IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received March 7, 2022, accepted March 20, 2022, date of publication March 30, 2022, date of current version April 11, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3163455

A Robust Multi-Stage Power Consumption Prediction Method in a Semi-Decentralized Network of Electric Vehicles

ZHISHANG WANG, (Graduate Student Member, IEEE), AND ABDERAZEK BEN ABDALLAH, (Senior Member, IEEE)

Adaptive Systems Laboratory, Graduate School of Computer Science and Engineering, The University of Aizu, Aizuwakamatsu, Fukushima 965-8580, Japan

Corresponding authors: Zhishang Wang (d8221102@u-aizu.ac.jp) and Abderazek Ben Abdallah (benab@u-aizu.ac.jp)

This work was supported by the Competitive Research Funding, The University of Aizu, under Grant CRF-6-2021.

ABSTRACT A Virtual Power Plant (VPP) balances the load on a power grid by allocating power generated by various interconnected units during periods of peak demand. In addition, demand-side energy devices such as Electric Vehicles (EVs) and mobile robots can also balance energy supply and demand when effectively deployed. However, the fluctuation of energy generated by renewable resources makes balancing energy supply a challenging goal. This paper proposes a semi-decentralized robust network of electric vehicles (NoEV) integration system for power management in a smart grid platform. The proposed approach integrates an aggregator with EV fleets into a blockchain framework. The EVs execute a multi-stage algorithm to predict the power consumption based on a novel federated learning algorithm named Federated Learning for Qualified Local Model Selection (FL-QLMS). From the evaluation results, the proposed system requires 35% fewer transactions in short intervals and propagation delays than the previous approaches and achieves better network efficiency while maintaining a high level of security. Moreover, NoEV achieves a 5.7% lower root mean square error (RMSE) than the conventional approach for power consumption prediction, which is a significant improvement. In addition, the FL-QLMS approach outperforms state-of-the-art methods in terms of robustness to client-side attacks. The evaluation results also show that the performance of FL-QLMS is not affected when 10% to 40% percent of the models are manipulated.

INDEX TERMS AI-enabled, blockchain-based, robust, power-management, EVs, smart grid.

I. INTRODUCTION

The utilization of renewable energy resources has increased significantly over the last decade. By the end of 2020, global renewable energy generation capacity reached 2799 gigawatts [1]. Meanwhile, European emission standards limit carbon dioxide emissions from regular cars to less than 95 g/km by 2020 [2]. Recently, a growing number of electric vehicles (EVs) are being integrated into smart grids to solve the problem of fluctuating renewable energy feed-in and shifting peak loads. To achieve efficient distribution and utilization of renewable energy, the concept of a virtual power plant (VPP) has been proposed as an intermediary between distributed energy resources, the power grid, controllable loads, and EVs [3]. When investigating the information exchange between an EV fleet and the VPP center, critical factors such as robustness and cost-efficiency of data storage, fast response to demand, and good scalability deserve much attention [4], [5].

The associate editor coordinating the review of this manuscript and approving it for publication was Kashif Saleem.

A. BACKGROUND AND MOTIVATION

Nowadays, modern EVs are equipped with devices for sensing, computation, communication, and data storage, providing a solution to offload cloud data centers [6]. Various efforts have been made to outsource edge computing tasks in vehicles [7], [8]. And a few studies have investigated the framework of vehicle edge computing for the VPP scenario [9]–[11]. For a complicated smart-vision task in a driving environment, vehicles must be equipped with high-speed systems that process a large amount of sensor data (about 1 Gb/s) [12]. However, one of the bottlenecks of today's local devices is still limited computing power. For example, the Renesas Xtreme, the most recent automotive microcontroller

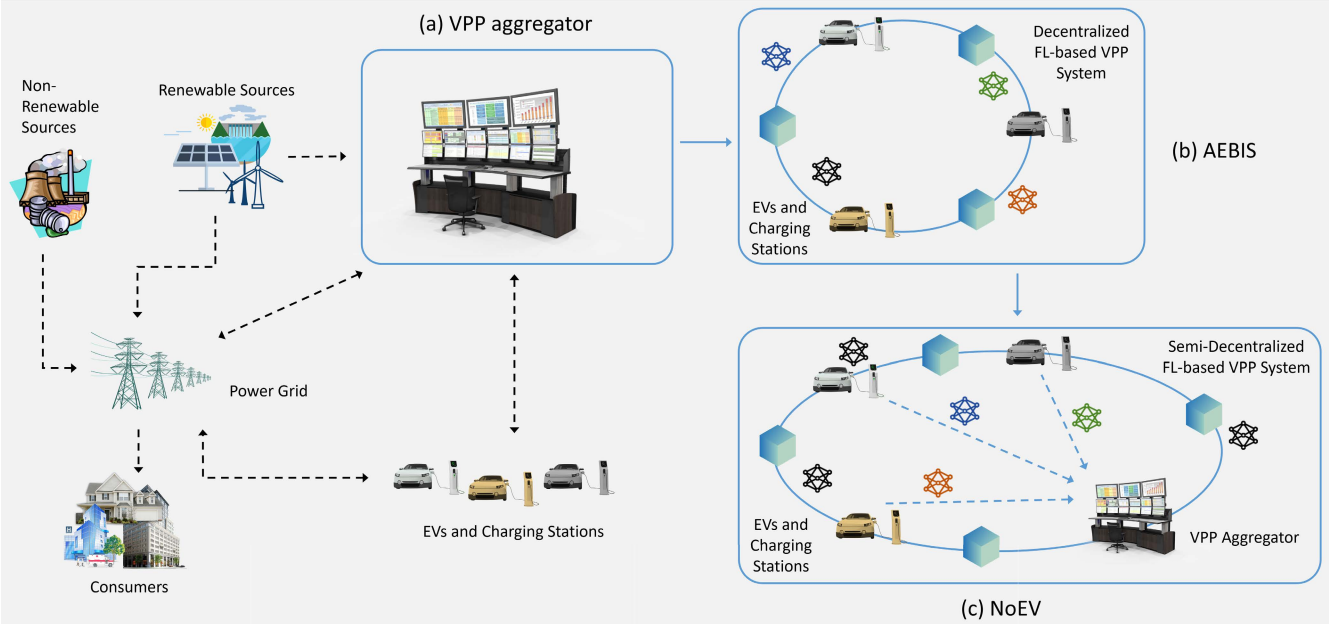


FIGURE 1. Virtual Power Plant (VPP): (a) conventional VPP aggregator, (b) AEBIS, (c) NoEV. In the conventional VPP architecture, the EV fleet is generally considered as a type of end consumers. A VPP aggregator monitors activity on the vehicular network. In AEBIS, where each EV participates in FL by sharing local models via the blockchain, the EV fleets form a blockchain network and the VPP aggregator is thus replaced. Compared to AEBIS, NoEV introduces a combination of VPP aggregator and EVs. The aggregator first merges the local models from EVs and then uploads the global models to the blockchain. In the proposed system, a substantial number of local models are not stored in the blockchain, which ensures a more efficient environment for collaborative learning. The colored models denote local models, and the models in black denote global models. Modified from Wang *et al.* [31].

family, includes devices with limited memory ranging from 32K flash/4K RAM to 8 flash/512K RAM [13].

Security and privacy are other concerns in vehicular edge computing (VEC), which has great significance in avoiding traffic collisions, improving road efficiency, and reducing environmental impact [14]. As a concrete example, protecting the functionality of range anxiety is critical for EV drivers. In addition, a cyberattack on EV or charging stations can result in a large-scale charging outage that can have a significant impact on the vehicle and the power grid. Secure data sharing and management [15]–[17] have been investigated, and various federated learning-based framework have been proposed for vehicular networks [18], [19]. Other privacy protection frameworks such as differential privacy attempt to deal with aggregation issues, however, with a challenge of achieving optimal trade-off between data utility and data leakage [20].

As a decentralized and secure framework, blockchain is a popular solution to replace the traditional approach in edge computing. It benefits federated learning in secure energy trading, management, and protection of EVs and drivers' data privacy. Secure bidirectional energy trading (charging and discharging) [21]–[24] for EVs has been investigated using a blockchain scheme. Research in [21], [25] studied both blockchain-based energy trading and data sharing in vehicle-to-grid (V2G) networks. The works in [26]–[28] proposed blockchain-based models for information authentication and trust management in a vehicular network. Other works proposed a variety of incentive-compatible schemes to encourage EV nodes to participate in demand response [29], [30]. While the above works addressed secure blockchain-based decentralized energy trading, EV participation, and data management issues in V2G, they did not concretely investigate secure data communication between the smart grid and the vehicular network. Moreover, the overall load on the network remains a significant challenge as the number of EVs continues to increase.

In previous work, we proposed an AI-Enabled Blockchain-based Electric Vehicle Integration System (AEBIS) for power management in smart grid platforms [31], [32]. AEBIS is a fully decentralized blockchain-based architecture for EV fleet model learning integrated with the VPP platform, as shown in Fig. 1(b). The system employs EV fleets as consumers and suppliers of electrical energy within a VPP platform [31]. A mechanism for charging the batteries of electric vehicles is proposed based on predicting the power consumption of the batteries using an artificial neural network. The neural network model is trained in a federated learning scheme and mapped into a reconfigurable AI chip [31]. Besides, by introducing blockchain technology into the system, secure and transparent service is achieved at the cost of storage and latency. In Fig. 1, the solid lines indicate the communication on the blockchain, and the dashed lines indicate the other communication activities. The overall decentralized architecture, EV charging mechanism, neural network configuration, and AI-chip integration were introduced in [31]. However, the earlier approach has the following shortcomings: (1) the constant proliferation choice

of new models for the blockchain solution leads to a heavy load on the network. The efficiency of the blockchain suffers greatly from this problem, making it challenging to apply in real-world scenarios, (2) the system is only designed for power consumption prediction for a local area, along with weather information at the start time. In a practical scenario where an EV travels to another city, the trained model cannot handle such a complicated case because the geographical and weather information changes during the journey, (3) the state-of-the-art federated learning approaches pay little attention to attack scenarios. The assumption that a malicious model can be uploaded in any training round leads to significant degradation of model accuracy. To the best of our knowledge, none of the previous approaches, including our work in [31], [32], have simultaneously considered system efficiency in blockchain-based vehicular network, EV participation with power consumption prediction, edge computation robustness for local devices.

