
    $C_L^t \equiv H(w_G^{t-1}) (\text{mod } K)$
    $C^t \leftarrow \{C_L^t, C_L^t + 1, \ldots, C_L^t + [\gamma K]\}$
    $w_k^t \leftarrow$ LocalTrain(w_{kp}^{t-1}, localTrainDataset)
    upload $H(w_k^t)$ to the shared ledger
    download w_G^t from the committee leader C_L^t
    $step \leftarrow (\alpha_u - \alpha_l)/R$
    for $r = 1, 2, \ldots, R$ **do** *// R rounds of negotiation*
      $\alpha_r \leftarrow \alpha_l + r * step$
      $w_{kr}^t \leftarrow \alpha_r \times w_k^t + (1 - \alpha_r) \times w_G^t$
      $A_{kr}^t \leftarrow$ LocalTest(w_{kr}^t, localTestDataset)
    end for
    upload $\{A_{kr}^t\}$, $\{\alpha_r\}$ ($r \in R$) to the shared ledger
    download α_{r*} from the shared ledger
    $w_{kp}^t \leftarrow \alpha_{r*} \times w_k^t + (1 - \alpha_{r*}) \times w_G^t$
  end for
end procedure

latest global model ensures the randomness and fairness of the committee election process.

Assume there are five peers in SCEI to enable collaborative model training and ensure a fair committee election. The majority of these edge nodes are assumed to be honest, ensuring training the global model in the right direction and deterring malicious edge nodes from joining the network. Additionally, DDoS attacks are launched from external servers. In this context, the committee, randomly elected based on hash values of global models, offers resilience against a single point of failure and provides defense against DDoS attacks toward the centralized aggregate server. Furthermore, the immutable hash values on the blockchain help trace and verify malicious local models, preventing any evildoing by edge nodes. As a result, the security of the training process in SCEI is assured.

By leveraging the smart contract and blockchain to facilitate the model training process, SCEI ensures the trustworthiness of edge devices. The smart contract functions as a program, exhibiting self-verifiable, self-enforcing, and tamper-proof features, and is executed simultaneously across the nodes. Through a consensus among all nodes prior to deployment, the behavior of each edge device becomes predictable and aligned with the shared commercial objective. Moreover, the availability of models, accompanied by their verifiable hash values on the blockchain, enables the detection and blacklisting of any edge node attempting to upload malicious models.

Algorithm 2: SCEI-Async.

1: **procedure** SMARTCONTRACT
2: initialize the global model w_G^0
3: wait for the latest local updates w_k^{t*}, $k \in K$
4: $w_G^t \leftarrow$ FedAvg(w_G^{t-1}, w_k^{t*})
5: upload $H(w_G^t)$ to the shared ledger
6: wait for $\{A_{kr}^{t*}\}$ and $\{\alpha_r\}$ ($r \in R$) from node k
7: **for** $r = 1, 2, \ldots, R$ **do** *// R rounds of negotiation*
8: $\bar{A}_r^t \leftarrow \sum_{k=1}^{K} A_{kr}^{t*} / K$
9: **end for**
10: $r* \leftarrow \arg\max_{r \in R}(\{\bar{A}_r^t\})$
11: upload α_{r*} to the shared ledger
12: **end procedure**

D. Asynchronous Solution

To improve scalability, an asynchronous variant of SCEI (SCEI-Async) is introduced, which performs aggregation promptly upon receiving a new local model. The SCEI-Async algorithm closely resembles SCEI, except for the modifications in the smart contract, as depicted in Algorithm 2. In this case, t represents the version of the global model instead of the round number of federated optimization, and $t*$ refers to the most recent training round of node k, which is independent of t.

Upon receiving the latest local updates w_k^{t*} from node k, a new global model w_G^t is created by averaging w_G^{t-1} and w_k^{t*}, as shown in lines 3 and 4. The hash value of w_G^t is then uploaded to the shared ledger in line 5. Subsequently, node k tests various personalized local models using different α values $\{\alpha_r\}$, obtaining the corresponding local test accuracies $\{A_{kr}^{t*}\}$. In line 6, node k uploads the latest local test accuracy set $\{A_{kr}^{t*}\}$ and its corresponding α set $\{\alpha_r\}$ to the blockchain. As a result of the asynchronous aggregation strategy, the smart contract no longer waits for other nodes to upload their updated local test accuracies. Instead, the smart contract computes the average test accuracy $\bar{A}_r^t$ based on the most recent local test accuracies of other nodes stored in the blockchain, and selects the optimal negotiation round number $r*$, as shown in lines 7–11.

In SCEI-Async, nodes are not required to wait for the completion of other nodes' training rounds. However, this results in a higher frequency of model aggregation, which introduces additional computing overhead than SCEI. Furthermore, the increased frequency of committee elections raises the risk of inconsistencies in the blockchain consensus. Additionally, since there is no synchronized training round in SCEI-Async, the local test accuracy set and the corresponding α set stored on the blockchain may not be up-to-date. As the value of α relies

TABLE II
PARAMETER SETTINGS FOR BLOCKCHAIN

Parameter	Value
Block generation frequency	0.2s
The maximum number of messages in a block	500
The maximum size of a block	100MB
Election timeout in Raft	1-2s

TABLE III
PARAMETER SETTINGS FOR PERSONALIZED FEDERATED LEARNING

Parameter	Value
The number of nodes K	10
Local minibatch size B	10
The local training rounds E	5
The federated optimization rounds T	50
The negotiation rounds R	10
The proportion of committee nodes γ	0.3
Learning rate η	0.01
The bounds of the local model weight α_l, α_u	[0.5, 0.8]

on the latest local test accuracy set stored in the blockchain, the negotiated α could be biased, causing a degradation of the accuracy of personalized models.

In the experiments, SCEI-Async is utilized as a benchmark to showcase the advantages and disadvantages of utilizing an asynchronous aggregation strategy in SCEI.

IV. SYSTEM EVALUATION

The experiments are conducted to answer the following research questions and assess the performance of *SCEI*.

- **RQ1:** Can *SCEI* enhance model accuracy compared to the state-of-the-art approaches?
- **RQ2:** What is the impact of the number of nodes in the underlying network on model accuracy and convergence in *SCEI*?
- **RQ3:** What is the computational and communication overhead associated with *SCEI*?

