Adaptive Traffic Prediction at the ITS Edge With Online Models and Blockchain-Based Federated Learning

Collin Meese, *Graduate Student Member, IEEE*, Hang Chen, Wanxin Li, *Member, IEEE*, Danielle Lee, Hao Guo, *Member, IEEE*, Chien-Chung Shen, *Member, IEEE*, and Mark Nejad, *Senior Member, IEEE*

***Abstract*— Managing urban traffic dynamics is critical in Intelligent Transportation Systems (ITS), where short-term traffic prediction is vital for effective congestion management and vehicle routing. While existing centralized deep learning (DL) models have achieved high prediction accuracy, their applicability is limited in decentralized ITS environments. The increasing use of connected vehicles and mobile sensors has led to decentralized data generation in ITS, presenting an opportunity to improve traffic prediction through collaborative machine learning. Recently, blockchain technology has shown promise in improving ITS efficiency, security, and reliability. In conjunction with blockchain, Federated Learning (FL) is a suitable approach to leverage online data streams in ITS; however, most research on FL for traffic prediction focuses on offline learning scenarios. This paper researches a blockchain-enhanced architecture for training online traffic prediction models using FL. The proposed approach enables decentralized model training at the edge of the ITS network, and extensive experiments used dynamically collected arterial traffic data shards as a case study to evaluate online learning performance. The results demonstrate that our online FL approach outperforms the per-device, non-federated baseline models for most sensors while maintaining a suitable execution time and latency for real-world deployment.**

***Index Terms*— Blockchain, deep learning, federated learning, online models, streaming models, traffic prediction.**

Manuscript received 21 February 2023; revised 14 August 2023 and 20 February 2024; accepted 26 March 2024. This work was supported in part by the Federal Highway Administration Grant "Artificial Intelligence Enhanced Integrated Transportation Management System" (2020–2023), in part by Guangdong Basic and Applied Basic Research Foundation (2021–2024) under Grant 2021A1515110286, in part by the Natural Science Foundation of Shaanxi Provincial Department of Education (2022–2023) under Grant 2022JQ-639, in part by the Basic Research Programs of Taicang (2022–2024) under Grant TC2022JC23, and in part by the Xi'an Jiaotong-Liverpool University (XJTLU) Research Development Fund (2023–2026) under Grant RDF-22-02-106. An earlier version of this paper was presented at the 2022 CCGrid: The 22nd IEEE/ACM International Symposium on Cluster, Cloud, and Internet Computing [DOI: 10.1109/CCGrid54584.2022.00041]. The Associate Editor for this article was V. Chamola. *(Corresponding author: Wanxin Li.)*

Collin Meese, Danielle Lee, and Mark Nejad are with the Department of Civil and Environmental Engineering, University of Delaware, Newark, DE 19716 USA (e-mail: cmeese@udel.edu; danilee@udel.edu; nejad@udel.edu).

Hang Chen and Chien-Chung Shen are with the Department of Computer and Information Sciences, University of Delaware, Newark, DE 19716 USA (e-mail: chenhang@udel.edu; cshen@udel.edu).

Wanxin Li is with the Department of Communications and Networking, Xi'an Jiaotong-Liverpool University, Suzhou 215123, China (e-mail: wanxin.li@xjtlu.edu.cn).

Hao Guo is with the Research and Development Institute, Northwestern Polytechnical University, Shenzhen 518057, China (e-mail: haoguo@nwpu.edu.cn).

Digital Object Identifier 10.1109/TITS.2024.3391053

I. Introduction

ACCURATE traffic prediction is crucial for Intelligent Transportation Systems (ITS) and serves as input for critical applications that mitigate the adverse impacts of traffic congestion, including additional fuel consumption, emissions, environmental effects, and unreliable travel time estimates [1]. In recent decades, there has been a steady increase in Vehicle Miles Traveled (VMT), leading to widespread traffic congestion; the 2019 Urban Mobility Report estimated that congestion in the United States resulted in an additional 8.8 billion hours of travel time and 3.3 billion gallons of additional fuel consumption [2]. Moreover, recent data from the Department of Energy indicates that VMT in 2022 has already reached pre-pandemic levels [3]. Accurate traffic prediction facilitates timely congestion resolution by providing drivers and stakeholders with informed dynamic rerouting and future route planning.

The success of deep learning (DL) models in traffic prediction is documented in the literatures [4], [5], and [6], achieving inference accuracy as high as 97%. However, existing approaches rely on offline training and centralized management, which requires collecting and aggregating substantial historical training data from data centers or the cloud. The substantial volume of data in ITS, which is increasingly decentralized due to the advent of mobile phones and connected vehicle sensors, may make conventional centralized cloud computing approaches inefficient. More specifically, the emergence of connected autonomous vehicle (CAV) and advances in Internet of Vehicles (IoV) have introduced low-latency online data streams; consequently, alternative methods, such as cooperative edge computing (CEC), are acquiring attention [7]. New methodologies are needed, specifically CEC-enabled decentralized online learning systems to effectively process and learn from the abundance of data continuously [8]. A paradigm shift in DL methodologies for traffic prediction is imperative to enable efficient, dynamic, and distributed online modeling that can effectively utilize traffic data streams.

Federated Learning (FL), where multiple participants collaborate in model training without exposing the underlying

data [9], has been investigated as a way to enable efficient and distributed knowledge sharing for various applications [10], [11], [12], including, for short-term traffic prediction (STP), empowering the model to be trained and updated dynamically in near-real-time on continuously collected incoming sensory data. Due to its decentralized nature, FL can also benefit from CEC by distributing the model training computation across a federation of edge devices and eliminating the need to share or aggregate local traffic data from each device. FL improves communication efficiency, reduces bandwidth consumption, and protects privacy for stakeholders making it a suitable candidate for matching the IoV environment and building decentralized and dynamic prediction models.

In addition, this study is motivated by the significant challenge of predicting traffic in transportation networks that suffer from data sparsity, particularly in scenarios where little to no historical data is available for offline model training. Deep learning-based traffic forecasting methods traditionally require an extensive database of network-wide sensing data and powerful computing hardware to learn an accurate representation of the traffic stream characteristics. Consequently, they face challenges in cold-start environments, creating a critical need for innovative approaches. To address this, we propose an online, FL-based approach, utilizing streaming models that can learn and adapt in near-real-time without requiring a pre-train or historical database.