B. CONTRIBUTION

This paper proposes a semi-decentralized Robust Network of Electric Vehicles (NoEV) integration system for power management in a smart grid platform. The proposed approach integrates an aggregator with EV fleets into a blockchain framework. Each EV in NoEV executes a multi-stage algorithm to predict its power consumption based on a novel federated learning algorithm named federated learning for qualified local model selection (FL-QLMS). The main contributions of this work are summarized as follows:

- A semi-decentralized robust network of electric vehicles (NoEV) integration system for power management in smart grid platform. The system maintains a high-security level while significantly increasing the efficiency of the blockchain network.
- A multi-stage power consumption prediction method which ensures the accurate prediction performance for intra and inter-district travel.
- A novel algorithm for robust federated learning, named federated learning for qualified local model selection (FL-QLMS).

The rest of this paper is organized as follows. In Section II, we discuss related work on the integration of blockchain and FL in edge computing, EV power consumption prediction, and client selection for federated learning. In Section III, we present a semi-decentralized blockchain-based platform, which is based on a multi-stage algorithm for power consumption prediction and a novel FL model selection mechanism. Section IV provides the performance evaluation of the proposed system. Section V provides discussion, and Section VI presents the conclusion and future work. The nomenclature used in this paper is given in Table 1 and 2.

II. RELATED WORK

In this section, we survey related works on 1) integration of blockchain and FL in the edge, 2) EV power consumption prediction, and 3) client selection in federated learning.

A. INTEGRATION OF BLOCKCHAIN AND FL IN THE EDGE

The work in [33] discussed the communication costs, resource allocation, incentive learning, and security and privacy issues. Weng *et al.* [34] proposed DeepChain, a framework with a value-based incentive mechanism based on blockchain for secure collaborative training. Wang *et al.* [35] studied two types of Byzantine attacks in a blockchain-empowered decentralized, secure multi-party learning system. Pokhrel *et al.* [19] proposed a local on-vehicle machine learning (oVML) method in an autonomous blockchain-based FL design. Bao *et al.* [36] proposed a decentralized FL system that provides incentives and disincentives for collaborative modeling. To analyze the latency performance and robustness of the blockchain system, decentralized architectures named BlockFL and FL-Block, were introduced in [37] and [38] respectively. Despite considering communication, computation costs, and incentive mechanisms, the increasing number of parties in the blockchain-based FL network poses a considerable challenge to the efficiency and applicability of the systems described in the works above.

B. EV POWER CONSUMPTION PREDICTION

Vatanparvar *et al.* [39] proposed a novel context-aware methodology to estimate driving behavior concerning future vehicle speeds for up to 30 seconds. In [40], a speed optimization framework is modeled for both battery life and power consumption of intelligent EVs during acceleration. Since these works focused only on the acceleration process, they are not suitable for *long-trip* scenarios. Ferro *et al.* [41] presented a detailed energy consumption model considering all aspects affecting the vehicle dynamics. Baek *et al.* [42] introduced a general methodology that allows predicting and optimizing the operation range of EVs. Zhao *et al.* [43] proposed a combined machine learning model for predicting the remaining range of EVs based on real driving data. One shortcoming of these methods is the complexity of their models. That is, the prediction for a single route requires a large amount of vehicle, routes, and battery data. Moreover, careful and elaborate *route-planning* for a terrestrial EV involves high time and data storage costs. Features, such as weather conditions and geography, were not investigated.

Gomez-Quiles *et al.* [44] proposed a novel ensemble method for predicting EV power consumption by examining the non-stationary time series of consumption. Although the algorithm is used for predictions for the next month or two, it is not unsuitable for specific driving activities.

C. CLIENT SELECTION IN FEDERATED LEARNING

The original FedAvg algorithm in [45] randomly selects a group of clients in each training round, which means that the communication quality and delay are difficult to evaluate. Authors in [46] researched performance degradation due to non-independently and identically distributed (non-IID) data in the FL protocol. The approach focuses on the resource

TABLE 1. Abbreviations.

Abbreviation	Meaning	Abbreviation	Meaning
AEBIS	AI-Enabled Blockchain-based Electric Vehicle Integration System	NoEV	Network of Electric Vehicles
AI	Artificial Intelligence	Non-IID	Non-Independently and Identically Distributed
EV	Electric Vehicle	oVML	on-Vehicle Machine Learning
FedAvg	Federated Average	OP_RETURN	Return Operator
FedCS	Federated Client Selection	PCP	Power Consumption Prediction
FL	Federated Learning	RMSE	Root Mean Square Error
FL-QLMS	Federated Learning for Qualified Local Model Selection	TX	Transaction
FPGA	Field Programmable Gate Array	V2G	Vehicle-to-Grid
IID	Independently and Identically Distributed	VPP	Virtual Power Plant

TABLE 2. Symbols.

Symbols	Meaning	Symbols	Meaning
$\{Lat_k\}_{k\in K}$	Latitude of all cities	$Long_d$	Longitude of the destination city
$\{Long_k\}_{k\in K}$	Longitude of all cities	$Long_p$	Longitude of the practical city
$\{Weather_{k,t}\}_{k\in K,t\in T}$	Weather information	$Long_s$	Longitude of the start city
∇g_G	Global Gradient	M	Model for neural network
∇g_L	Local Gradient	M_{aux}	Auxiliary model
b^r	Biases in the r-th training round	M_{local}	Local model
B_{delay}	Block propagation delay	$M_{selected}$	List of selected models
B_{main}	Number of blocks included in the main chain	n	Pre-determined value
B_{size}	Block size	N	Number of clients, number of models
B_{stale}	Blocks that were successfully mined but not inclued in the current best chain	$N_{0...62}$	Nodes in the blockchain
B_{total}	Total amount of blocks generated	$N_{selected}$	Number of selected models
$City_d$	Destination city	N_{total}	Total number of cities
$City_{ID}$	List of city IDs	P_a, P_b	Parameters of model a and b
$City_s$	Start city	P_{aux}	Parameters of the auxiliary model
D	Data set	P_i	Parameters of the local model i
DI	Diversity between model parameters	P^j	A single parameter of parameter vector P
D_{iid}	Local I.I.D data set	PC_{pred}	Predicted Power consumption
D_{local}	Size of the local data set	r_s	Stale block rate
$D_{non-iid}$	Local non-I.I.D data set	S	Sample for the neural network model
DoD	Duration of Driving	t	Time
D_{train}	Training data	t_s	Start time
$E(W)$	Loss function with respect to set of weights	T	Time period of the weather data
ED	Euclidean distance	TI	Time interval
ED_{min}	Minimum Euclidean distance	T_{size}	Transaction size
H	Hash function	$User\_Info$	User Information
k	Fraction of hacked clients	W	Set of weights
K	Set of city IDs	W^{local}	Local Weights
Lat_c	Latitude of the calculated	W^r	Weights in the r-th training round
Lat_d	Latitude of the destination city	α	Fraction for the number of selected models
Lat_p	Latitude of the practical city	η	Learning rate
Lat_s	Latitude of the start city	σ^2	Variance
$Long_c$	Longitude of the calculated point		

constraints of clients, including data heterogeneity, computation limitation, and communication capability. In [47], the authors proposed a multicriteria-based approach for client selection in FL, which aims to group many clients in each round to reduce the communication rounds. However, none of these works considered the importance of local data that affects learning performance.

He *et al.* [48] proposed another scheme for data selection and resource allocation based on the importance of data in the FL system to improve the learning efficiency. Authors in [49] identified a fundamental property of FL, namely the temporal pattern and varying significance of different learning rounds. They formulated a long-term client selection and bandwidth allocation problem under finite energy constraints and proposed a new Lyapunov-based online optimization algorithm to guarantee long-term performance. Cho *et al.* [50] presented a convergence analysis of FL with limited client selection and demonstrates how local loss affects convergence speed. Zhang *et al.* [51] proposed a weight-based client selection mechanism to recognize the

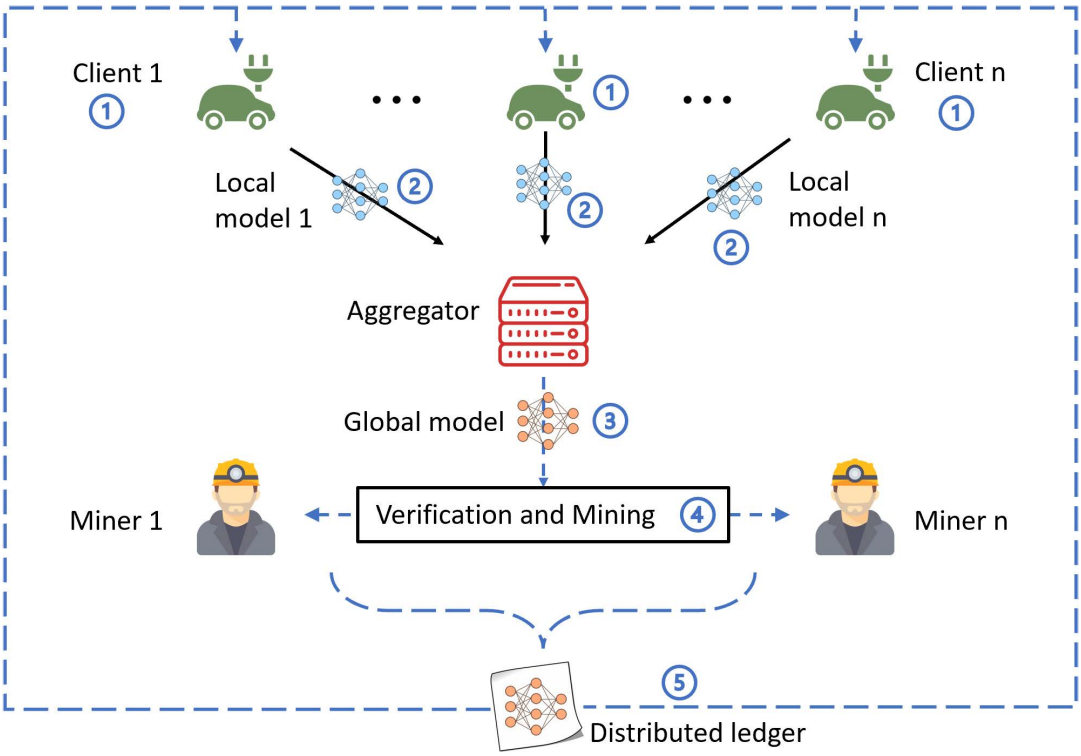


FIGURE 2. Overview of the proposed secure semi-decentralized FL-based framework. The black solid lines means that the local models are uploaded from clients to the aggregator. This communication is not conducted in the blockchain. Activities in the blockchain network are denoted by blue dashed lines. A VPP aggregator, EV fleets and a group of miners are integrated into the blockchain network. The workflow is briefly divided into five steps: 1) Each EV node trains a local model. From the second training round, each EV node updates the local model until convergence. 2) Each EV node uploads the local model to the aggregator. 3) We apply the FL-QLMS algorithm to select the qualified models for aggregation, resulting in a global model. 4) The aggregator creates and broadcasts a transaction (containing the global model) in the blockchain. After validation and mining, a distributed ledger is produced. 5) Each client downloads the global model from the distributed ledger for model update.

non-IID degrees of local data. However, the strategies mentioned above were adopted only when the clients' reputations remained unchanged. Considering that an edge node is prone to attacks in any training round, the quality of the model decreases due to tampering. Therefore, a *long-term* client selection mechanism is required to achieve a robust FL model.