A. Experiment Setup

The simulation environment consists of ten default nodes, each of which is a virtual machine (VM) with identical hardware and software configurations. Each VM is equipped with eight Cores, 8 GB of RAM, and an NVIDIA GeForce GTX 1080 Ti GPU. In terms of software, Python 3.6 and PyTorch v1.6.0 are installed on each VM to support model training. The blockchain infrastructure relies on Hyperledger Fabric v2.2.0, while a middleware application is developed on Express.js v4.16.1 and deployed on each VM to facilitate peer communication within the blockchain network. Besides, the election timeout of Raft [35] is randomized within a range of 1 to 2 seconds. The default settings for the blockchain network are provided in Table II.

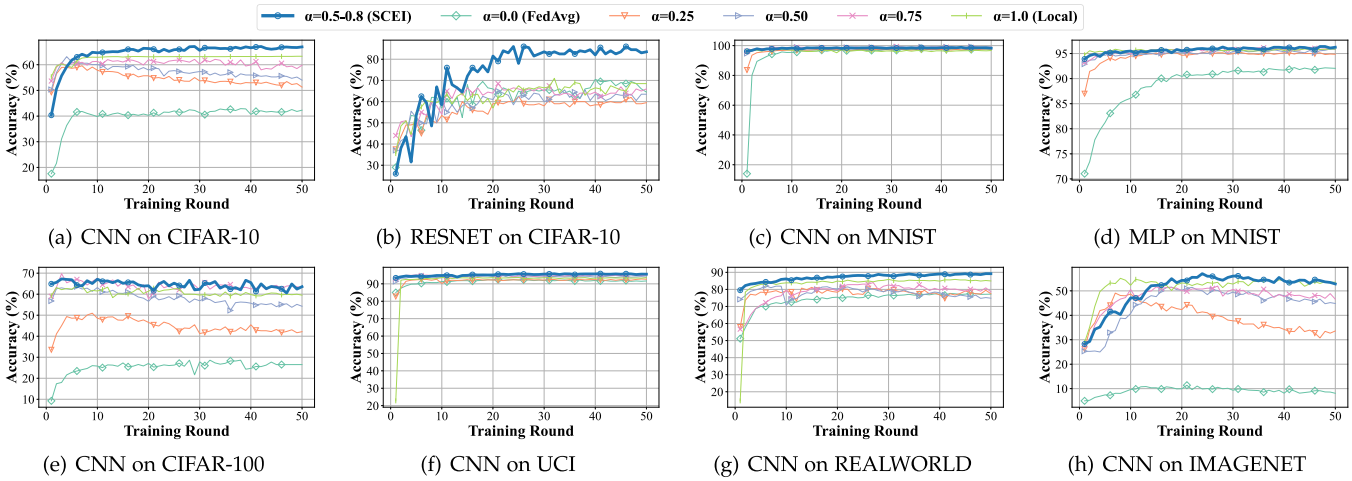
The performance of *SCEI* is compared with three other state-of-the-art schemes: FedAvg [3], Local Training [3] (each node trains its deep learning model locally and independently), and APFL [7]. In line with [3], [39], the parameter settings for personalized federated learning are described in Table III. To ensure generalization capability of neural networks and improve experimental efficiency, the range of 0.5 to 0.8 is selected for *SCEI* to dynamically adjust the value of α.

Six public datasets are used in the experiments, including MNIST, CIFAR-10, CIFAR-100, IMAGENET [40], UCI [41], and REALWORLD [42]. MNIST, CIFAR-10, CIFAR-100, and IMAGENET are usually adopted as benchmark datasets [3], [10], [39]. UCI and REALWORLD are human activity recognition datasets built from the recordings of 30 participants performing six activities and 15 participants performing eight activities, respectively.

To enable training of diverse personalized local models, the dataset settings are designed to be non-iid across nodes. Specifically, on each node, the dataset is sampled from four randomly selected classes within the original dataset. By default, each class contains 150 image samples, except for UCI and REALWORLD where each class comprises 500 image samples to ensure the model accuracy. To evaluate the performance of each personalized local model, a small portion of skewed data is introduced into the test dataset of each node. The skewed data is randomly sampled from classes other than the four selected classes. For example, in the case of the MNIST dataset, the training dataset on a node consists of images labeled "0", "1", "2", and "3", while the skewed test data is sampled from images labeled "4" to "9". The model accuracy is evaluated using the test dataset with varying data skew ratios, namely 0% (no skew), 5%, 10%, 15%, and 20%. This indicates that the test data derived from images labeled "4" to "9" accounts for 0%, 5%, 10%, 15%, and 20% of the overall test data, respectively.

In line with [3], the experiments employ three training models: MLP, a multilayer-perceptron with ReLU activation, consisting of 2 hidden layers and 200 units; CNN, a deep neural network with two 5×5 convolution layers, a fully connected layer comprising 512 units with ReLU activation, and a softmax output layer; RESNET, a CNN network with nine layers, featuring two residual blocks that allow the input to bypass certain layers [43].

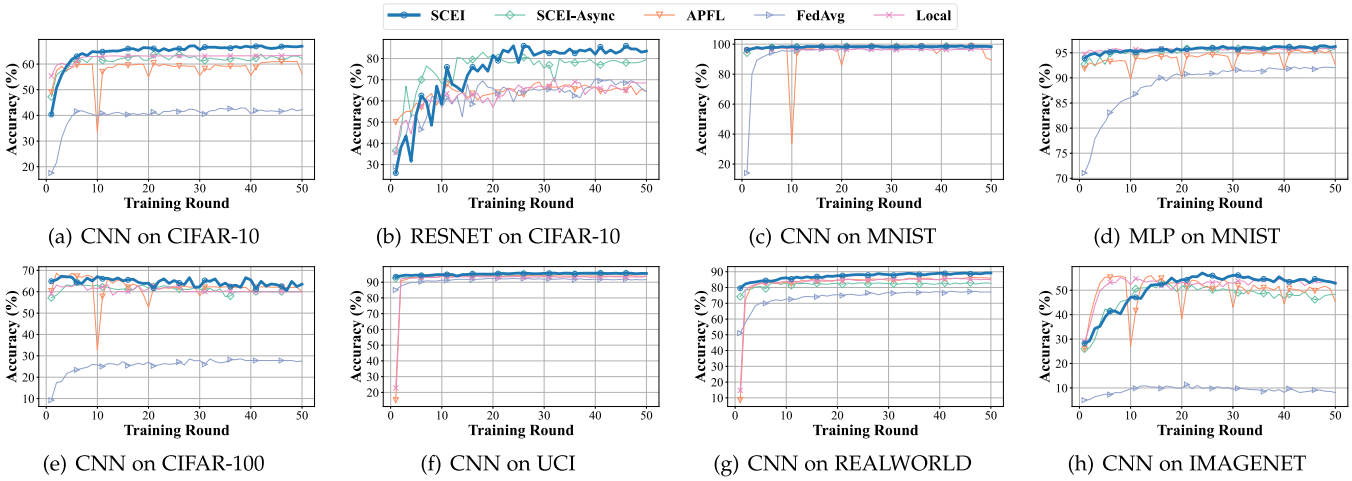
To answer *RQ1*, we compare the average local test accuracy of models using dynamic α and static α (ranging from 0.0 to 1.0). The average local test accuracy is computed by averaging the accuracies of personalized local models tested on each node, which quantifies the effectiveness of different schemes in training optimal local models. Then, the average local test accuracy of *SCEI* is compared with that of Local Training, FedAvg, and APFL over 50 rounds without data skew. Besides, the average local test accuracy is evaluated for different levels of data skew with the results presented in box plots. To answer *RQ2*, the average local test accuracy of models in *SCEI* is compared with 5, 10, and 20 nodes. To answer *RQ3*, the average overall time cost and the average communication time cost are recorded to reveal the computation and communication overheads of *SCEI* over 50 training rounds. The overall time cost of a training round is the time taken by a node to complete a training round, including communication time. The communication time cost is