Considering these observations, this paper proposes a decentralized, edge-compatible framework for online traffic prediction by integrating FL with a permissioned blockchain network. Specifically, we study simple, edge-compatible models and train them online using *Federated Averaging (FedAvg)* and live traffic data streams. The road side units (RSUs), paired with an edge device and its connected sensors, collects local traffic data from a fixed position on the roadway. The collected data can be sensed by a nearby fixed-position sensor or received by communicating with the onboard units of connected vehicles and cellular devices entering its communication range. After sufficient data is collected, a combination of new and recent historical data trains and updates localized online traffic prediction models. After each training update, RSUs shares their knowledge with the other devices by sending only the current model parameters instead of sharing the local traffic data. CEC distributes the training workload across the edge devices within the transportation network.

Blockchain offers a decentralized network technology that ensures integrity and security for all stored data, including local and global model updates throughout the FL process [13], [14], [15]. In recent ITS research, blockchain has shown the potential to enhance the efficiency, security, and reliability of communication and information sharing [16], [17], [18]. However, there is a lack of standardization and various competing standards in blockchain technology. This study proposes a design based on the permissioned blockchain approach, where a consortium of participants governs membership and data access rights [19], [20]. This choice contrasts with traditional public blockchains such as Bitcoin, which offer users unrestricted network access and participation in block consensus. The inherent properties of permissioned blockchain technology, including hierarchical permission levels, make it an ideal candidate for the FL framework, especially when managing mission-critical ITS with heterogeneous user groups (e.g., drivers, government agencies, regulators).

Together, FL and a permissioned blockchain network provide a new approach to train and update traffic prediction models online using live data streams from ITS while distributing the computation among data holders. Furthermore, online FL offers additional benefits of privacy preservation, decentralization, and low-latency model updates. On the contrary, existing static and centrally trained models require a significant delay in learning from the newly collected data [21]. The proposed system, integrating lightweight, edge-compatible FL and a permissioned blockchain to enable decentralized modeling in ITS, can help lay the foundation for many decentralized modeling applications due to its model- and data-agnostic design.

Contributions. This paper presents an online FL system for training traffic prediction models at the ITS network edge using live traffic data streams. This work makes the following contributions:

- We propose an online and secure-by-design FL framework for collaboratively training ITS models leveraging RSUs, lightweight edge computing devices, and a permissioned blockchain network. In contrast to a centralized server, the permissioned blockchain network decentralizes the information sharing between clients during the FL process, acting as the system controller. Specifically, the system is designed with the heterogeneous edge computing environment in mind, leveraging simple model architectures trained from a cold start without needing a historical database.
- To study online FL in ITS, contrary to existing works, [15], [22], [23], we design an online FL process that considers the location-specific and time-dependent datasets of each client. Specifically, we designed algorithms 1, 2, 3, and 4, to perform federated data preparation, time-dependent training batch formation, and local inference online using live ITS data streams. We simulate the online training and streaming data of the proposed FL system by coordinating the data collection, local training, and local inference operations of clients with the data collection period and the duration of the FL round.
- Compared to per-device baseline models, we conducted extensive experiments on two traffic datasets to analyze the model prediction accuracy of the proposed online FL system. We also measured the transaction latency and throughput of the permissioned blockchain network in this context using the Hyperledger Caliper [24] benchmarking tool and Hyperledger Fabric as an example framework. We conducted experiments on a resource-constrained edge device, Raspberry Pi 4 model B, to quantify the execution time for critical operations in the proposed system. The experimental results demonstrate that the system has a suitable response time for real-world deployment, and the example online FL models outperform the per-device baselines for 63.74% of

clients on average across both datasets. Additionally, the experiments demonstrated that the models are trainable by resource-constrained edge devices.

II. Related Work

A. Federated Learning (FL) in Transportation

FL is an emerging approach for facilitating collaborative machine learning in a distributed setting [25], and is gaining significant research attention in various application areas, including healthcare [26], Industrial Internet of Things (IIoT) [27], and IoV [28]. FL is a promising approach for improving traffic prediction models in ITS compared to centralized learning methods due to its inherent data privacy preservation and CEC properties, resulting in lower computation, network bandwidth, and storage requirements. Moreover, FL can harness many live data streams in smart cities and ITS in real-time to build dynamic and easy-to-update traffic models.

Participating edge devices use their collected data to train localized prediction models, and a centralized server periodically aggregates the resulting gradients to update the global prediction model. After updating, the new global model parameters return to the participants for further training. This process repeats sequentially over many rounds until a stopping condition. Notably, there exist three primary variations of FL where choice depends on the data partitioning: (1) horizontal federated learning (HFL) [29], [30], where the data exhibits identical features and is assumed to be independent and identically distributed (IID); (2) vertical federated learning (VFL) for scenarios where participants have various features describing the same data instances and a single participant is assumed to hold the target label [31]; (3) federated transfer learning (FTL) where HFL and VFL methods facilitate transfer learning using a pre-trained model from a related domain [32].

The traditional FL approaches still rely on a centralized server to perform gradient aggregation. Recently, blockchain technology has achieved true decentralization and safeguarded the FL system from the security and reliability drawbacks of utilizing a centralized cloud server. In blockchain-based designs, smart contracts typically compute the global model updates [33], [34], [35]. These approaches leverage the blockchain features of audibility, traceability, fault tolerance, and immutability to improve the robustness and security of FL systems.

In ITS research, FL is receiving increasing attention due to the growing number of dynamic and heterogeneous data feeds resulting from sensors, mobile phones, connected vehicles, and connected infrastructure systems [36]. In [37], the authors comprehensively analyze the potential benefits and feasibility of using FL-based learning methods instead of centralized learning for building IoV applications, concluding that FL methods can reduce the communication overhead by about 28 times compared to centralized learning and that FL facilitates knowledge sharing within the IoV [18]. The authors of [38] proposed an FL-empowered modeling, assignment, and reward scheme for distributed machine learning. In [15], the authors treat data sharing in the IoV as a multi-leader and multi-player game where the authors improve the traditional FL process by proposing a hierarchical FL and blockchain design to match the large-scale IoV environment. Moreover, the work of [39] improved the security and reliability of the FL gradient aggregation process for traffic prediction by integrating *FedAvg* with a permissioned blockchain consensus mechanism.