III. ROBUST NETWORK OF ELECTRIC VEHICLES (NoEV) INTEGRATION SYSTEM

The proposed NoEV system integrates an aggregator with EV fleets into a blockchain framework. The EVs execute a multi-stage algorithm to predict the power consumption based on a novel federated learning algorithm named federated learning for qualified local model selection (FL-QLMS).

A. SECURE SEMI-DECENTRALIZED FL-BASED FRAMEWORK

As we explained in the previous section, the proposed system is based on a semi-decentralized architecture. As shown in Fig. 2, the solid lines in black means that the local models are uploaded from clients to the aggregator. This communication is not conducted in the blockchain. Other activities, which are denoted by dashed lines in blue, belong to the blockchain network. A VPP aggregator, EV fleets and a group of miners are integrated into the blockchain network. In the proposed architecture, the miners are vehicles itself, while we display EVs and miners separately in Fig. 2 for the sake of explanation. The overall workflow for each training round is described as follows:

1) In the first training round, each EV node initializes and trains a local model M_{local}. From the second training round, each EV node updates the local model until convergence.
2) Each EV node uploads the local model M_{local} to the aggregator.
3) After collecting local models, we apply the FL-QLMS algorithm to the model selection process. Then, the qualified models are selected for aggregation, resulting in a global model M_{global}.
4) a) At first, the global model is recorded as metadata in a new transaction TX_0. The aggregator feeds the transaction into a hash function H and generates a hash value $H(TX_0)$.
   b) The aggregator feeds $H(TX_0)$ to a signature algorithm with aggregator's private key, whereby an encrypted message is produced.

TABLE 3. Comparison between decentralized and semi-decentralized (proposed) FL-based system.

	Decentralized System	Semi-Decentralized System
Third Party Involvement	No	Yes
Data Management	Kept by clients	Kept by clients
Local Model	Clients to blockchain	Clients to aggregator
Global Model	Clients to blockchain	Aggregator to blockchain
Stability	High	High
System Complexity	High	Very High
Time and Energy Consumption	Very High	Medium

c) The aggregator then creates a transaction TX that contains the original transaction TX_0, the encrypted message and a public key.
d) The transaction will be sent from the aggregator to one of the nodes and then broadcasted to all miners.
e) Each miner can start performing validation. One will use the same hash function H and generate the hash value of TX_0. We denote the hash value by H_1. Since the same hash function always produces the same output, H_1 should be identical to $H(TX_0)$. Besides, the encrypted message will then be decrypted using the public key. If the resulted value matches H_1, the digital signature is proven to be valid. Therefore, TX is considered valid and added to each node's transaction pool. Once TX is confirmed by the blockchain network, it is added to the block.
f) A block header contains a 32-Byte previous block hash, 32-Byte Merkle root, 4-Byte timestamp, 4-Byte difficulty target, and 4-Byte nonce. A nonce is a 32-bit target that is guessed by miners by solving the following equation:

$$H(nonce) = \underbrace{0 \ldots 0}_{n\ bits} x_{n+1} \ldots x_{256} \tag{1}$$

where, n is a pre-determined value controlling the mining difficulty.
g) Once the nonce is found, the mined block is added to the distributed ledger.

5) Each client downloads the global model from the distributed ledger for model update.

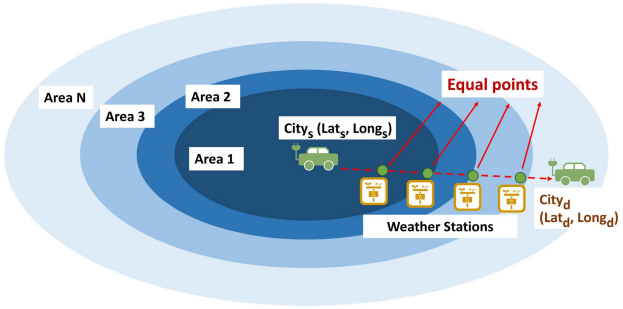
The local model is transmitted and merged without blockchain support. To ensure the robustness of the model aggregation, we present a novel algorithm named Federated Learning for Qualified Local Model Selection (FL-QLMS). Thus, the *fake* models are out and therefore do not affect the model aggregation. Besides, a multi-stage power consumption prediction method is proposed to improve the accuracy of the models, which will be introduced in section III-B. We will present the FL-QLMS algorithm in section III-C. The proposed semi-decentralized FL-based platform drastically reduces blockchain congestion while maintaining a high level of system security. A functionality comparison between the decentralized (i.e., AEBIS) and the proposed semi-decentralized (i.e., NoEV) systems is given in Table 3.

B. MULTI-STAGE POWER CONSUMPTION PREDICTION

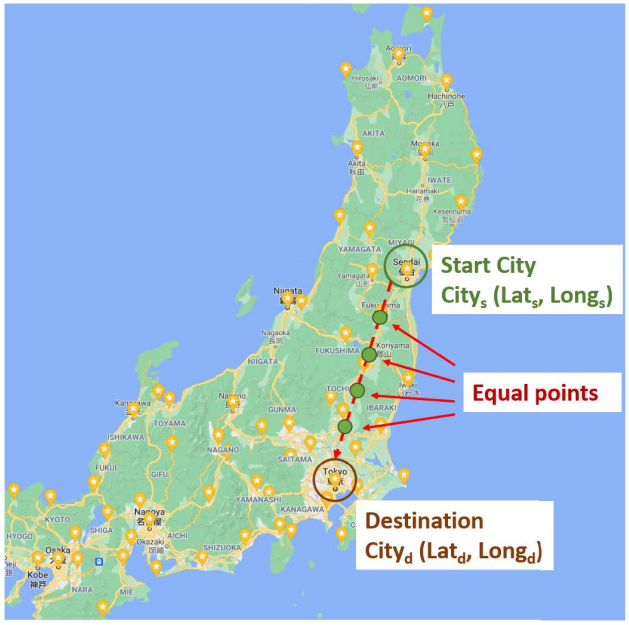
To present the multi-stage power consumption prediction, we consider a single trip from a start city to a destination, as shown in Fig. 3(a). The start city is located in **Area 1** and is denoted by $City_s$. The destination is located in **Area N** and is denoted by $City_d$. Each city is associated with latitude and longitude, e.g., $City_s$ is associated with latitude Lat_s and longitude $Long_s$. The duration of driving is abbreviated as DoD. We assume that DoD takes only integers and ranges from 1 to 12 hours to simplify the problem. The start time is denoted by t_s. We also assume that the EV moves at a constant speed in a straight line. Therefore, we can calculate the position of the EV at each time t, $t \in \{t_s, t_s + 1, \ldots, t_s + DoD - 1\}$. Each calculated position $City_c$ is called an "equal point" because the distance between two adjacent points is the same. The equal points are marked by green dots, as shown in Fig. 3(a). These equal points divide the entire path into multiple sections. We then predict the power consumption for each section and sum up the results. For each section, we need the following features: 1) start time t, 2) weather information at time t, 3) geographic information (latitude and longitude), 4) user information, and 5) duration of driving. For each equal point, we use the weather data from the nearest weather station, which is highlighted in yellow in Fig. 3(a). Algorithm 1 describes the proposed approach to predict power consumption in detail.

For ease of understanding, we split the entire algorithm into the following four stages: 1) *Initialization* (Line 1–4), 2) *Intermediate Position Calculation* (Line 5–8), 3) *Practical Position Calculation* (Line 9–23), and 4) *AI Prediction* (Line 24–33).

First, a start city $City_s$ (Lat_s, $Long_s$), a destination $City_d$ (Lat_d, $Long_d$), DoD, and start time t_s are given. Latitude and longitude of all cities are stored in $\{Lat_k\}_{k\in K}$ and $\{Long_k\}_{k\in K}$, respectively, where K denotes the set of city IDs. The weather information is presented by $\{Weather_{k,t}\}_{k\in K, t\in T}$, including temperature, rainfall, humidity, and wind speed, where T is the time period of the weather data and is given by each hour. $User\_Info$ contains information about the driver's gender and age. N_{total} denotes the total number of cities, and M is the neural network model for power consumption prediction.



(a) **Illustration of Power Consumption Prediction for A Single Trip. The green dots indicate positions that the car will pass through. Icons in yellow denote the nearest weather stations with respect to the green dots.**



(b) **An Example of Power Consumption Prediction for A Single Trip from Sendai to Tokyo in Japan. Each yellow star denotes a city associated with an explicit weather record. Created from Google Map [52].**

FIGURE 3. Illustration of the optimized power consumption prediction.