(a) CNN on CIFAR-10 (b) RESNET on CIFAR-10 (c) CNN on MNIST (d) MLP on MNIST

(e) CNN on CIFAR-100 (f) CNN on UCI (g) CNN on REALWORLD (h) CNN on IMAGENET

Fig. 4. Comparison of average local test accuracy between SCEI with negotiated α values and SCEI with static α values.



(a) CNN on CIFAR-10 (b) RESNET on CIFAR-10 (c) CNN on MNIST (d) MLP on MNIST

(e) CNN on CIFAR-100 (f) CNN on UCI (g) CNN on REALWORLD (h) CNN on IMAGENET

Fig. 5. Comparison of average local test accuracy between SCEI and other schemes: FedAvg, APFL, and Local Training.

the time taken by a node to communicate with others, including committee election, model transmission, and uploading hash values to the blockchain.

B. Results Analysis

1) RQ1. Model Accuracy: When $\alpha = 0.0$, *SCEI* performs equivalently to FedAvg. Conversely, when $\alpha = 1.0$, the personalized models trained by *SCEI* lose their generalization capability and become similar to Local Training. Fig. 4 illustrates that the personalized models generated through the negotiation of α values achieve satisfactory accuracy after approximately 20 training rounds, with *SCEI* demonstrating a faster convergence speed than FedAvg.

Models trained with various static α settings exhibit varied performance, but the dynamically negotiated α consistently achieves the highest average local test accuracy. This is because *SCEI* learns additional features from the global model, surpassing the performance of local training (when $\alpha > 0.5$), and dynamically balancing the weights of local and global models based on the local test accuracy of each node. In contrast, FedAvg disregards personalized features and exhibits the worst performance, especially when training CNN on CIFAR-100 and IMAGENET datasets.

The comparison of average local test accuracy among five schemes, namely Local Training, FedAvg, APFL, SCEI-Async, and *SCEI*, is presented in Fig. 5. *SCEI* consistently outperforms the other schemes. Specifically, when training CNN on CIFAR-10, *SCEI* achieves higher average local test accuracy than Local Training, APFL, FedAvg, SCEI-Async by 3%, 6%, 25%, and 3%, respectively. Notably, when training CNN on UCI and REALWORLD datasets, the improvement of *SCEI* in average local test accuracy is still noticeable. This improvement is more prominent due to the smaller number of classes (6 and 8, respectively) in these datasets, which reduces the classification difficulty. By contrast, when training CNN on the more challenging IMAGENET dataset with 1000 classes, *SCEI* exhibits substantial improvement in model accuracy compared

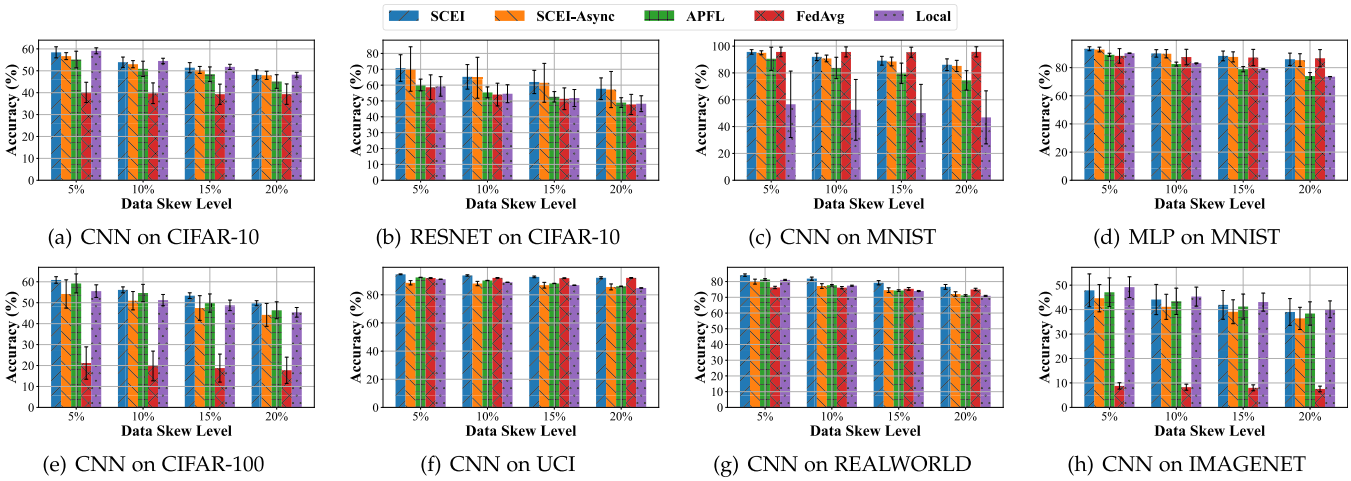


Fig. 6. Comparison of average local test accuracy between SCEI and other schemes under varying levels of data skewness.

to state-of-the-art schemes. This verifies the effectiveness of *SCEI* on complex datasets. By comparing with CNN, when training a more complex model RESNET on CIFAR-10, *SCEI* achieves 10% higher model accuracy than APFL, FedAvg, and Local Training, because a more complex model allows *SCEI* to learn more features from each node. This also reveals that *SCEI* has a greater potential in utilizing the complex neural network. Besides, *SCEI* outperforms Local Training in average local test accuracy, because the model trained on each node learns important features from the local models of other nodes. FedAvg always achieves the lowest average local test accuracy due to the inadequate adaptation of the global model to non-iid data.