Despite much research in other related areas and strong theoretical applicability, FL is rarely studied in the traffic flow and speed prediction domains. In one of the existing studies, the authors of [22] proposed using FL to train a Gated Recurrent Unit (GRU) traffic prediction model by combining FL and a clustering-based ensemble scheme to group and sample participants before performing FL. Similarly, [23] also proposed a similar combination of models for traffic prediction; however, in this work, a blockchain network is used to coordinate the FL process in contrast to a centralized server. Reference [40] proposed a multi-task FL framework for traffic prediction and studied its application to route planning in ITS.

However, the prior works in ITS primarily focus on the offline FL setting where training and inference do not consider online data streams for dynamic updates and continuous training. Moreover, their experimental results do not analyze or report the varied performance among sensor locations, even though most traffic data are considered non-independent and identically distributed (non-IID). In other related research areas, such as Internet of Things (IoT), online FL systems that consider the potential non-IID data properties have been proposed.

For example, the authors of [41] developed an asynchronous FL system for training online streaming models from a cold start on non-IID data streams with an asynchronous model update framework. In addition, they further improved the model performance by considering the heterogeneity of the sensor clients and dynamically adjusted the learning step size accordingly. Lastly, they leveraged a centralized feature learner to help overcome the challenges of global model convergence compared with entirely centralized models. Their experimental results outline the benefits of their method compared to many baselines, including *FedAvg*, *FedProx* [42], and *FedAsync* [43]. Interestingly, their experiments highlight that *FedProx* and FedAsync struggle to consistently outperform *FedAvg* by a significant amount across multiple non-IID datasets despite having higher overhead than *FedAvg* in the online setting. This observation motivated us to focus on analyzing the performance of a lightweight system utilizing online *FedAvg* and experiment with traffic datasets, which the work above does not consider. Traffic data is considered non-IID due to the inherent spatiotemporal correlations amongst sensors and measurements, but the non-IID properties are interesting because they are stochastic and time-dependent.

B. Permissioned Blockchain

Blockchain technology encompasses two primary variants: permissioned (consortium) and permissionless [19]. In the context of permissionless blockchains, such as Bitcoin, membership and participation are unrestricted, resulting in a

transparent and open ledger of transactions [19]. Conversely, a permissioned blockchain provides programmable authentication and control schemes, wherein a consortium of one or more entities collaboratively manage membership, data access controls, and governance policies [44]. The ability to regulate membership in a permissioned blockchain system allows the utilization of lighter-weight consensus algorithms [45], improving network performance. Moreover, programmable access controls in these systems provide a means to regulate access to sensitive data, allowing more privacy protection [46]. These properties make permissioned blockchain technology more suitable for applications requiring high transaction throughput and underlying data protection.

Recent research has leveraged permissioned blockchain technology for a variety of different application areas, such as energy trading [47], IIoT [48], and healthcare [49]. In ITS, blockchain research has received attention as a secure-by-design system controller and networking layer for various IoV applications [13]. Reference [50] proposed a new delegated proof-of-stake consensus algorithm for permissioned blockchain networks, focusing on the data-sharing problem within the IoV. To establish trust among autonomous trucks in a mixed fleet environment, [14] presented a location-aware and privacy-preserving verification protocol based on zero-knowledge proof and permissioned blockchain.

A proposed blockchain-based signal control system was to protect data security in [51]. The authors applied this design to defend the Intelligent Traffic Signal System (I-SIG), a U.S. Department of Transportation-approved connected vehicle (CV) pilot program, against congestion attacks. To protect the security of 5G-ITS, [52] proposed the heterogeneous Blockchain-based Hierarchical Trust Evaluation (BHTE) strategy utilizing federated DL. Reference [53] designed a new blockchain-enabled certificate-based authentication scheme for vehicle accident detection and notification in ITS.

III. System Design

A. Problem Definition

FL-based traffic prediction is an online traffic forecasting problem in which a traffic prediction model is trained and evaluated using live ITS data streams. The transportation agency oversees the network and manages and operates a set of deployed RSUs, connected to various sensors (e.g., loop detectors, light detection and ranging (LiDAR) sensors, cameras) that act as traffic collection and communication devices. We propose to deploy an edge computing device alongside each RSU to provide resources at the edge of the network to perform model training and traffic prediction operations online with low communication latency compared to a centralized cloud server. We denote the combination of an RSU, the sensing device(s), and an edge device as a Road Side Edge Node (REN). Notably, each REN has local computation and storage capabilities that enable them to receive, process, and store additional data (e.g., speed, trajectory) and local models received from connected vehicles and mobile phones that pass within their communication zone. To execute the *FedAvg* algorithm, store, and retrieve local model updates, RENs interact with a blockchain network supported by back-end edge computing servers deployed in the fog.

Motivated by the design decisions of the Hyperledger Fabric blockchain framework, we define the entities of *clients*, *peers*, and *orderers* as the system participants. There are J clients, $\mathcal{C} = \{c_i | i = 1, \ldots, J\}$, who perform model training and interact with the blockchain network to store and retrieve data but do not expend additional resources on maintenance operations. K peers, $\mathcal{P} = \{p_i | i = 1, \ldots, K\}$, maintain a complete copy of the blockchain ledger and execute, endorse, and validate new transactions (e.g., state updates). Lastly, H orderers, $\mathcal{O} = \{o_i | i = 1, \ldots, H\}$, represent the participants responsible for consensus processing, ordering the new transactions into consistent data blocks, and distributing them to peers to ensure fault tolerance for the network.

Model training and data collection occur in synchronized and time-dependent rounds, $R = \{r_i | i = 1, \ldots, N\}$, where N is the total number of rounds. Synchronized and time-dependent rounds ensure that each client is training with a collection of new and recent historical data that is contextually relevant to the current network conditions. All notation is from the perspective of a single sensing client, client j. Each client device maintains a dynamic and online data set, D, updated at a time resolution of t as the client, c, collects a new sensor reading, x_t, in the field. Notably, x_t could be supplemented with additional field data communicated by nearby connected vehicles and mobile phone sensors. Consequently, D contains the traffic time series data collected from a fixed area of observation by a client, c. The length of the input sequence, τ, and the input vector at time t during the round, r_i, is defined as $\mathbf{X}_t = \{x_t, x_{t-1}, \ldots, x_{t-\tau+1}\}$. In this study, we use simple RNN models designed to take a vector, $\mathbf{X}_t$, as input and predict the next value in the sequence, x_{t+1}. An input vector and its target value are $(\mathbf{X}_t, y_t) \in D$ where $y_t = x_{t+1}$.