Algorithm 1 Multi-Stage Power Consumption Prediction

Require: Lat_s, Lat_d, $Long_s$, $Long_d$, DoD, t_s, $\{Lat_k\}_{k\in K}$, $\{Long_k\}_{k\in K}$, $\{Weather_{k,t}\}_{k\in K, t\in T}$, $User\_Info$, N_{total}, M
Ensure: Predicted Power Consumption PC_{pred}
1: Initialize empty arrays Lat_c, $Long_c$, Lat_p and $Long_p$
2: Initialize $City\_ID$
3: Initialize temporary variables ED and ED_{min}
4: Initialize sample S of size 11, which will be fed into model M
5: **for each** $i \in [0, DoD)$ **do**
6: $\quad Lat_c[i] = Lat_s + \frac{Lat_d - Lat_s}{DoD} i$
7: $\quad Long_c[i] = Long_s + \frac{Long_d - Long_s}{DoD} i$
8: **end for**
9: **for each** $i \in [0, DoD)$ **do**
10: $\quad ED_{min} = \sqrt{(Lat_c[i] - Lat_0)^2 + (Long_c[i] - Long_0)^2}$
11: $\quad Lat_p[i] = Lat_0$
12: $\quad Long_p[i] = Long_0$
13: $\quad City\_ID[i] = 0$
14: $\quad$ **for each** $j \in [1, N_{total})$ **do**
15: $\quad\quad ED = \sqrt{(Lat_c[i] - Lat_j)^2 + (Long_c[i] - Long_j)^2}$
16: $\quad\quad$ **if** $ED < ED_{min}$ **then**
17: $\quad\quad\quad ED_{min} = ED$
18: $\quad\quad\quad Lat_p[i] = Lat_j$
19: $\quad\quad\quad Long_p[i] = Long_j$
20: $\quad\quad\quad City\_ID[i] = j$
21: $\quad\quad$ **end if**
22: $\quad$ **end for**
23: **end for**
24: $PC_{pred} = 0$
25: **for each** $i \in [0, DoD)$ **do**
26: $\quad S[0], S[1] \leftarrow$ hour, weekday from $t_s + i - 1$
27: $\quad S[2], S[3], S[4], S[5] \leftarrow$ temperature, rainfall, humidity, and wind speed from $Weather_{City\_ID[i], t_s+i-1}$
28: $\quad S[6] = Lat_p[i], S[7] = Long_p[i]$
29: $\quad S[8], S[9] \leftarrow$ gender, age from User_Info
30: $\quad S[10] = DoD$
31: $\quad PC_{pred} = PC_{pred} + M(S)$
32: **end for**
33: **return** PC_{pred}

In Stage 1, the empty arrays Lat_c, $Long_c$ are initialized for recording equal points. Lat_p, $Long_p$, and $City\_ID$ are used to record nearest cities to each equal point. As shown in Line 6 and 7, we find coordinates of point that divide the line segment, $City_s City_d$, into multiple equal parts. The length of each array is set to DoD. The temporary variables ED and ED_{min} are initialized for calculating and storing distance information. An empty sample S is prepared as input for model prediction. In Stage 2, the latitude and longitude of each equal point are calculated, given Lat_s, $Long_s$, Lat_c, $Long_c$ and DoD. In Stage 3, for each equal point, we traverse all practical cities and find the nearest one by Euclidean distance. In Stage 4, we prepare samples with respect to each section and perform prediction. We extract the hour and day of the week from time t_s+i-1, $i \in [0, DoD)$. We extract gender and age from $User\_Info$. Given the weather data at time $t_s + i - 1$ and a city with $City\_ID[i]$, we obtain temperature, rainfall, humidity, and wind speed. We also obtain the latitude Lat_p and the longitude $Long_p$. Finally, we input the sample S into the model M. When the prediction is completed for each driving section, we obtain the final result PC_{pred}.

C. FL-QLMS ALGORITHM

As we explained in section II-C, the conventional approaches (i.e., work in [45]) randomly select a group of clients in each training round, which means that the communication quality and delay are challenging to evaluate. Moreover, the approach makes the model vulnerable to client attacks, which eventually leads to severe degradation of the prediction performance (e.g., accuracy in classification or root mean squared error in linear regression). Therefore, to ensure a robust learning environment, it is necessary to always select the *"qualified"* local models for aggregation, where qualified models are considered not polluted and contribute to the performance of the global model.

In the proposed FL-QLMS algorithm, we focus on selecting a group of *"qualified"* local models for model aggregation. In general, if the distribution of the data is similar, the convergence trend of a local model should also be similar

to the centralized model [53]. Thus, if the parameters of a local model are similar to those of the centralized model, that is, if the parameter diversity between the two models is low, the local model is considered to contribute to model aggregation. On the other hand, if a local model is contaminated by a malicious attack, the diversity between the contaminated model and the centralized model should be high. The diversity between two models can be expressed as follows:

$$DI_{a,b} = \|P_a - P_b\| \tag{2}$$

where $DI_{a,b}$ denotes the diversity between model parameters P_a and P_b.

Consider a FL process with N clients, each training round consists of the following six steps:

1) First, each client trains its local model using the collected local data set. In each local model, the gradient ∇g_L is calculated using adaptive moment estimation (Adam) optimizer [54], as shown by the following formula:

$$\nabla g_L = \frac{\delta E(W)}{\delta W} \tag{3}$$

where W denotes a set of weights, and $E(W)$ denotes the loss function with respect to W. $E(W)$ is used for measuring the model error and finding an optimal solution. Also, δ indicates partial derivatives.

2) Each client uploads the local model M^i_{local} to the aggregator. Besides, the aggregator is informed of the local data size $\left|D^i_{local}\right|$ from each client, where D^i_{local} denotes the local data set of the client i, $i \in N$.

3) The aggregator selects a group of uploaded models based on the FL-QLMS algorithm. The number of selected models is determined by the parameter α, i.e., $\alpha\%$ of all models used for aggregation. Given a total set of N models, the number of selected models is $N_{selected} = \lceil \alpha\% \cdot N \rceil$. The list of selected models is denoted by $M_{selected}$.

4) Before aggregating the models, we need to calculate the contribution of each selected model concerning the corresponding data size [45]:

$$w^i_{local} = \frac{\left|D^i_{local}\right|}{\sum_m^{N_{selected}} \left|D^m_{local}\right|}, i, m \in N_{selected} \tag{4}$$

where $\sum_m^{N_{selected}} \left|D^m_{local}\right|$ is the total data size with respect to the selected models.

5) The selected models are aggregated, resulting in a global model with gradient ∇g_L [45]:

$$\nabla g_G = \sum_{i=1}^{N_{selected}} w^i_{local} \nabla g^i_L \tag{5}$$

6) Once the edge nodes receive the global model from the server-side, they update the parameters as follows [54]:

$$W^{r+1} = W^r - \eta \nabla g_G \tag{6}$$

$$b^{r+1} = b^r - \eta \nabla g_G \tag{7}$$

where W^r and b^r denote the weights and biases in the r-th training round, respectively. η denotes the learning rate.

Algorithm 2 FL-QLMS With Auxiliary Model

Require: Auxiliary model M_{aux}, local models $\{M^i_{local}\}_{i \in N}$, the total number of clients N, and parameter α

Ensure: List of selected models for aggregation

1: Initialize an empty list $M_{selected}$, which is used to store the selected local models
2: Store all parameters of M_{aux} as a one-dimensional array, denoted by P_{aux}
3: Store all parameters of each M^i_{local} as a one-dimensional array, denoted by P^i_{local}
4: **for each** $i \in N$ **do**
5: Calculate the diversity between P_{aux} and P^i_{local} using the Manhattan distance, denoted by $DI_{aux,i}$
6: **end for**
7: Select $\lceil \alpha\% \cdot N \rceil$ models with lowest $DI_{aux,i}$ and store them to the list $M_{selected}$
8: **return** $M_{selected}$

We present the FL-QLMS algorithm with and without auxiliary model. Algorithm 2 describes how FL-QLMS works when an auxiliary data set is available. There are two ways to obtain a reliable auxiliary data set. One option is to pay the EV clients for the data set and get the data set on the spot. Another possibility is that the aggregator uses a group of EVs to collect data. Both methods collect the data without online data transmission, thus avoiding data leakage. The auxiliary data set is prepared on the aggregator's side. We denote the auxiliary model as M_{aux}. First, we store all parameters (weights and biases) of M_{aux} as a one-dimensional vector, denoted by P_{aux}. We treat each local model M^i_{local} in the same way and obtain the flattened vector P_i. P_{aux} and P_i have the same size, i.e., $|P_{aux}| = |P_i|$. Then, for each model, we calculate the diversity between P_{aux} and P_i using the Manhattan distance:

$$DI_{aux,i} = \sum_j^{|P_{aux}|} \left| p^j_{aux} - p^j_i \right| \tag{8}$$

where p^j_{aux} is a parameter of P_{aux}, and p^j_i is a parameter of P_i. Then, $\lceil \alpha \cdot N \rceil$ models with the lowest $DI_{aux,i}$ are selected for aggregation.

Algorithm 3 describes how FL-QLMS works when an auxiliary data set is not available. For each local model M^i_{local}, we store all parameters (weights and biases) as a one-dimensional vector, denoted by P^i_{local}. We then calculate the diversity $DI_{i,j}$ between P^i_{local} and each P^j_{local}, where $j \in N$ and $j \neq i$. Therefore, the average diversity of M^i_{local} can be computed as follows:

$$\bar{DI}_i = \frac{1}{N-1} \sum_{j=1, j \neq i}^{N} DI_{i,j} \tag{9}$$

Algorithm 3 FL-QLMS Without Auxiliary Model

Require: Local models $\{M^i_{local}\}_{i\in N}$, the total number of clients N, parameter α

Ensure: List of selected models for aggregation

1: Initialize an empty list $M_{selected}$ used to store the selected local models
2: Store all parameters of each M^i_{local} as a one-dimensional array, denoted by P^i_{local}
3: **for each** $i \in N$ **do**
4: **for each** $j \in N$ and $j \neq i$ **do**
5: Calculate the diversity between P_i and P_j using the Manhattan distance, denoted by $DI_{i,j}$
6: **end for**
7: $\bar{DI}_i = \frac{1}{N-1}\sum_{j=1,j\neq i}^{N} DI_{i,j}$ /* Calculate the average diversity between P_i and $\left\{P^j_{local}\right\}_{j\in N, j\neq i}$
8: **end for**
9: Select $\lceil \alpha\% \cdot N \rceil$ models with lowest DI_i and store them to the list $M_{selected}$
10: **return** $M_{selected}$

A model with a lower average diversity is considered more representative. In other words, the data set associated with the model is considered to have a similar distribution to the entire data set. For this purpose, $\lceil \alpha \cdot N \rceil$ models with the lowest $\bar{DI}_i$ are selected for aggregation.