Compared to Local Training and APFL, the improvement of *SCEI* in average local test accuracy is not as significant when training models on the MNIST dataset, because the MNIST dataset is the simplest among all the datasets. As depicted in Fig. 5, Local Training, serving as a benchmark, produces higher average local test accuracy (about 96%) when training CNN on MNIST than the other datasets. Besides, MLP exhibits a lower average local test accuracy than CNN due to its limited expressive capacity. Therefore, *SCEI* consistently demonstrates a stable performance improvement over state-of-the-art schemes despite the simplicity of the dataset.

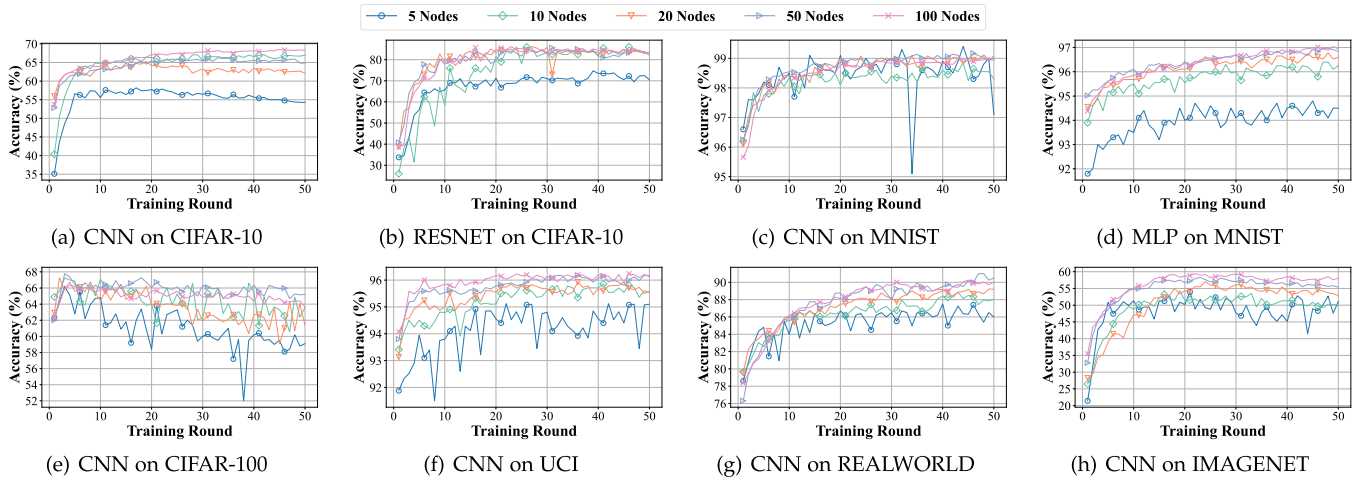
In SCEI-Async, the asynchronous aggregation strategy leads to the calculation of the optimal α based on outdated model accuracies of nodes. Since the decision on the optimal α can be influenced by outdated model accuracies from slower nodes, SCEI-Async consistently achieves lower model accuracy than SCEI, as shown in Fig. 5.

In general, as α approaches 1, the model converges faster and achieves higher accuracy. However, as α increases, the model's generalization capability declines, causing the personalized model to overfit the local training data. Such personalized models struggle to handle new data that does not fit the local data distribution. To investigate this, experiments are conducted by introducing skewed data ranging from 5% to 20% to each node. As depicted in Fig. 6, the accuracy of models trained on locally skewed data (at levels of 5%, 10%, 15%, and 20%) is compared. With an increase in skewed data, the model accuracy for all schemes decreases to varying extents. Specifically, FedAvg demonstrates the slowest rate of performance degradation since it is built for an ideal global model that can handle skewed data effectively. However, *SCEI* consistently outperforms other state-of-the-art schemes across all data-skew scenarios due to its federated optimization approach.

Two further findings are worth noticing. Firstly, when there are 5% skewed data, *SCEI* demonstrates better model accuracy than FedAvg regardless of the model and dataset. This highlights the suitability of *SCEI* for scenarios with low data skew, which are prevalent in real-world scenarios. For example, in a smart hospital, about 5% of patients may require referrals to other facilities due to complex conditions. Similarly, on a smart farm, around 5% of crops may be replaced with new varieties every year. Moreover, if the proportion of skewed data exceeds 5%, one possible solution is to adjust α to zero and switches *SCEI* to FedAvg mode to obtain a more generalized global model. Secondly, when training the CNN model on complex datasets, such as CIFAR-100, UCI, REALWORLD, and IMAGENET, *SCEI* consistently outperforms other schemes regardless of the data skew level. This indicates that *SCEI* is the most effective solution for achieving a balance between personalized and federated models in scenarios involving complex datasets with skewed data.

> Result 1: *SCEI* outperforms state-of-the-art schemes in terms of model accuracy, particularly when confronted with complex datasets. Besides, *SCEI* is capable of dealing with data skew, a common challenge faced by personalized model training approaches.

2) RQ2. Scalability: There is a balance between model accuracy and costs. As shown in Fig. 7, *SCEI* achieves higher average local test accuracy with an increased number of nodes (as well as increased computing and communication costs).



(a) CNN on CIFAR-10 (b) RESNET on CIFAR-10 (c) CNN on MNIST (d) MLP on MNIST

(e) CNN on CIFAR-100 (f) CNN on UCI (g) CNN on REALWORLD (h) CNN on IMAGENET

Fig. 7. Comparison of average local test accuracy for SCEI across different numbers of nodes.