During the online FL process, devices collaboratively train a global prediction model, G_{r_i}, using their local data samples, only sharing the local model parameters between the devices in contrast to their owned data. G_{r_i} represents a time-dependent and online model updated during each FL round after τ new sensor readings. We denote the learnable weights of G_{r_i} by $w^G_{r_i}$. In any round, r_i, after collecting τ new measurements, c will add a new input data vector $\mathbf{X}_t$ to $D \in \mathbb{R}^{r_i \times \tau}$ and then form the online model training batch $M_{r_i} = \{\mathbf{X}_t, \mathbf{X}_{t-1}, \ldots, \mathbf{X}_{t-\beta+\tau}\}$ using the most recent β sensor readings and the sliding window method. This process for forming online training batches using a combination of newly collected and recent historical data helps alleviate overfitting during the online training process and reduces the computation requirements of model training for the lower-powered edge devices. Eq. 1 denotes the sliding window method used in this paper to create M_{r_i}. In particular, M_{r_i} is organized chronologically without shuffling and $|M_{r_i}| = (\beta - \tau)$ vectors of length $\tau + 1$.

$$M_{r_i} = \{X_{t+i-1} = [x_{t+i-1}, x_{t+i-2}, \cdots, x_{t+i-\tau}], \; y_t = x_{t+i} \mid i = 1, 2, \cdots, \beta - \tau\} \tag{1}$$

Similarly to G_{r_i}, we define the online local model of the client, c_j, during the round, r_i, as L_{r_i}, while

$W_{r_i}^{L}\{w_{r_i}^{L,j}|j=1,\ldots,J\}$ denotes the local model parameter set for all participants. The i^{th} parameter of the j^{th} participant's local model is indicated as $p_{r_i}^{i,j} \in w_{r_i}^{L,j}$. The weights for all models are updated using the Adaptive Moment Estimation (ADAM) optimizer with default parameters, and the mean squared error (MSE) function computes the training loss. An HFL approach computes the global model, where all participants have identical model architectures and hyperparameters, Δ. In this study, we leverage Recurrent Neural Networks (RNNs) to model the traffic time series. Specifically, we study lightweight models constructed using the GRU or long short-term memory (LSTM) variations of the RNN cell as example federated global models. GRU and LSTM are both improvements over traditional RNN cells. We selected these models to use as examples when analyzing the proposed FL system due to their simplified architectures that do not require a Graphics Processing Unit (GPU) for training. Furthermore, in the HFL setting, the number of learnable parameters is the same between the local models of all clients and the global model $|w_{r_i}^{L,j}| = |w_{r_i}^{G}|$. To compute the update of the global model, $w_{r_i}^{G}$, in each FL round, r_i, we utilize the *FedAvg* algorithm [9] according to Eq. 2.

$$\forall(p_{r_t}^{i,G} \in w_{r_i}^{G}) : p_{r_{i+1}}^{i,g} \longleftarrow \sum_{j=1}^{J} \frac{1}{J} p_{r_i}^{i,j} \tag{2}$$

During training and inference, we consider multiple error metrics for quantifying the performance of model predictions, including mean absolute error (MAE) (Eq. 3) and root mean squared error (RMSE) (Eq. 4). During training, the MSE function is the loss function, and MAE and RMSE compute the real-time inference errors for all predictions of all clients during each round, r_i.

$$L(y, \hat{y}) = \frac{1}{N}\sum_{i=0}^{N}(y_i - \hat{y}_i) \tag{3}$$

$$L(y, \hat{y}) = \sqrt{\frac{\sum_{i=0}^{N}(y_i - \hat{y}_i)^2}{N}} \tag{4}$$

where $L(y, \hat{y})$ is the loss or error of the predicted values, $\hat{y}$, compared to the ground truth data points, y, and $N = \tau$ is the total number of prediction points for each FL round controlled by τ.

B. System Architecture

Fig. 1 illustrates the architectural components of the system, including their operations, relationships, and interactions, and maps the participants and their roles to the permissioned blockchain network. Additionally, Fig. 1 depicts the step-by-step workflow and operations during a single FL round from the perspective of an individual client device (e.g., REN). We propose to use live data streams from existing sensing and connectivity infrastructure as input for training and updating of FL models online as new sensing data becomes available. Within the scope of our study, devices including RENs, mobile edge servers, and base stations are owned and operated by a trusted transportation authority, such as the traffic management center (TMC), or its partners, who are responsible for maintaining the ITS.

Algorithm 1 Operations of All Clients $c \in \mathcal{C}$ in Round r_i

for all clients $c \in \mathcal{C}$ in round r_i of R in parallel **do**
  D', $\mathbf{E}_{r_i}^{G}$, $\mathbf{E}_{r_i}^{B} \leftarrow c$.LIVE_SENSE($D$, $G_{r_{i-1}}$, $B_{r_{i-1}}$, r_i)
  M_{r_i}, $\mathbf{X}_{r_i}$, $\mathbf{Y}_{r_i} \leftarrow c_j$.FORM_BATCH($D'$, β)
  $w_{r_i}^{L} \leftarrow c$.FIT($G_{r_{i-1}}$, $\mathbf{X}_{r_i}$, $\mathbf{Y}_{r_i}$)
  $B_{r_i} \leftarrow c$.FIT($B_{r_{i-1}}$, $\mathbf{X}_{r_i}$, $\mathbf{Y}_{r_i}$)
  $tx_{r_i}^{1} \leftarrow c$.TRANSACTION(HDF5($w_{r_i}^{L}$), Δ)
  $etx_{r_i}^{1} \leftarrow c$.SENDTOPEERS($tx_{r_i}^{1}$, $\mathcal{P}$)
  c.SENDTOORDERERS($etx_{r_i}^{1}$, $signature_j$, $\mathcal{O}$)
  $tx_{r_i}^{2} \leftarrow c$.TRANSACTION(FEDAVG($\Delta$))
  $etx_{r_i}^{2} \leftarrow c$.SENDTOPEERS($tx_{r_i}^{2}$, $\mathcal{P}$)
  c.SENDTOORDERERS($etx_{r_i}^{2}$, $signature_j$, $\mathcal{O}$)
  $w_{r_i}^{G} \leftarrow c$.RECEIVEMODEL($\mathcal{P}$)
  $G_{r_i} \leftarrow$ REPLACE_WEIGHTS($w_{r_i}^{G}$)
  $D = D'$
end for

In line with Fig. 1, the following devices participate in the blockchain-based FL process:

- **Base Station** is a radio receiver/transmitter that serves as the hub of the local wireless network and can be the gateway between the REN, the wireless network, the mobile edge servers, and other components.
- **Mobile Edge Server** is a high-performance computing entity that can collaborate to perform blockchain tasks deployed in the fog at data centers.
- **Permissioned Blockchain** network manages the FL process by providing an immutable and distributed storage system to record and retrieve local and global model updates and also to distributively compute *FedAvg* to update G_{r_i}.
- **Peer Node** is a blockchain entity that performs transaction execution, endorsement, and block validation. They maintain a replica of the ledger state.
- **Orderer Node** is a blockchain entity that executes consensus procedures to create an unambiguous order of state updates (transactions), ensuring a consistent ledger state and providing fault tolerance.
- **REN** is an FL and blockchain *client* in our system and collects and stores traffic data from the static detection area along the road. They provide network connectivity and edge computing resources to process and store streaming data, update FL models, and perform online inference to predict future traffic conditions. It consists of an RSU to provide connectivity infrastructure, the associated road sensor, and an edge computing device.

RENs maintains a current copy of the global traffic prediction model and continuously updates it with the collection of new data. Although the system can operate with any HFL-compatible model, we limit the scope of this study to the analysis of lightweight constructions of GRU and LSTM cells for time series prediction. The peer and orderer nodes carry out the blockchain operations, represented as mobile edge servers

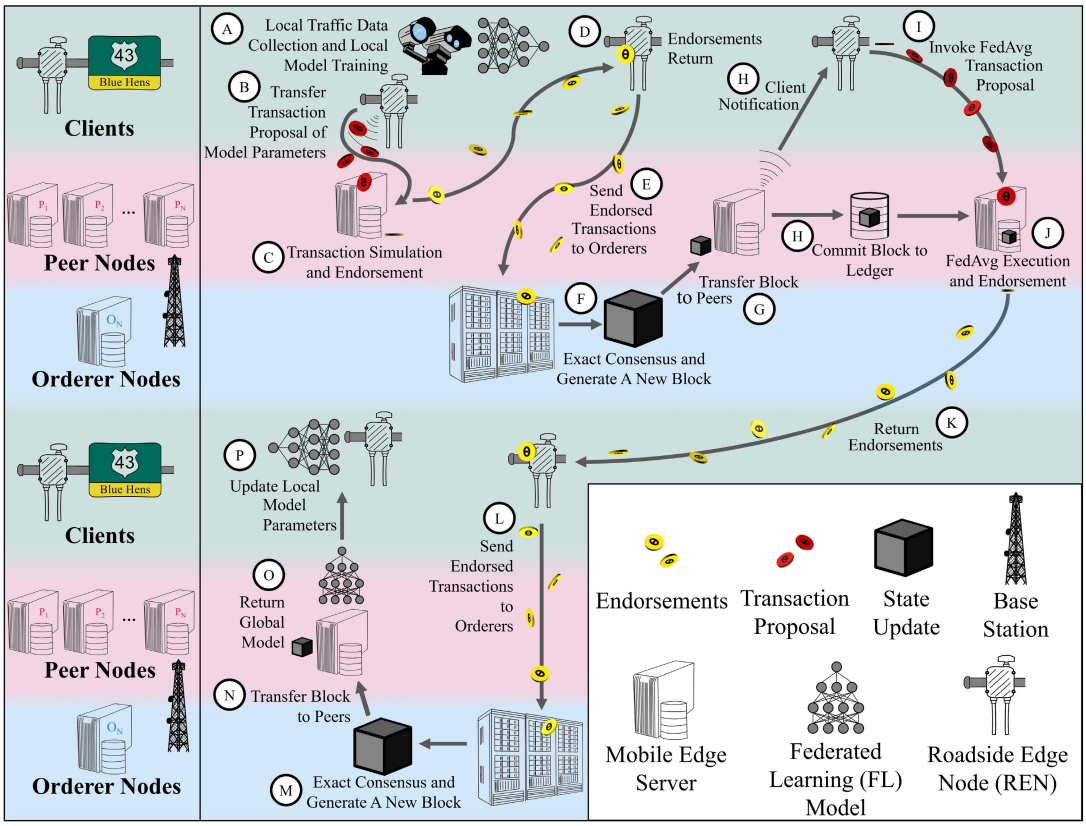


Fig. 1. The proposed system architecture.

deployed along with the base stations for communication. An edge server is configurable to run as a peer, orderer, or both depending on available resources.

C. Online Federated Learning (FL) for Traffic Prediction

In the proposed system, we continuously train federated time series models online using live traffic data streams, resulting in flexible and efficient traffic prediction models. During the online FL process, clients, $\mathcal{C}$, collaborate to train a single continuously updated global prediction model, G_{r_i}, using *FedAvg*, leveraging their local traffic information incoming from live sensor data streams, in a series of communication rounds, R. Fig. 1 illustrates the workflow from the perspective of a single client device during any FL round. The formalization of client operations is in Algo. 1.

During any round, r_i, every client, $c \in \mathcal{C}$, performs the following sequence of operations in parallel: (1) collect local data; (2) form the round-dependent online training batch M_{r_i} using the last β data points and the sliding window method; (3-4) train both $G_{r_{i-1}}$ and $B_{r_{i-1}}$ with local data; (5) generate a transaction proposal with the appropriate payload (updated model parameters at this step); (6) send the transaction proposal to all peers; (7) upon receiving endorsements, package the received endorsements into a transaction, digitally sign it and submit the transaction to the orderers; (8) wait until recording the local models of all clients on the blockchain, or until reaching the round timeout value; (9) execute the blockchain transaction workflow again to invoke *FedAvg*; (10) receive the updated global model parameters from the blockchain peers; (11) replace the model weights with the updated weights. Steps (5)-(7) represent the blockchain transaction workflow as referenced in the subsequent steps. Transaction processing is modeled on the execute-order-validate method used in Hyperledger Fabric, which separates execution and ordering operations to improve parallelization and performance.