IV. EVALUATION

A. EVALUATION METHODOLOGY

To show the advantage of our proposed system in terms of cost-efficiency, we studied the network load in a blockchain system and compared the proposed NoEV with AEBIS, oVML and DeepChain. We mainly focused on the number of blocks and transactions generated in a given period. We used an extensible simulation tool *BlockSim* for blockchain systems introduced in [55]. The configurations are summarized in Table 4. We simulated 63 nodes for AEBIS, oVML, and DeepChain and 63 + 1 nodes (1 additional node for the aggregator) for NoEV. We implemented ten runs for each simulation, with each run lasting 6000 seconds.

As discussed previously, the data set for the power consumption prediction includes weather, geography, and user information. We collected weather data from December 2019 to November 2020 in 63 cities in Japan [56]. The start time of vehicle reservation was set from 0:00 to 23:00 and the duration of driving from 1 to 12 hours. We considered the age of drivers ranging from 21 to 69 years old. The daily power consumption was measured considering the input characteristics and the measurement model [31]. We summarize the detailed information of the data set in Table 5. The data set contains a total of 66000 samples. We compared the proposed multi-stage power consumption prediction with the original power consumption prediction (PCP). We investigated the performance of the two methods under different driving activities — (a) short-distance journey, (b) mid-distance journey, and (c) long-distance journey. We summarize our definition of the above three activities in Table 6.

We considered a set of $N = 63$ clients in the federated learning environment. The data set contains 63000 training samples and 3000 test samples. First, we studied the effects of the model initialization methods — a) global initialization and b) local initialization. We considered an independent and identically distributed (IID) setting and employed the FedAvg (Federated Average) algorithm [45]. Then we considered a scenario where the local data is non-IID. Finally, going a step further, we compared the robustness of different FL algorithms against client attacks. For each algorithm, the simulation was repeated 20 times. Each simulation included 50 iterations. We used the root mean square error (RMSE) to measure the model's performance.

B. EVALUATION RESULTS

1) BLOCKCHAIN EFFICIENCY ANALYSIS

The block size was set to 1 megabyte (MB). We considered different combinations of TI and B_{delay}, which represent the average time to generate a new block and the propagation delay of a block, respectively. In [55], the transaction size T_{size} is 572.5 bytes by default. In our experiment, T_{size} is larger because each transaction must additionally store a portion of a model. The total number of parameters for our fully connected network (11-8-8-1) is 157. Each parameter in floating-point format occupies 4 bytes; thus, if we extract the parameters from the model, the total size is $157 \times 4 = 628$ bytes. In general, the return operator (OP_RETURN), which is part of the Bitcoin script language, is used to allow storing metadata on the blockchain with a maximum storage limit of 83 bytes according to release 0.12.0 [57]. Therefore, at least eight transactions are required for each model. The updated transaction size T_{size} is 572.5 bytes + 628/8 bytes = 650 bytes. We implemented 63 nodes (N_0 to N_{62}) for AEBIS, oVML, and DeepChain simulation with respect to a total of 63 EV clients. For simplicity, we consider a simple scenario that each miner has the same hash power. Therefore, given 63 nodes and the total hash power of 1, each of them will have a hash power of approximately 1.587%. For the NoEV simulation, the aggregator is introduced as an additional node N_{63}. Since N_{63} is not assigned any mining task, its hash power is set to 0%. We assume that the number of transactions (T_n) created per second is eight in NoEV. Accordingly, $T_n = 8 \times 63 = 504$ in AEBIS since 63 nodes are considered.

Table 7 summarizes the results of AEBIS, NoEV, oVML, and DeepChain on the BlockSim simulator. When the average block interval increases, the total number of blocks decreases accordingly. Moreover, as the block propagation delay increases, the number of blocks included in the main chain decreases, while the number of stale blocks increases. The stale blocks have been successfully mined but are not included in the current best chain. Therefore, the overall rate of stale blocks increases. When comparing with other

TABLE 4. Configuration for *BlockSim* simulation.

Parameters	Value	Description
TI	30, 60, 120 seconds	Time interval of block generation
B_{size}	1 MB	Block size
B_{delay}	1, 3, 6, 12 seconds	Block propagation delay
T_{size}	650 bytes	Transaction size
Nodes	**Hash Power**	**Description**
N_0	1.587%	In total, 63 nodes (miners) are considered in AEBIS,
N_1	1.587%	oVML and DeepChain.
...	...	Each miner has the same computing power.
N_{62}	1.587%	For NoEV, the aggregator acts as an
N_{63}	0%	additional node with a hash power of 0.

Note: We collected the data set regarding 63 cities. Therefore, to simply the preparation of data set allocation and federated learning schedule, we simulate 63 EV nodes for model training. Besides, in our blockchain proposal, each EV nodes also acts as a miner, thus we use 63 nodes in this work. To fairly compare the proposed work with other state-of-the-art works, we use 63 nodes for AEBIS, oVML and DeepChain too.

TABLE 5. Data set for vehicle energy consumption.

Input Feature	Value	Unit and Datatype
Start Time	0 to 23	-, Int
Weekday	1 to 7	Mon. to Sun., Int
Temperature	-13.6 to 39.5	$^{\circ}C$, Float
Rainfall	0 to 97.5	mm, Float
Humidity	0.05 to 1	-, Float
Wind Speed	0 to 26.2	m/s, Float
Latitude	34.09 to 41.30	$^{\circ}N$, Float
Longitude	134.84 to 141.94	$^{\circ}E$, Float
Gender	0 or 1	Male/Female, Int
Age	21 to 69	Years old, Int
Duration of Driving	1 to 12	Hour, Int
Output	**Value**	**Unit and Datatype**
Power Consumption	5.43 to 139.97	kWh, Float

TABLE 6. Driving activities.

Driving Activity	Duration of Driving	Driving Distance
Short Distance	1 – 2 Hours	< 250 km
Mid Distance	4 – 6 Hours	250 km – 500 km
Long Distance	8 – 12 Hours	> 800 km

methods, it is observed that NoEV generally requires the fewest transactions, especially for short TI. For example, for a short block interval ($TI = 30$) and short block propagation delay ($B_{delay} = 1$), NoEV requires an average of 25166 transactions, which is 38%, 37%, and 35% less compared to AEBIS, oVML, and DeepChain respectively. The significant decrease in NoEV can be explained by the fewer number of transactions, because the NoEV requires only global model transmission on the block, while the other methods require frequent local model transmission. DeepChain averaged the model updates every 10 to 20 iterations rather than at each iteration to increase communication efficiency, as in AEBIS and oVML. However, DeepChain and oVML still require the exchange of local models over the blockchain network.

2) MULTI-STAGE POWER CONSUMPTION PREDICTION

A comparison between PCP and the proposed multi-stage PCP is illustrated in Fig. 4. The overall prediction results are shown in Fig. 4(a), where the multi-stage PCP achieves 5.7% lower RMSE compared to PCP. We observed that the multi-stage PCP performs better in scenarios with the short-distance journey. This result is surprising because the original PCP mainly focuses on local driving activities and has achieved decent performance. Our most compelling case is long-distance driving. As illustrated in Fig. 4(d), the multi-stage PCP still achieves better results by completing 14.3% lower RMSE. Besides, we analyzed the performance variance of the two methods in each case. For medium and long distances, the variance of RMSE of the multi-stage PCP is more significant than that of PCP. The multi-stage approach can explain the reason. The multi-stage PCP first divides the journey into multiple sections for a long trip and then runs the prediction model for each section. When the prediction results are summed up, the errors caused by each prediction are also accumulated. Therefore, the multi-stage PCP leads to higher variability. On the other hand, for a short trip, e.g., one or two hours, the multi-stage approach has little effect, and therefore the variance of the multi-stage PCP is lower.

3) FEDERATED LEARNING FOR QUALIFIED LOCAL MODEL SELECTION (FL-QLMS)

We considered a set of $N = 63$ clients for the FL schedule. We split the whole data set D into the training set D_{train} of 63000 samples and test set D_{test} of 3000 samples. First, we evaluated two approaches of model initialization: a) global initialization and b) local initialization. Global initialization means that the aggregator creates an initial model and distributes it to all clients. On the other hand, local initialization involves each client creating its initial model and performing the training task. FedAvg is used for model aggregation. The number of local updates is set to one before each global aggregation. We randomly assigned 1000 samples to each client. Thus, each subset D^i_{iid} follows independent and

TABLE 7. The blockchain simulation results of AEBIS, NoEV, oVML, and DeepChain for different combinations of parameters.

Parameters		AEBIS [31]					NoEV (this work)				
TI	B_{delay}	B_{total}	B_{main}	B_{stale}	r_s	TX	B_{total}	B_{main}	B_{stale}	r_s	TX
30	1	196.4	190.8	5.6	2.9%	40619	200.1	193.9	6.3	3.12%	25166
	3	200.25	182.9	17.4	8.7%	38141	197	180.6	16.4	8.31%	24532
	6	197.5	170.1	27.4	13.9%	33425	209.5	176.3	33.3	15.87%	23299
	12	194.5	148.4	46.1	23.7%	30467	203.9	155.4	48.5	23.79%	20560
60	1	103.4	102	1.4	1.3%	21296	98.5	96.9	1.6	1.7%	17201
	3	104.1	97.5	6.6	6.4%	19851	100.8	95.4	5.4	5.33%	16811
	6	102.1	94.5	7.6	7.5%	19925	100.9	93.3	7.8	7.68%	15782
	12	100.1	84.8	15.4	15.4%	18160	106.8	90	16.8	15.7%	15287
120	1	46.8	46.1	0.6	1.3%	9031	48.4	48.4	0	0.00%	8727
	3	50.2	48.6	1.6	3.2%	10021	51	49.6	1.4	2.7%	9620
	6	52.6	49.9	2.8	5.2%	11673	50	47.1	2.9	5.8%	8768
	12	55.3	50.4	4.9	8.8%	10419	50.4	46.5	3.9	7.7%	9086
Parameters		**oVML [19]**					**DeepChain [34]**				
TI	B_{delay}	B_{total}	B_{main}	B_{stale}	r_s	TX	B_{total}	B_{main}	B_{stale}	r_s	TX
30	1	196.6	191.9	4.8	2.4%	39805	198.3	192.6	5.6	2.8%	38807
	3	192.4	175.6	16.8	8.7%	36892	195.8	180.5	15.3	7.8%	35937
	6	193.4	164.8	28.6	14.8%	37477	197.6	167.9	29.8	15.1%	30397
	12	203.1	152.1	51	25.1%	30204	203	154.5	48.5	23.9%	29601
60	1	101	99.6	1.4	1.4%	21750	102.1	100.1	2	2.0%	21224
	3	101.1	96.4	4.8	4.7%	20652	103.3	98	5.3	5.1%	18598
	6	103.6	93.5	10.1	9.8%	19640	95	86.5	8.5	9.0%	17374
	12	95.6	81.5	14.1	14.8%	16767	97.3	84.9	12.4	12.7%	16510
120	1	50	49.6	0.4	0.8%	8418	49.3	48.8	0.5	1.0%	10375
	3	50.5	49.1	1.4	2.7%	10623	50.6	50	0.6	1.2%	9504
	6	51.1	48.4	2.8	5.4%	9446	52.1	48.9	3.3	6.2%	9741
	12	48.8	44.1	4.6	9.5%	8522	48.3	44.4	3.9	8.0%	9284