This improvement is due to the inclusion of more nodes with diverse data, which improves the accuracy of personalized local models in non-iid scenarios. However, the rate of improvement in average local test accuracy diminishes when the number of nodes surpasses 20, the ideal node number. For example, when training CNN on CIFAR-10, increasing the number of nodes to 10 increases the accuracy by 10%, whereas increasing it to 100 only increases the accuracy by 5%. This is because the CNN model reaches a saturation point in terms of learning new features from federated optimization after learning sufficient features from a certain number of nodes. Notably, the ideal node number 20 is application specific. For instance, as the IMAGENET dataset encompasses the largest number of classes and features to learn, *SCEI* consistently demonstrates enhanced performance as the number of nodes increases. Therefore, the ideal node number may increase if the model, dataset, and application scenario become more complicated.

When training CNN on MNIST with an expanding number of nodes ranging from 5 to 100, the increase in average local test accuracy is not obvious. The reason for this is that the CNN model is advanced enough to fit the MNIST dataset well and achieve high average local test accuracy, even with a limited amount of training data. By contrast, when training MLP on MNIST, the increase in the average local test accuracy is more noticeable (about 3%) as the number of nodes increases from 5 to 100. This disparity arises due to the simpler architecture of the MLP model, which requires a larger amount of data to achieve higher accuracy. In fact, when comparing training the MLP model with 100 nodes to training the CNN model with 5 nodes on the MNIST dataset, CNN easily attains superior model accuracy.

Result 2: *SCEI* quickly converges to an acceptable average local test accuracy and shows the potential for a better personalized local model with additional local data.

3) RQ3. Computation and Communication Overhead: As shown in Fig. 8, Local Training always exhibits the lowest overall time cost in each training round due to no communication involved. Therefore, Local Training serves as a benchmark for the other three schemes. However, when training CNN on the UCI and REALWORLD datasets, the overall time cost of Local Training is significantly higher (around 10 seconds) compared to other datasets. This discrepancy can be attributed to the increased complexity of these datasets and CNN models. Generally, the more perceptrons there are in the model, the greater the number of weights to be tuned. The back-propagation algorithm employed for weight adjustment is performed for every training example and is repeated over numerous local training rounds. Consequently, increasing the number of perceptrons, training samples, and local training rounds leads to more weight adjustments, resulting in greater training time costs. Additionally, different neural network architectures contribute to varying computational workloads. For instance, in the case of CNN, the size of filters employed significantly impacts the computational workload. A 10×10 filter entails adjusting 100 parameters, while a 3×3 filter requires only 9 parameter adjustments. Consider an image with dimensions of 150×150 and a stride size of 1. With a 10×10 filter, the filter would be applied $140 \times 140 = 19,600$ times, thus necessitating a total adjustment of $19,600 \times 100 = 1,960,000$ parameters. On the other hand, a 3×3 filter would be applied $147 \times 147 = 21,609$ times, resulting in a total adjustment of $21,609 \times 9 = 194,481$ parameters. However, for MLP, the concept of filters does not apply, and ResNet incorporates shortcuts to bypass the computation of weights in certain layers.

Despite the variations among the models, when focused on a particular model and dataset, FedAvg usually incurs a higher average overall time cost than Local Training due to the additional steps of model parameter transmission and global model aggregation. Similarly, APFL requires more time than Local Training due to its two-phase training approach and periodic communication every 10 rounds. *SCEI* maintains an acceptable average overall time cost of less than 30 seconds

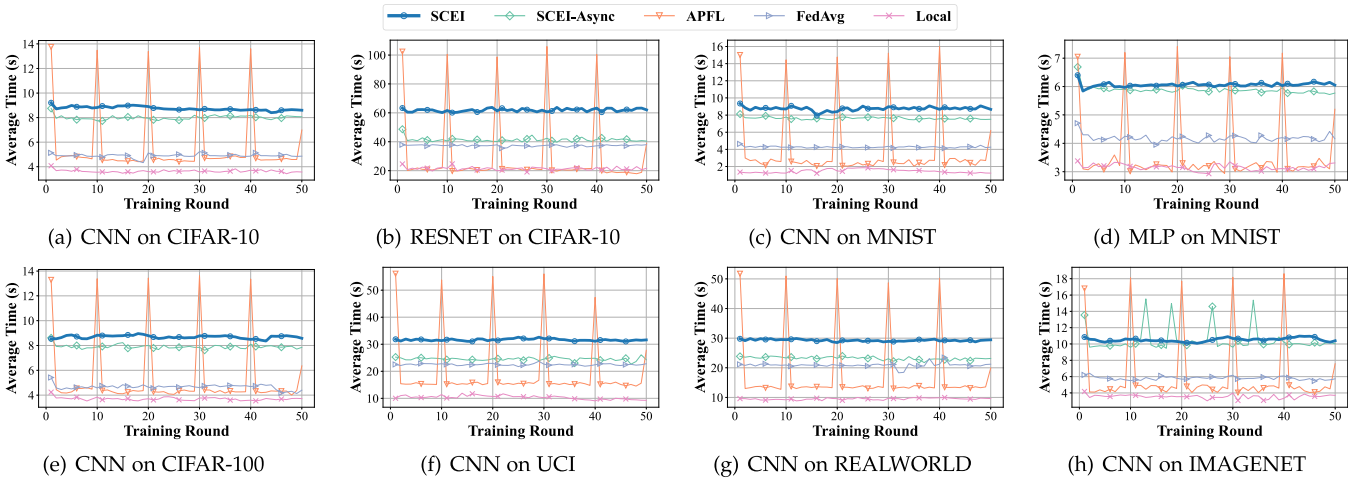


Fig. 8. Comparison of average overall time cost per training round for SCEI, FedAvg, APFL, and Local Training.