Initially, each client, c, has an identical model, G_{r_0}, before the first round, r_1. In this paper, we study the proposed system in a cold start environment, where G_{r_0} and B_{r_0} are randomly initialized before being distributed to each client c; however, a pre-trained model could be used to accelerate the learning process. Each FL round occurs in two distinct phases: (1) each client executes Algo. 2, *Live_Sense()*, in parallel, to perform data collection and inference in a loop until τ new data points are collected; (2) clients train their local copy of the global model from the previous round, $G_{r_{i-1}}$, as well as their per-device baseline model, $B_{r_{i-1}}$, using a combination of the new and recently collected historical data controlled by β. We collect $(2 * \tau)$ data instances in r_1 because the model needs $(\tau + 1)$ data points to train, where τ controls the round duration. Online inference begins in r_2 after the first round of model training is complete.

After data collection and inference using Algo. 2, each client executes Algo. 3 to form the round-dependent online training batch, M_{r_i}, using the most recent β data instances. The parameter β is introduced to mitigate the overfitting of older data during the online learning process and reduce the

Algorithm 2 LIVE_SENSE(): Online Traffic Data Collection and Inference of All Clients $c \in \mathcal{C}$ in Round r_i

Input: D, $G_{r_{i-1}}$, $B_{r_{i-1}}$, r_i
Output: D', $\mathbf{E}^G_{r_i}$, $\mathbf{E}^B_{r_i}$
for all clients $c_j \in \mathcal{C}$ in round r_i of R in parallel **do**
  $n = 1$
  $\mathbf{X}_{t+1}$ =[]
  if $i == 1$ **then**
    while $n <= (2 * \tau)$ **do**
      $x_{t+n} \leftarrow c$.COLLECT_DATA()
      $\mathbf{X}_{t+1}$.APPEND(x_{t+n})
      $n = n + 1$
    end while
  else
    $y^G_{r_i}$ =[],
    y^{TRUE} =[] ▷ Temporary vector.
    $\mathbf{X}_t = D[-\tau :]$
    $\mathbf{X}_{t+1} = D[-\tau :]$
    while $n <= \tau$ **do**
      $y^G_{r_i}$.APPEND(c.PREDICT($G_{r_{i-1}}$, $\mathbf{X}_{t+1}$))
      $y^B_{r_i}$.APPEND(c.PREDICT($B_{r_{i-1}}$, $\mathbf{X}_{t+1}$))
      $x_{t+n} \leftarrow c$.COLLECT_DATA()
      $\mathbf{X}_{t+1}$.APPEND(x_{t+n})
      y^{TRUE}.APPEND(x_{t+n})
      $\mathbf{X}_{t+1}$.POP_LEFT()
      $n+ = 1$
    end while
    $\mathbf{E}^G_{r_i} \leftarrow$ CALC_ERROR(y^{TRUE}, $y^G_{r_i}$)
    $\mathbf{E}^B_{r_i} \leftarrow$ CALC_ERROR(y^{TRUE}, $y^B_{r_i}$)
  end if
  $D \leftarrow D \cup \mathbf{X}_{t+1}$
end for

Algorithm 3 FORM_BATCH(): Online Training Batch Formation for Client $c_j \in \mathcal{C}$ in Round r_i

Input: D, β
Output: M_{r_i}, $\mathbf{X}_{r_i}$, $\mathbf{Y}_{r_i}$
M_{r_i} = []
if $|D| < \beta$ **then**
  $\beta = |D|$
end if
$D' = D[-\beta :]$
for $i = \tau$ to β **do**
  M_{r_i}.APPEND($D'[i - \tau : i + 1]$
end for
$\mathbf{X}_{r_i} = M_{r_i}[:, : -1]$
$\mathbf{Y}_{r_i} = M_{r_i}[:, -1]$

Algorithm 4 Feed-Forward Inference of Client c in Round r_i Where $i > 1$

Input: $G_{r_{i-1}}$, D
Output: y'_{r_i}
$n = 1$
$\mathbf{X}_t = D[-\tau :]$
$\mathbf{X}' = \mathbf{X}_t$ ▷ Copy input array.
y'_{r_i} =[] ▷ Empty prediction array.
while $n <= \tau$ **do**
  $x_{t+n} \leftarrow c$.PREDICT($G_{r_{i-1}}$, $\mathbf{X}'$)
  y'_{r_i}.APPEND(x_{t+n})
  $\mathbf{X}'$.POP_LEFT()
  $\mathbf{X}'$.APPEND(x_{t+n})
  $n = n + 1$
end while

computational training demand. Sensing and field data are stored locally on each REN client and not on the blockchain network, which stores model updates exclusively.

D. Feed-Forward (FF) Inference Approach

Online traffic prediction systems provide critical input required by many ITS applications; however, they often require time-dependent predictions. For example, in the vehicle routing problem, predicting traffic along nearby roadways in the immediate term is necessary while also considering future traffic conditions at distant locations between the origin and destination points. We propose a feed-forward (FF) inference approach for generating multi-step predictions within the online learning framework to extend the forecast horizon of single-inference FL models.

Fig. 2 illustrates the FF approach in the system and Algo. 4 outlines the steps for producing the FF predictions. We designed FF to be executed dynamically by clients as needed and, as a result, it is not in the round-specific workflow defined in Algo. 2. At the start of execution, the client first extracts τ recent data points from D and stores them in a temporary input vector, $\mathbf{X}'$. $\mathbf{X}'$ will be modified during the FF process and used as input to produce predictions with $G_{r_{i-1}}$. Next, c generates a 5-minute look-ahead prediction, x_{t+n}. x_{t+n} is appended to both the prediction output vector, y'_{r_i}, and the input vector, $\mathbf{X}'$. The oldest historical measurement, $x_{t+n-\tau}$, is removed from the vector. This process is repeatable τ times to produce τ look-ahead predictions at any time t during round r_i. Finally, after the sensor collects future ground truth data, the FF inference error can be computed to quantify the performance of the method using y'_{r_i}.

E. Permissioned Blockchain Network

In contrast to a cloud server or data center, we propose to use a permissioned blockchain network to provide the communication and storage infrastructure in the FL system. The motivation to select the permissioned variant of blockchain technology stems from its membership and data access control properties, lighter weight consensus, lack of transaction fees, and modular design, providing increased deployment flexibility compared to public blockchains (e.g., Ethereum). Consequently, the permissioned variant of blockchain technology can better match the ITS environment where existing authorities, such as the TMC and the Department of Motor Vehicles (DMV), manage the transportation system and user identities are known [11], [12], [15], [17], [18].