B_{total}: The total amount of blocks generated.
B_{main}: The number of blocks included in the main chain.
B_{stale}: Blocks that were successfully mined but not included in the current best chain.
r_s: Stale block rate.
TX: Transactions.

identical distribution (IID), where $D_{train} = D^1_{iid} \cup D^2_{iid} \cup \cdots \cup D^N_{iid}$. Fig. 5 illustrates the impact of two model initialization options on training performance. The red and blue shaded regions denote local and global initialization performance fluctuation, respectively. While local initialization leads to slower convergence in the first 20 iterations, it achieves a lower average RMSE of 7.77 than global initialization at the end of training. This shows that it makes more sense to build the initial models on the client-side rather than on the server-side. Therefore, we implement local initialization in the following FL simulations.

We then considered a scenario where all local data is non-IID. We refer to this scenario as **Scenario-I**. We distributed the entire dataset across $N = 63$ clients, each of which is associated with 1 to 5 start cities. Besides, each local data set $D^i_{non-iid}$ contains different reservation times, i.e., morning, afternoon, or evening. For each $D^i_{non-iid}$, $i \in N$, the data size ranges from 200 to 2000. Similar to IID scenario, we have $D_{train} = D^1_{non-iid} \cup D^2_{non-iid} \cup \cdots \cup D^N_{non-iid}$. We compared the performance of FedAvg, FCS, and the proposed FL-QLMS with or without auxiliary model M_{aux}. As shown in Fig. 6, the FL-QLMS with an additional model has a similar performance as FedAvg, while both algorithms cannot keep up with FedCS with an average RMSE of 7.28. The reason for this is the robustness of FedAvg and FedCS against the Non-I.I.D setting to some extent. Also, compared to FedCS and FL-QLMS, FedCS allows two times as many clients in each training round. We then found that the average RMSE of FL-QLMS without an auxiliary model is higher than the other methods, reflecting the importance of an additional model during training.

We further investigated the impact of hacked clients on various FL algorithms. We refer to this scenario as **Scenario-II**. We assume that $k\%$ of all clients are hacked in each training round. Each hacked client uploads a malicious model where all parameters range from -1 to 1 randomly. Compared to **Scenario-I**, we used the same setting for data distribution and training simulation. From Fig. 7 we can see how each method performs against model attacks of varying severity. FL-QLMS (with M_{aux}) is shown to be robust when 10% to 40% of clients are hacked, holding average performance constant. In contrast, FedAvg and FedCS are highly sensitive to attacks, as the training process hardly converges when the number of faked models increases. For FL-QLMS (without M_{aux}), it always leads to convergence, but with slightly worse performance than FL-QLMS (with M_{aux}).

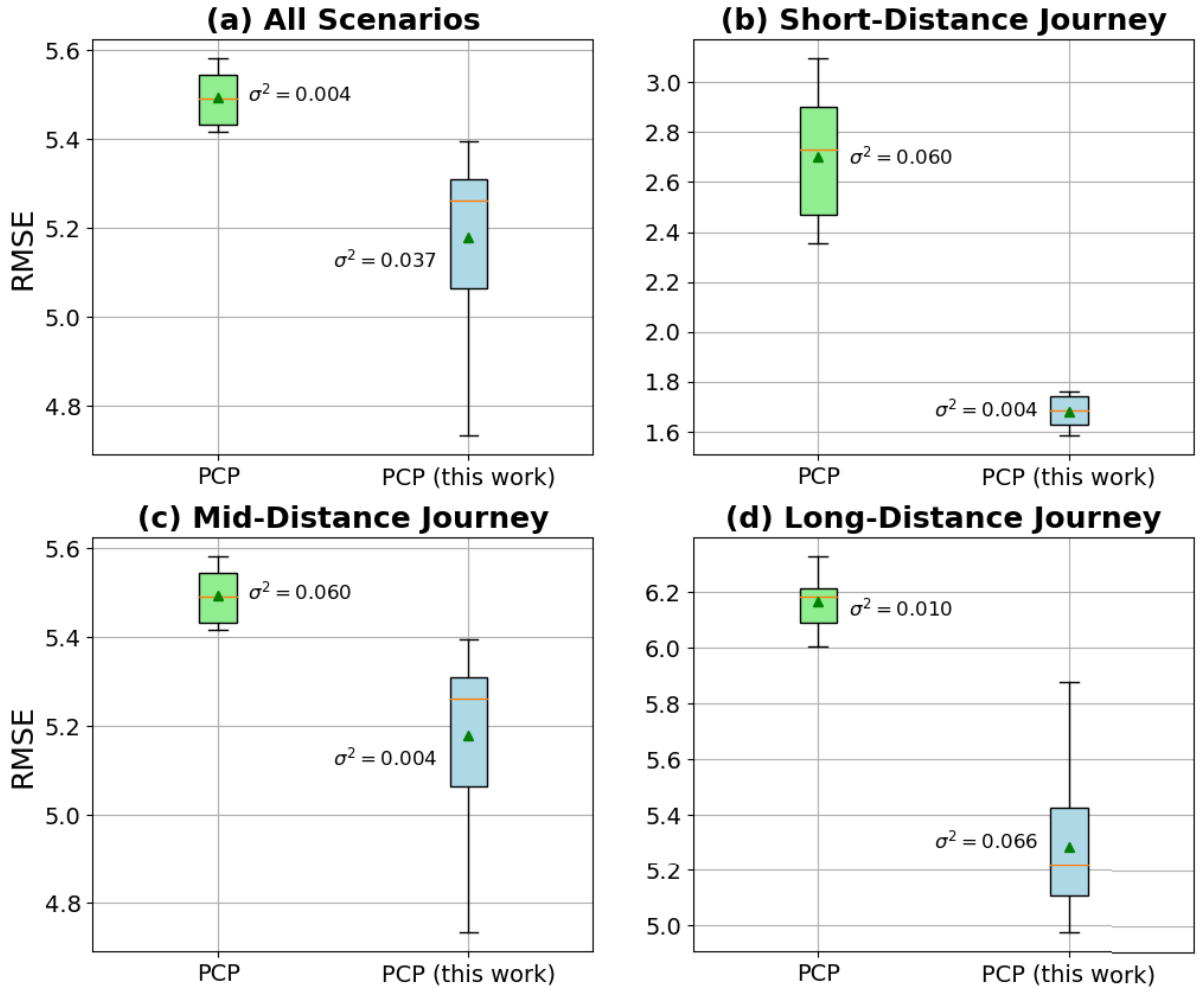


FIGURE 4. Comparison between PCP and the multi-stage PCP (this work) in different scenarios.

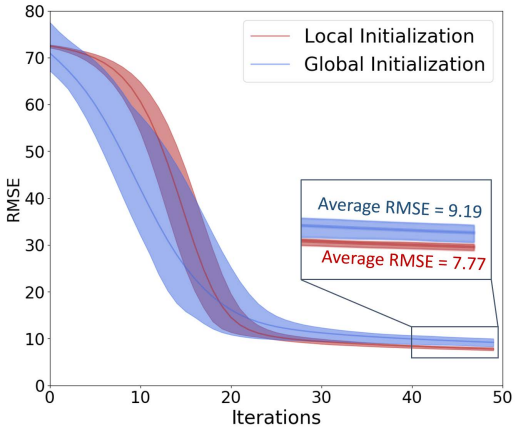


FIGURE 5. Comparison between two model initialization methods in federated learning. Shaded regions denote the fluctuation of the performance. The meaning of iteration is the number of times that the models were aggregated.

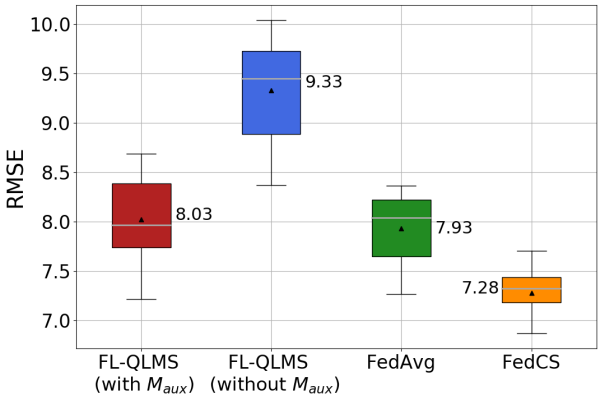


FIGURE 6. Comparison among FedAvg, FedCS [46], and the proposed FL-QLMS (w/o the auxiliary model). In this experiment, a Non-IID setting is considered. An average RMSE is shown beside each boxplot.

V. DISCUSSION

A semi-decentralized FL-based architecture is proposed to integrate both an aggregator and edge nodes into a blockchain platform. Although the blockchain does not secure the transmission of local models from the client to the server side, the aggregator could perform a robust model selection strategy to remedy potential model attacks during or before dispatch.