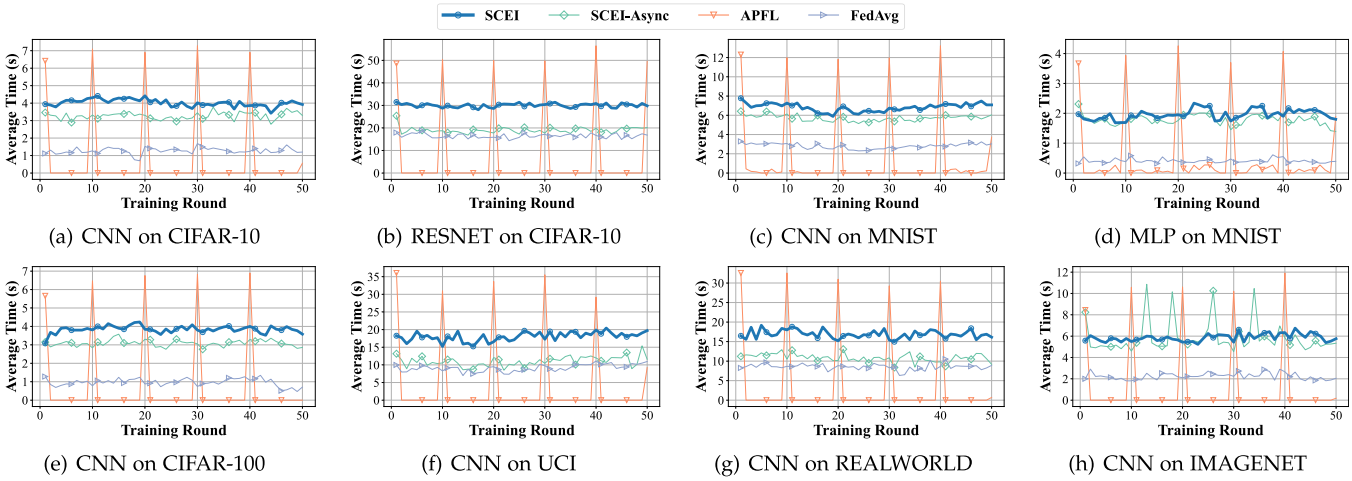


Fig. 9. Comparison of average communication time cost per training round for SCEI, FedAvg, and APFL.

in most cases, making it suitable for scenarios like smart hospitals and smart farms. When training CNN on the UCI and REALWORLD datasets, *SCEI* takes about 10 seconds longer than FedAvg per round, which is reasonable considering the improved model accuracy achieved on skewed data. By using an asynchronous aggregation strategy, SCEI-Async reduces the waiting time for other nodes, resulting in a lower average overall time cost than *SCEI*, at the expense of model accuracy.

As shown in Fig. 9, both *SCEI* and SCEI-async exhibit an average communication time cost of no more than 20 seconds per round, which is higher than that of FedAvg. This increase is due to the additional consensus procedure imposed by the blockchain. However, as the complexity of the model grows, such as when training the RESNET model on the CIFAR-10 dataset, the average communication time cost of *SCEI* per round grows to roughly 30 seconds. This is attributed to the heavier computational workload involved in the training process, which slows down the consensus, model transmission, and encryption/decryption steps. One way to improve efficiency is to reduce the waiting time for other nodes by adopting an asynchronous aggregation strategy, like SCEI-async. When dealing with more complex datasets and models, SCEI-async shows a greater improvement in terms of time efficiency (reducing by around 10 seconds per training round). However, this improvement comes at the expense of model accuracy, as SCEI-async achieves around 4% lower model accuracy than SCEI when training RESNET on CIFAR-10, as shown in Fig. 5. Reducing the committee size is another approach to decrease the communication time cost, although this compromises the security of the blockchain to some extent. In addition, since APFL requires model aggregations every 10 training rounds, the fastest node must wait for up to 9 rounds for the slowest node, resulting in longer communication time than *SCEI*.

Besides, for SCEI, there is no trade-off between training time and communication time. Generally, a more complex model will cause a higher training time and higher communication time, as

more parameters are required to be transmitted. This insight can be observed by comparing the results in Figs. 8 and 9.

> Result 3: In comparison to state-of-the-art schemes, *SCEI* exhibits a reasonable training overhead primarily due to its communication requirements within the secure blockchain infrastructure. However, this overhead remains relatively consistent and is lower than that observed in recent research on personalized models.

C. Discussion

The proposed scheme is a general framework that works effectively in cross-silo FL scenarios with reliable communication [19], [39]. To the best knowledge, the proposed framework is promising in terms of model accuracy, scalability, credibility, and security. In the experiments, as samples of each node are retrieved from four different classes and each class represents 25% of the total, data samples with a skew of over 20% are no longer considered skewed. Besides, the experiment results highlight that *SCEI* outperforms state-of-the-art schemes when dealing with complex models and datasets, even when a moderate level of data skew is present. However, due to resource and hardware limitations, experiments were not conducted in a real-world mobile IoT environment with realistic and ad-hoc communication time for nodes [44].

V. Conclusion

To address the challenges posed by non-iid and skewed data, as well as address trust and security concerns in distributed learning schemes in IoT systems, a smart contract-driven edge intelligence framework (SCEI) is proposed. SCEI introduces a novel personalized model training scheme by leveraging smart contracts to facilitate federated optimization across participating learning nodes. An open-sourced prototype of SCEI has been implemented, and experimental evaluations have been conducted using various learning models and datasets. The results demonstrate that SCEI outperforms state-of-the-art personalized learning schemes when handling skewed data. Future work will involve further assessments of the proposed framework using diverse models, datasets, and real-world multi-modal sensor data.

Acknowledgment

Many thanks to Wanping Bai for her valuable time and contributions in enhancing the quality of this paper.