We modeled the system's blockchain transaction processing workflow based on the design principles adopted by

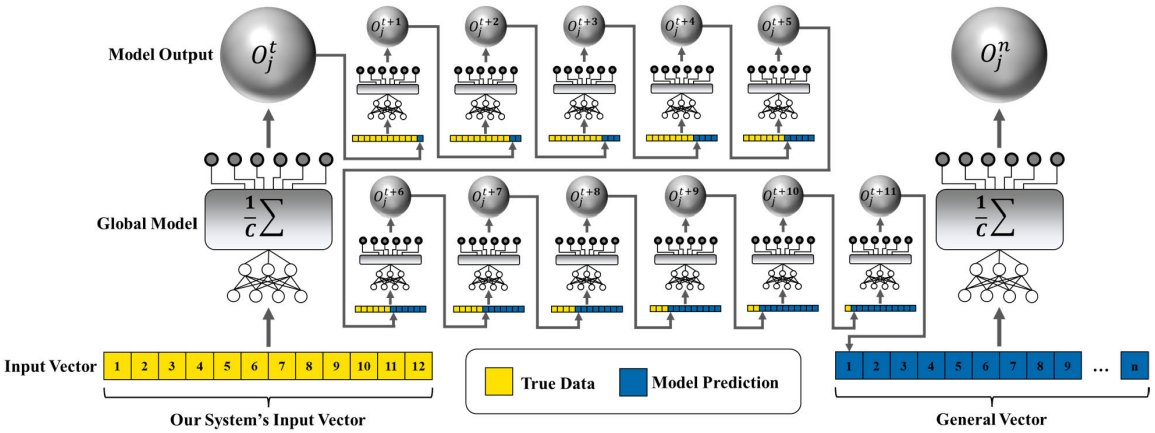


Fig. 2. Illustration of an example feed-forward approach with $\tau = 12$.

Hyperledger Fabric [54], a widely supported and open-source enterprise blockchain network software. Specifically, we adopt peer and orderer node classifications and the execute-order-validate transaction model, where only a subset of blockchain nodes are required to execute transactions before they are ordered and subsequently validated by the remaining nodes. This approach allows for parallelizing parts of the transaction processing workflow and improving network performance. The peer nodes endorse and validate transactions while maintaining an updated replica of the ledger state. On the contrary, the orderer nodes are only responsible for consensus on ordering state updates to ensure fault tolerance.

In our blockchain-based FL system, the permissioned blockchain manages the FL workflow by providing an immutable and distributed storage system to record and retrieve local and global model updates. It provides distributed computing resources to compute *FedAvg* and update the global model during each FL round. Within the workflow, the blockchain nodes (peers and orderers) are responsible for processing and storing the transactions, tx_{r_i}, which contain updates to the local model parameters of the client devices. Critical information, including model parameters in HDF5 format and associated metadata on the FL process, is encapsulated within a data structure for storage. The historical FL outputs can be retrieved and audited for quality assurance, preservation auditing, and other purposes by keeping a historical record of all clients' local model updates. The data structure used to store the updates of the local and global model within the blockchain network is as follows:

```
type ModelUpdate struct {
    FederatedID          string
    DetectorID           string
    Round                int
    ModelParameters      HDF5
    GlobalModel          HDF5
    Operator             string
}
```

The parameters of the $ModelUpdate$ data structure are defined as follows: $FederatedID$ represents the identifier associated with a specific FL process (e.g., "LSTM I-95"); $DetectorID$ is the unique client identifier; $Round$ denotes the FL round index for a given model update; $ModelParameters$ contains the model parameters in HDF5 format; and $Operator$ refers to the operation to be executed (e.g., PUT or GET model.

IV. Experimental Results

A. Experimental Design

1) Setup: We conducted the modeling experiments on Google Colab with one NVIDIA Tesla V100 GPU, two Intel(R) Xeon(R) Central Processing Units (CPUs) @ 2.30GHz, and 13 gigabytes of Random Access Memory (RAM). In all experiments, there is a one-to-one mapping of sensor data streams to RENs. We simulate the online ITS data streams by having each client, c, dynamically creating an updated training batch, M_{r_i}, for each r_i, using Algo. 3 and Algo. 2, which includes only the most recent historical data as defined by β. Clients execute 5 local training epochs in each FL round during training. To better understand and analyze the performance of the proposed online FL system, G_{r_0} is randomly initialized, in contrast to providing a pre-trained model at r_0. All clients utilize the *FedAvg* algorithm to compute the parameters of the global model. In our cast study, we trained federated LSTM and GRU models as an example model architecture for G. Notably, the edge device execution time experiments are conducted on a Raspberry Pi 4 model B having a 1.5GHz Advanced RISC Machines (ARM) processor and 8 gigabytes of RAM running the Ubuntu OS.

The blockchain experiments were conducted on a virtual machine provisioned with 24 gigabytes of RAM and eight cores of an Intel(R) Core(TM) i9-10900K CPU @ 3.70GHz. We simulated the proposed system of edge devices on Hyperledger Fabric version 2.2, using Docker containers to represent the blockchain peers and orderers. The design of each Docker container simulated low-resource edge devices. Accordingly, we set resource restrictions for all Docker containers, providing each peer container with 2 gigabytes of RAM. 50% of the clock cycles are available to each orderer container; the orderer containers are each allocated a maximum of 4 gigabytes of RAM. For each blockchain experiment, we instantiate

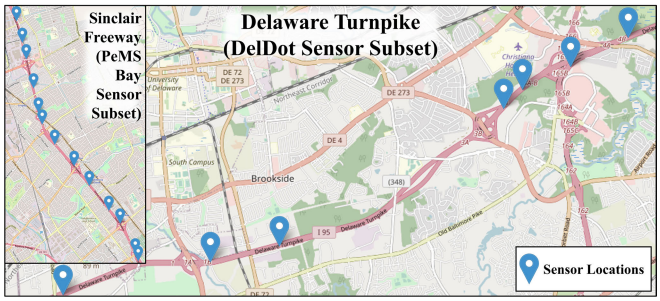


Fig. 3. The selected study area for the experiments.

four peers and five orderers. All experiments use raft [55] consensus protocol. The Hyperledger Caliper benchmarking tool assesses the transaction throughput and latency of the blockchain network under different input loads.

2) Datasets: The online FL experiments were performed on two different data sets: (1) Delaware Department of Transportation (DelDOT) traffic flow data collected from a major arterial (I-95); and (2) a subset of the widely used Performance Measurement System for the San Francisco Bay Area (PeMS Bay) reference dataset containing traffic speed measurements from a complex highway network. Fig. 3 displays the locations of the RENs within each study area. For all experiments, we simulate 1,165 FL rounds, which is equivalent to about eight weeks of continuous online sensing and learning.