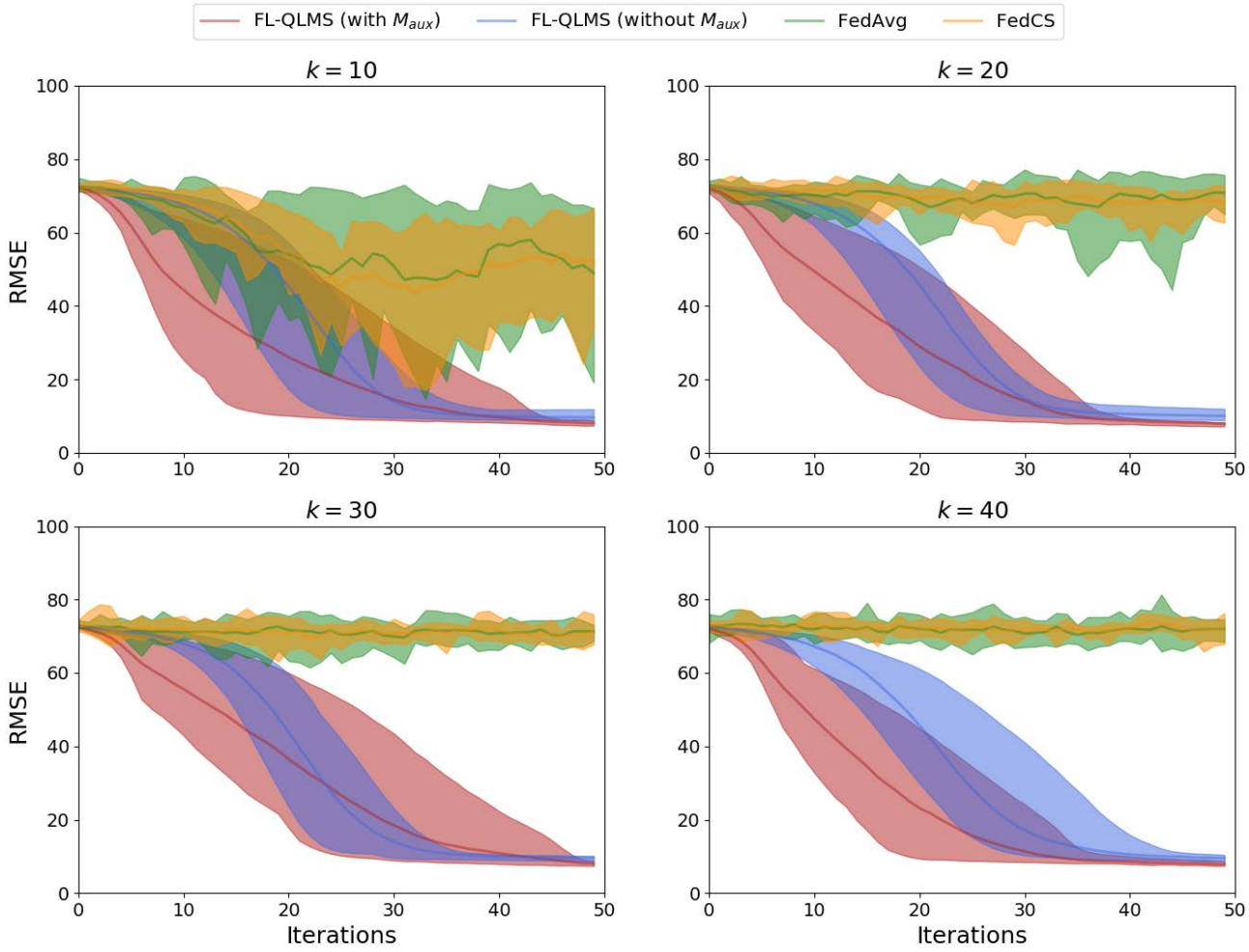


FIGURE 7. Comparison among FedAvg, FedCS, and the proposed FL-QLMS (w/o the auxiliary model) under client attacks. Different severity of the attack is considered. Shaded regions denote the fluctuation of the performance.

In this way, we significantly reduce the communication load on the blockchain and still maintain a robust and secure network. Using a simple AI model to predict battery power consumption for a long-distance trip is inappropriate for accurate prediction. Since a long journey can be divided into multiple small sections, a multi-stage algorithm helps reduce the prediction error. A shortcoming of our strategy lies in the assumption that the driving activity is a uniform linear motion, which is ideal in practice. To transform the process into a real scenario, we prefer to create an optimal route based on the global positioning system. Moreover, the efficient division of the whole trip into several sections remains a problem to be optimized. Besides, a qualified local model selection is essential to ensure the robustness of federated learning. The FL-QLMS algorithm demonstrates robustness against model attacks during the federated process. However, the performance of the current FL-QLMS algorithm is highly dependent on a prepared auxiliary data set, which raises two critical issues. First, the supplemental data should ideally have the same distribution as the entire data is not guaranteed. In addition, since the client-side data is updated daily, the ancillary information is unreliable for the local model selection. Second, due to privacy and security awareness, edge nodes may not share raw data to the server.

VI. CONCLUSION

This work presented a semi-decentralized Robust Network of Electric Vehicles (NoEV) integration system for power management in smart grid platform. NoEV integrates an aggregator with EV fleets into a blockchain framework, where EVs execute a multi-stage algorithm to predict EV power consumption using a novel FL-QLMS algorithm. In addition, we evaluated the proposed semi-decentralized system regarding storage and communication efficiency in the blockchain network. Compared to the previous approaches, NoEV requires 35% fewer transactions in short intervals and propagation delays. The comparison results show that the proposed system achieves better network efficiency while maintaining the system's security level. Moreover, the system achieves a 5.7% lower root mean square error (RMSE) than the

conventional PCP approach, significantly improving power consumption prediction. In addition, FL-QLMS approach outperforms state-of-the-art methods in terms of robustness to client-side attacks. The evaluation results demonstrate that the performance of FL-QLMS is not affected when 10% to 40% percent of the models are manipulated.

Nevertheless, the proposed system still has room to improve. First, the robustness of the system relies on the high stability of the aggregator. The aggregator collects local updates and broadcasts the global model to the blockchain. However, once the aggregator is not working, a backup server is needed to maintain the system. Second, in our blockchain proposal, EV clients are deployed to act as miners. If the local training needs a more powerful edge device, then the computing device on the EV may not be enough. Therefore, other miners need to be associated with EVs, and the system architecture needs to be redesigned.

In our future work, we plan to investigate the mining reward mechanism by extending our work to both public car-sharing and private car services. In addition, we will also study the security issue in more complicated attack scenarios.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their constructive comments, which helped us improve the manuscript.

REFERENCES

[1] *Renewable Capacity Statistics 2021*. Accessed: Mar. 3, 2022. [Online]. Available: https://www.irena.org/publications/2021/March/Renewable-Capacity-Statistics-2021

[2] *Road Transport: Reducing CO_2 Emissions From Vehicles*. Accessed: Mar. 3, 2022. [Online]. Available: https://ec.europa.eu/clima/eu-action/transport-emissions/road-transport-reducing-co2-emissions-vehicles/co2-emission-performance-standards-cars-and-vans_en

[3] G. Zhang, C. Jiang, and X. Wang, "Comprehensive review on structure and operation of virtual power plant in electrical system," *IET Gener., Transmiss. Distrib.*, vol. 13, no. 2, pp. 145–156, Jan. 2019.

[4] Y. Yang, Q.-S. Jia, X. Guan, X. Zhang, Z. Qiu, and G. Deconinck, "Decentralized EV-based charging optimization with building integrated wind energy," *IEEE Trans. Autom. Sci. Eng.*, vol. 16, no. 3, pp. 1002–1017, Jul. 2019.

[5] W. Wang, P. Chen, D. Zeng, and J. Liu, "Electric vehicle fleet integration in a virtual power plant with large-scale wind power," *IEEE Trans. Ind. Appl.*, vol. 56, no. 5, pp. 5924–5931, Sep. 2020.

[6] L. Liu, C. Chen, Q. Pei, S. Maharjan, and Y. Zhang, "Vehicular edge computing and networking: A survey," *Mobile Netw. Appl.*, vol. 4, pp. 1–24, Oct. 2020.

[7] J. Zhao, Q. Li, Y. Gong, and K. Zhang, "Computation offloading and resource allocation for cloud assisted mobile edge computing in vehicular networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 8, pp. 7944–7956, Aug. 2019.

[8] C. Sonmez, C. Tunca, A. Ozgovde, and C. Ersoy, "Machine learning-based workload orchestrator for vehicular edge computing," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 4, pp. 2239–2251, Apr. 2021.

[9] V. Chamola, A. Sancheti, S. Chakravarty, N. Kumar, and M. Guizani, "An IoT and edge computing based framework for charge scheduling and EV selection in V2G systems," *IEEE Trans. Veh. Technol.*, vol. 69, no. 10, pp. 10569–10580, Oct. 2020.

[10] Y. Huang, Y. Lu, F. Wang, X. Fan, J. Liu, and V. C. Leung, "An edge computing framework for real-time monitoring in smart grid," in *Proc. Int. Conf. Ind. Internet (ICII)*, 2018, pp. 99–108.

[11] H. Ko, S. Pack, and V. C. M. Leung, "Mobility-aware vehicle-to-grid control algorithm in microgrids," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 7, pp. 2165–2174, Jul. 2018.

[12] G. Qiao, S. Leng, K. Zhang, and Y. He, "Collaborative task offloading in vehicular edge multi-access networks," *IEEE Commun. Mag.*, vol. 56, no. 8, pp. 48–54, Aug. 2018.

[13] *RX 32-Bit Performming Efficiency MCUs*. Accessed: Mar. 3, 2022. [Online]. Available: https://www.renesas.com/us/en/products/microcontrollers-microprocessors/rx-32-bit-performance-efficiency-mcus

[14] M. A. Razzaque *et al.*, "Security and privacy in vehicular ad-hoc networks: Survey and the road ahead," in *Wireless Networks and Security*. Berlin, Germany: Springer, 2013, pp. 107–132.

[15] J. Cui, L. Wei, H. Zhong, J. Zhang, Y. Xu, and L. Liu, "Edge computing in VANETs–An efficient and privacy-preserving cooperative downloading scheme," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 6, pp. 1191–1204, Jun. 2020.

[16] J. A. Onieva, R. Rios, R. Roman, and J. Lopez, "Edge-assisted vehicular networks security," *IEEE Internet Things J.*, vol. 6, no. 5, pp. 8038–8045, Oct. 2019.