References

[1] A. Srivastava et al., "Deep learning for detecting diseases in gastrointestinal biopsy images," in *Proc. IEEE Syst. Inf. Eng. Des. Symp.*, 2019, pp. 1–4.
[2] A. D. Boursianis et al., "Internet of Things (IoT) and agricultural unmanned aerial vehicles (UAVs) in smart farming: A comprehensive review," *Internet Things*, vol. 18, 2020, Art. no. 100187.
[3] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. Artif. Intell. Statist.*, 2017, pp. 1273–1282.
[4] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Trans. Intell. Syst. Technol.*, vol. 10, no. 2, pp. 1–19, 2019.
[5] H. Wang, Z. Kaplan, D. Niu, and B. Li, "Optimizing federated learning on non-IID data with reinforcement learning," in *Proc. IEEE Conf. Comput. Commun.*, 2020, pp. 1698–1707.
[6] L. Zhu, Z. Liu, and S. Han, "Deep leakage from gradients," in *Proc. Int. Conf. Adv. Neural Inf. Process. Syst.*, vol. 32, pp. 14774–14784, 2019.
[7] Y. Deng, M. M. Kamani, and M. Mahdavi, "Adaptive personalized federated learning," 2020, *arXiv:2003.13461*.
[8] Q. Wu, K. He, and X. Chen, "Personalized federated learning for intelligent IoT applications: A cloud-edge based framework," *IEEE Open J. Comput. Soc.*, vol. 1, pp. 35–44, 2020.
[9] S. Lee, X. Zheng, J. Hua, H. Vikalo, and C. Julien, "Opportunistic federated learning: An exploration of egocentric collaboration for pervasive computing applications," in *Proc. IEEE Int. Conf. Pervasive Comput. Commun.*, 2021, pp. 1–8.
[10] C. Xu, Y. Qu, Y. Xiang, and L. Gao, "Asynchronous federated learning on heterogeneous devices: A survey," 2021, *arXiv:2109.04269*.
[11] C. Xu, Y. Qu, T. H. Luan, P. W. Eklund, Y. Xiang, and L. Gao, "An efficient and reliable asynchronous federated learning scheme for smart public transportation," *IEEE Trans. Veh. Technol.*, vol. 72, no. 5, pp. 6584–6598, May 2023.
[12] S. Warnat-Herresthal et al., "Swarm learning for decentralized and confidential clinical machine learning," *Nature*, vol. 594, no. 7862, pp. 265–270, 2021.
[13] C. Xu, Y. Qu, T. H. Luan, P. W. Eklund, Y. Xiang, and L. Gao, "A lightweight and attack-proof bidirectional blockchain paradigm for Internet of Things," *IEEE Internet Things J.*, vol. 9, no. 6, pp. 4371–4384, Mar. 2022.
[14] J. Weng, J. Weng, J. Zhang, M. Li, Y. Zhang, and W. Luo, "DeepChain: Auditable and privacy-preserving deep learning with blockchain-based incentive," *IEEE Trans. Dependable Secure Comput.*, vol. 18, no. 5, pp. 2438–2455, Sep./Oct. 2021.
[15] S. Rathore, Y. Pan, and J. H. Park, "BlockDeepNet: A blockchain-based secure deep learning for IoT network," *Sustainability*, vol. 11, no. 14, 2019, Art. no. 3974.
[16] Y. Qu, C. Xu, L. Gao, Y. Xiang, and S. Yu, "FL-SEC: Privacy-preserving decentralized federated learning using SignSGD for the Internet of Artificially Intelligent Things," *IEEE Internet Things Mag.*, vol. 5, no. 1, pp. 85–90, Mar. 2022.
[17] C. Xu, Y. Qu, P. W. Eklund, Y. Xiang, and L. Gao, "BAFL: An efficient blockchain-based asynchronous federated learning framework," in *Proc. IEEE Symp. Comput. Commun.*, 2021, pp. 1–6.
[18] R. Hu, Y. Guo, H. Li, Q. Pei, and Y. Gong, "Personalized federated learning with differential privacy," *IEEE Internet Things J.*, vol. 7, no. 10, pp. 9530–9539, Oct. 2020.
[19] H. B. McMahan et al., "Advances and open problems in federated learning," *Found. Trends Mach. Learn.*, vol. 14, pp. 1–210, 2021.
[20] M. Khodak, M.-F. F. Balcan, and A. S. Talwalkar, "Adaptive gradient-based meta-learning methods," in *Proc. 33rd Int. Conf. Adv. Neural Inf. Process. Syst.*, 2019, vol. 32, pp. 5917–5928.
[21] V. Smith, C.-K. Chiang, M. Sanjabi, and A. S. Talwalkar, "Federated multi-task learning," in *Proc. 31st Int. Conf. Adv. Neural Inf. Process. Syst.*, 2017, vol. 30, pp. 4427–4437.
[22] N. Shlezinger, S. Rini, and Y. C. Eldar, "The communication-aware clustered federated learning problem," in *Proc. IEEE Int. Symp. Inf. Theory*, 2020, pp. 2610–2615.
[23] K. Wang, R. Mathews, C. Kiddon, H. Eichner, F. Beaufays, and D. Ramage, "Federated evaluation of on-device personalization," 2019, *arXiv:1910.10252*.
[24] H. Kim, J. Park, M. Bennis, and S.-L. Kim, "Blockchained on-device federated learning," *IEEE Commun. Lett.*, vol. 24, no. 6, pp. 1279–1283, Jun. 2020.
[25] J. Kang, Z. Xiong, D. Niyato, S. Xie, and J. Zhang, "Incentive mechanism for reliable federated learning: A joint optimization approach to combining reputation and contract theory," *IEEE Internet Things J.*, vol. 6, no. 6, pp. 10700–10714, Dec. 2019.
[26] J. Li et al., "Blockchain assisted decentralized federated learning (BLADE-FL): Performance analysis and resource allocation," *IEEE Trans. Parallel Distrib. Syst.*, vol. 33, no. 10, pp. 2401–2415, Oct. 2022.
[27] S. R. Pokhrel and J. Choi, "Federated learning with blockchain for autonomous vehicles: Analysis and design challenges," *IEEE Trans. Commun.*, vol. 68, no. 8, pp. 4734–4746, Aug. 2020.