[17] G. Luo, H. Zhou, N. Cheng, Q. Yuan, J. Li, F. Yang, and X. Shen, "Software-defined cooperative data sharing in edge computing assisted 5G-VANET," *IEEE Trans. Mobile Comput.*, vol. 20, no. 3, pp. 1212–1229, Mar. 2021.

[18] A. M. Elbir, B. Soner, and S. Coleri, "Federated learning in vehicular networks," 2020, *arXiv:2006.01412*.

[19] 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.

[20] M. S. Alvim, M. E. Andrés, K. Chatzikokolakis, P. Degano, and C. Palamidessi, "Differential privacy: On the trade-off between utility and information leakage," in *Proc. Int. Workshop Formal Aspects Secur. Trust*. Berlin, Germany: Springer, 2011, pp. 39–54.

[21] A. Sheikh, V. Kamuni, A. Urooj, S. Wagh, N. Singh, and D. Patel, "Secured energy trading using byzantine-based blockchain consensus," *IEEE Access*, vol. 8, pp. 8554–8571, 2020.

[22] Y. Li and B. Hu, "A consortium blockchain-enabled secure and privacy-preserving optimized charging and discharging trading scheme for electric vehicles," *IEEE Trans. Ind. Informat.*, vol. 17, no. 3, pp. 1968–1977, Mar. 2021.

[23] J. Kang, R. Yu, X. Huang, S. Maharjan, Y. Zhang, and E. Hossain, "Enabling localized peer-to-peer electricity trading among plug-in hybrid electric vehicles using consortium blockchains," *IEEE Trans. Ind. Informat.*, vol. 13, no. 6, pp. 3154–3164, Dec. 2017.

[24] L. P. Qian, Y. Wu, X. Xu, B. Ji, Z. Shi, and W. Jia, "Distributed charging-record management for electric vehicle networks via blockchain," *IEEE Internet Things J.*, vol. 8, no. 4, pp. 2150–2162, Feb. 2021.

[25] V. Hassija, V. Chamola, S. Garg, N. G. K. Dara, G. Kaddoum, and D. N. K. Jayakody, "A blockchain-based framework for lightweight data sharing and energy trading in V2G network," *IEEE Trans. Veh. Technol.*, vol. 69, no. 6, pp. 5799–5812, Jun. 2020.

[26] Z. Yang, K. Yang, L. Lei, K. Zheng, and V. C. M. Leung, "Blockchain-based decentralized trust management in vehicular networks," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 1495–1505, Apr. 2019.

[27] U. Javaid, M. N. Aman, and B. Sikdar, "A scalable protocol for driving trust management in internet of vehicles with blockchain," *IEEE Internet Things J.*, vol. 7, no. 12, pp. 11815–11829, Dec. 2020.

[28] D. Gabay, K. Akkaya, and M. Cebe, "Privacy-preserving authentication scheme for connected electric vehicles using blockchain and zero knowledge proofs," *IEEE Trans. Veh. Technol.*, vol. 69, no. 6, pp. 5760–5772, Jun. 2020.

[29] L. Li, J. Liu, L. Cheng, S. Qiu, and W. Wang, "Creditcoin: A privacy-preserving blockchain-based incentive announcement network for communications of smart vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 7, pp. 2204–2220, Jul. 2018.

[30] Y. Wang, Z. Su, and N. Zhang, "BSIS: Blockchain-based secure incentive scheme for energy delivery in vehicular energy network," *IEEE Trans. Ind. Informat.*, vol. 15, no. 6, pp. 3620–3631, Jun. 2019.

[31] Z. Wang, M. Ogbodo, H. Huang, C. Qiu, M. Hisada, and A. B. Abdallah, "AEBIS: AI-enabled blockchain-based electric vehicle integration system for power management in smart grid platform," *IEEE Access*, vol. 8, pp. 226409–226421, 2020.

[32] A. B. M. Abdallah Hisada, "Virtual power platform control system," Japanese Patent 2020 033 678, Feb. 28, 2020.

[33] D. C. Nguyen, M. Ding, Q.-V. Pham, P. N. Pathirana, L. B. Le, A. Seneviratne, J. Li, D. Niyato, and H. V. Poor, "Federated learning meets blockchain in edge computing: Opportunities and challenges," *IEEE Internet Things J.*, vol. 8, no. 16, pp. 12806–12825, Aug. 2021.
[34] 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, Oct. 2021.
[35] Q. Wang, Y. Guo, X. Wang, T. Ji, L. Yu, and P. Li, "AI at the edge: Blockchain-empowered secure multiparty learning with heterogeneous models," *IEEE Internet Things J.*, vol. 7, no. 10, pp. 9600–9610, Oct. 2020.
[36] X. Bao, C. Su, Y. Xiong, W. Huang, and Y. Hu, "FLChain: A blockchain for auditable federated learning with trust and incentive," in *Proc. 5th Int. Conf. Big Data Comput. Commun. (BIGCOM)*, Aug. 2019, pp. 151–159.
[37] 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.
[38] Y. Qu, L. Gao, T. H. Luan, Y. Xiang, S. Yu, B. Li, and G. Zheng, "Decentralized privacy using blockchain-enabled federated learning in fog computing," *IEEE Internet Things J.*, vol. 7, no. 6, pp. 5171–5183, Jun. 2020.
[39] K. Vatanparvar, S. Faezi, I. Burago, M. Levorato, and M. A. Al Faruque, "Extended range electric vehicle with driving behavior estimation in energy management," *IEEE Trans. Smart Grid*, vol. 10, no. 3, pp. 2959–2968, May 2019.
[40] B. Gao, L. Guo, Q. Zheng, B. Huang, and H. Chen, "Acceleration speed optimization of intelligent EVs in consideration of battery aging," *IEEE Trans. Veh. Technol.*, vol. 67, no. 9, pp. 8009–8018, Sep. 2018.
[41] G. Ferro, M. Paolucci, and M. Robba, "Optimal charging and routing of electric vehicles with power constraints and time-of-use energy prices," *IEEE Trans. Veh. Technol.*, vol. 69, no. 12, pp. 14436–14447, Dec. 2020.
[42] D. Baek, Y. Chen, A. Bocca, L. Bottaccioli, S. D. Cataldo, V. Gatteschi, D. J. Pagliari, E. Patti, G. Urgese, N. Chang, A. Macii, E. Macii, P. Montuschi, and M. Poncino, "Battery-aware operation range estimation for terrestrial and aerial electric vehicles," *IEEE Trans. Veh. Technol.*, vol. 68, no. 6, pp. 5471–5482, Jun. 2019.
[43] L. Zhao, W. Yao, Y. Wang, and J. Hu, "Machine learning-based method for remaining range prediction of electric vehicles," *IEEE Access*, vol. 8, pp. 212423–212441, 2020.
[44] C. Gomez-Quiles, G. Asencio-Cortes, A. Gastalver-Rubio, F. Martinez-Alvarez, A. Troncoso, J. Manresa, J. C. Riquelme, and J. M. Riquelme-Santos, "A novel ensemble method for electric vehicle power consumption forecasting: Application to the Spanish system," *IEEE Access*, vol. 7, pp. 120840–120856, 2019.
[45] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. Conf. Artif. Intell. Statist.*, 2017, pp. 1273–1282.
[46] T. Nishio and R. Yonetani, "Client selection for federated learning with heterogeneous resources in mobile edge," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2019, pp. 1–7.
[47] S. Abdulrahman, H. Tout, A. Mourad, and C. Talhi, "FedMCCS: Multicriteria client selection model for optimal IoT federated learning," *IEEE Internet Things J.*, vol. 8, no. 6, pp. 4723–4735, Mar. 2021.
[48] Y. He, J. Ren, G. Yu, and J. Yuan, "Importance-aware data selection and resource allocation in federated edge learning system," *IEEE Trans. Veh. Technol.*, vol. 69, no. 11, pp. 13593–13605, Nov. 2020.
[49] J. Xu and H. Wang, "Client selection and bandwidth allocation in wireless federated learning networks: A long-term perspective," *IEEE Trans. Wireless Commun.*, vol. 20, no. 2, pp. 1188–1200, Feb. 2020.
[50] Y. Jee Cho, J. Wang, and G. Joshi, "Client selection in federated learning: Convergence analysis and power-of-choice selection strategies," 2020, *arXiv:2010.01243*.
[51] W. Zhang, X. Wang, P. Zhou, W. Wu, and X. Zhang, "Client selection for federated learning with non-IID data in mobile edge computing," *IEEE Access*, vol. 9, pp. 24462–24474, 2021.
[52] *Google Maps Cities Japan*. Accessed: Mar. 3, 2022. [Online]. Available: https://www.google.com/maps
[53] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated learning with non-IID data," 2018, *arXiv:1806.00582*.
[54] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.
[55] M. Alharby and A. van Moorsel, "BlockSim: An extensible simulation tool for blockchain systems," 2020, *arXiv:2004.13438*.
[56] *Japan Meteorological Agency*. Accessed: Mar. 3, 2022. [Online]. Available: http://www.data.jma.go.jp/gmd/risk/obsdl/index.php
[57] (2016). *Bitcoin Release 0.12.0*. Accessed: Mar. 3, 2022. [Online]. Available: https://bitcoin.org/en/release/v0.12.0

ZHISHANG WANG (Graduate Student Member, IEEE) received the B.S. degree in computer science from Wuhan University, China, in 2014, and the M.S. degree in computer science from the University of Freiburg, Germany, in 2019. He is currently pursuing the Ph.D. degree with the Adaptive Systems Laboratory (ASL), The University of Aizu. He is also a member of the ASL, The University of Aizu. His current research interests include machine learning systems, collaborative learning, blockchain, and trustworthy AI. He is also interested in event-driven neuromorphic systems targeted for a new generation of brain-inspired computing technologies and adaptive edge computing systems.

ABDERAZEK BEN ABDALLAH (Senior Member, IEEE) received the Ph.D. degree in computer engineering from The University of Electro-Communications, Tokyo, in 2002. He is currently a Full Professor of computer science and engineering at The University of Aizu, Japan. He has been concurrently the Head of the Division of Computer Engineering, School of Computer Science and Engineering, since April 2014. He researche computer systems, with an emphasis on adaptive/self-organizing systems, brain-inspired computing, interconnection networks, and AI-powered cyber-physical systems. His current research interests include neural processing systems, focusing on machine learning systems, spike-based neural network dynamics, and spike-based learning. He is a Senior Member of ACM.