[28] Y. Lu, X. Huang, K. Zhang, S. Maharjan, and Y. Zhang, "Communication-efficient federated learning and permissioned blockchain for digital twin edge networks," *IEEE Internet Things J.*, vol. 8, no. 4, pp. 2276–2288, Feb. 2021.
[29] P. Ramanan and K. Nakayama, "BAFFLE: Blockchain based aggregator free federated learning," in *Proc. IEEE Int. Conf. Blockchain*, 2020, pp. 72–81.
[30] H. B. Desai, M. S. Ozdayi, and M. Kantarcioglu, "BlockFLA: Accountable federated learning via hybrid blockchain architecture," in *Proc. 11th ACM Conf. Data Appl. Secur. Privacy*, 2021, pp. 101–112.
[31] B. K. Mohanta, S. S. Panda, and D. Jena, "An overview of smart contract and use cases in blockchain technology," in *Proc. IEEE 9th Int. Conf. Comput., Commun. Netw. Technol.*, 2018, pp. 1–4.
[32] L. Cui et al., "CREAT: Blockchain-assisted compression algorithm of federated learning for content caching in edge computing," *IEEE Internet Things J.*, vol. 9, no. 16, pp. 14151–14161, Aug. 2022.
[33] M. A. Rahman, M. S. Hossain, M. S. Islam, N. A. Alrajeh, and G. Muhammad, "Secure and provenance enhanced Internet of Health Things framework: A blockchain managed federated learning approach," *IEEE Access*, vol. 8, pp. 205071–205087, 2020.
[34] D. C. Nguyen et al., "Federated learning meets blockchain in edge computing: Opportunities and challenges," *IEEE Internet Things J.*, vol. 8, no. 16, pp. 12806–12825, Aug. 2021.
[35] D. Ongaro and J. Ousterhout, "In search of an understandable consensus algorithm," in *Proc. USENIX Annu. Tech. Conf.*, 2014, pp. 305–319.
[36] W. J. Bolosky, D. Bradshaw, R. B. Haagens, N. P. Kusters, and P. Li, "Paxos replicated state machines as the basis of a high-performance data store," in *Proc. USENIX Conf. Netw. Syst. Des. Implementation*, 2011, pp. 141–154.
[37] G. Idoje, T. Dagiuklas, and M. Iqbal, "Survey for smart farming technologies: Challenges and issues," *Comput. Elect. Eng.*, vol. 92, 2021, Art. no. 107104.
[38] B. Korber et al., "Tracking changes in SARS-CoV-2 spike: Evidence that D614G increases infectivity of the COVID-19 virus," *Cell*, vol. 182, no. 4, pp. 812–827, 2020.
[39] Y. Li, Y. Zhou, A. Jolfaei, D. Yu, G. Xu, and X. Zheng, "Privacy-preserving federated learning framework based on chained secure multi-party computing," *IEEE Internet Things J.*, vol. 8, no. 8, pp. 6178–6186, Apr. 2021.
[40] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2009, pp. 248–255.
[41] D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L. Reyes-Ortiz, "A public domain dataset for human activity recognition using smartphones," in *Proc. ESANN*, 2013, pp. 437–442.
[42] T. Sztyler and H. Stuckenschmidt, "On-body localization of wearable devices: An investigation of position-aware activity recognition," in *Proc. IEEE Int. Conf. Pervasive Comput. Commun.*, 2016, pp. 1–9.
[43] C. Rajagopalan, D. Rawlinson, E. Goldberg, and G. Kowadlo, "Deep learning in a bilateral brain with hemispheric specialization," 2022, *arXiv:2209.06862*.
[44] T. Zhang, Z. Shen, J. Jin, and X. Zheng, "A democratically collaborative learning scheme for fog-enabled pervasive environments," in *Proc. IEEE Int. Conf. Pervasive Comput. Commun. Workshops*, 2020, pp. 1–4.

Chenhao Xu received the BS degree in software engineering from the Beijing Institute of Technology, Beijing, China, in 2018, and the PhD degree in information technology from Deakin University, Australia, in 2023. He is currently an associate research fellow with the School of Information Technology, Deakin University, Australia. His research interests include blockchain, federated learning, and edge computing. He was the web chair and Technical Program Committee member of EAI TRIDENTCOM 2022. He is also a review editor of *Frontiers in Big Data*.

Jiaqi Ge received the BS and MS degrees from the College of Computer Science and Technology, Jilin University, Changchun, China, where she is currently working toward the PhD degree. She is currently working on federated learning and blockchain. Her research focuses on distributed computing.

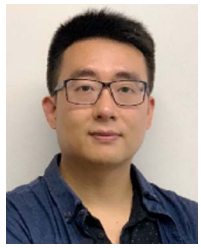
Yong Li received the MS degree in architecture of computer system in 2004 from the Jilin University, Changchun, China, where he is currently working toward the PhD degree. He is currently an associate professor of network engineering with the Changchun University of Technology, Changchun. His research interests include federated learning, egde-computing, information security, and privacy-preserving.

Yao Deng received the bachelor's degree in information technology from Deakin University, Australia, in March 2018, the second bachelor's degree in software engineering from SouthWest University, China, in July 2018, and the Master of Research degree from the Macquarie University. He is currently working the PhD degree with the Macquarie University, Australia. His current research interests include adversarial attacks and defenses, testing and anomaly detection of deep learning based autonomous driving systems.

Longxiang Gao (Senior Member, IEEE) received the PhD degree in computer science from Deakin University, Australia. He is currently a professor with the Qilu University of Technology (Shandong Academy of Sciences) and Shandong Computer Science Center (National Supercomputer Center in Jinan). He was a senior lecturer with the School of Information Technology, Deakin University and a postdoctoral research fellow with IBM Research and Development, Australia. He has more than 90 publications, including patent, monograph, book chapter, journal and conference papers. Some of his publications have been published in the top venue, such as *IEEE Transactions on Mobile Computing*, *IEEE Transactions on Parallel and Distributed Systems, The IEEE Internet of Things Journal*, *IEEE Transactions on Dependable and Secure Computing*, *IEEE Transactions on Vehicular Technology*, *IEEE Transactions on Computational Social Systems*, *IEEE Transactions on Industrial Informatics,* and *IEEE Transactions on Network Science and Engineering*. His research interests include fog/edge computing, blockchain, data analysis, and privacy protection. He has being the chief investigator (CI) for more than 20 research projects (the total awarded amount is more than $5 million), from pure research project to contracted industry research.

Mengshi Zhang received the BS degree in electronic engineering from Tsinghua University, in 2014, and the PhD degree from the Department of Electrical and Computer Engineering, University of Texas at Austin, Austin, TX, USA, in 2019. His research interests include fault localization, program repair, and machine-learning-oriented software engineering.

Yong Xiang (Senior Member, IEEE) received the BE and ME degrees from the University of Electronic Science and Technology of China, China, and the PhD degree from The University of Melbourne, Melbourne, VIC, Australia. He is currently a professor with the School of Information Technology, Deakin University, Australia. He is also the associate head of school (Research) and the director of the Artificial Intelligence and Data Analytics Research Cluster. He has authored or coauthored numerous research papers in high-quality international journals and conferences. He is the coinventor of two U.S. patents and some of his research results have been commercialised. He received a number of research grants, including several ARC Discovery and Linkage grants from the Australian Research Council. He is the editor/guest editor of several international journals. He has been invited to give keynote speeches and chair committees in a number of international conferences, review papers for many international journals and conferences, serve on conference program committees, and chair technical sessions in conferences.



Xi Zheng (Member, IEEE) received the PhD degree in software engineering from The University of Texas at Austin, Austin, TX, USA, in 2015. From 2005 to 2012, he was the chief solution architect for Menulog Australia. He is currently the director of Intelligent Systems Research Group (ITSEG.ORG), senior lecturer (aka associate professor U.S.) and deputy program leader of software engineering with Macquarie University, Sydney, NSW, Australia. His research interests include CPS verification, machine learning security, human vehicle interaction, edge intelligence and intelligent software engineering. He has a number of highly cited papers and is a PC members for PerCom (CORE A*) and TrustCom (CORE A).