DelDOT traffic flow dataset [56] includes traffic flow data collected from DelDOT-maintained roadways at a time resolution of 5-minutes. We selected seven LiDAR detectors along the I-95 north arterial to simulate RENs as the FL clients for the first set of experiments. The dataset for each REN contains eight weeks of location-specific flow data from the beginning of August 2019 until the end of September 2019.

PeMS Bay dataset [57] contains traffic speed measurements collected by 325 fixed-position sensors in a large arterial network with a time resolution of 5-minutes. A subset of 13 sensors is selected as data providers for the 13 REN clients. The dataset for each REN contains eight weeks of location-specific speed data from the beginning of January 2017 until the end of February 2017.

3) Models: In our case study, we experiment with simple constructions of RNN-based models as an example to demonstrate that the system can train simple, edge-compatible models with online FL. Related research has shown the success of RNN-based models (e.g., LSTM and GRU cells) for accurately predicting traffic flow using sensor data collected along major arterial roadways [22], [23], [56]. Motivated by these findings, we analyze the example GRU and LSTM models during the online FL experiments, subsequently referred to as *GRU-Fed* and *LSTM-Fed*. To accommodate the limited computational resources inherent to RENs, the designs of the *GRU-Fed* and *LSTM-Fed* are shallow, two-layer RNN models. Specifically, both federated models consist of two RNN cells (GRU or LSTM), with the output produced by a fully connected layer. Based on our previous work [56], we initialize each LSTM layer with a hidden size of 128, with 50 for the GRU layers. A dropout layer with $p = 0.2$ resides between the last RNN cell and the final output layer to alleviate overfitting during online learning. The experiments use the ADAM optimizer with the default learning rate to train the models, and the output layer uses a Rectified Linear Units (ReLU) activation function.

In all online FL experiments, each REN trains its per-device baseline model, B_{r_i}, in addition to the federated model, G_{r_i}, using the same dynamically updated univariate sensor data, D. In particular, B_{r_i} has a construction identical to G_{r_i} to provide a fair comparison between the per-device baseline and the proposed online FL system. We denote B_{r_i} as either *GRU-Base* or *LSTM-Base*. Specifically, each FL round, c trains both $G_{r_{i-1}}$ and $B_{r_{i-1}}$ for five local epochs using the same online batch, M_{r_i}; however, the baseline model, B_{r_i}, of each client will be trained again in the subsequent round, r_{i+1}. In contrast, the new global model, G_{r_i}, will replace the local models of all clients after performing *FedAvg* when it progresses to subsequent rounds. By comparing the baseline and federated models, we can assess the effectiveness of FL for online traffic prediction in ITS.

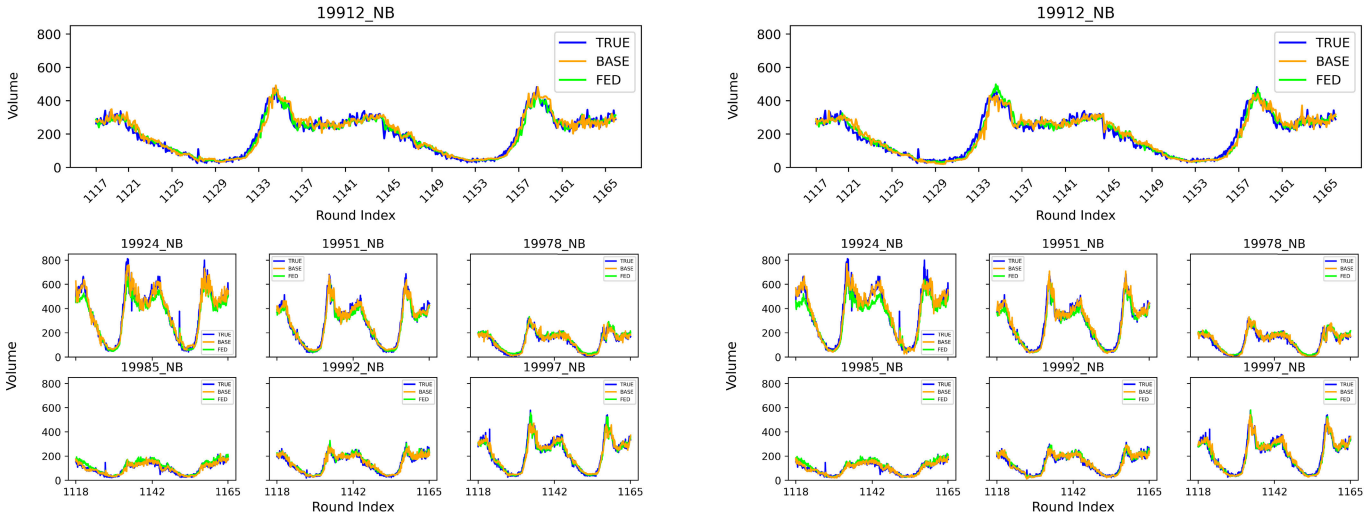
In the experiments, all federated and baseline RNN models have a 12 neuron input layer with a single neuron in the output layer. Given a data resolution of 5-minutes, this equates to 1 hour of traffic data as input to produce a single 5-minute look-ahead prediction. $\tau = 12$ and the FL rounds operate hourly as controlled by τ. Each REN client, c, collects τ new sensor readings and produces τ look-ahead predictions in any round, $r_i > 1$, prior to training $G_{r_{i-1}}$ and $B_{r_{i-1}}$. Representing the input data vector at time t is $\mathbf{X}_t$, which is created dynamically from the online dataset, D, in any round, r_i, using Algo. 2. As D is initially devoid of content on each client, the models undergo training from their foundational state. We collect $(2*\tau) = 24$ data points during the first round, r_1, to provide sufficient input for training, where inference begins in round r_2.

After designing the four models (i.e, *LSTM-Base*, *LSTM-Fed*, *GRU-Base*, and *GRU-Fed*), we perform experiments with $\beta = 72$ as the sliding window size for online batch creation influences the prediction accuracy [56]. β is similar to the batch size in centralized and offline learning, as it controls the number of training vectors in each epoch, as well as the temporal window size for training (Algo. 3) when dynamically forming the online training batch, M_{r_i}. We report the performance of the federated and baseline models with $\beta = 72$ in this work for comparative analysis.

B. Performance on Real-Time Inferences

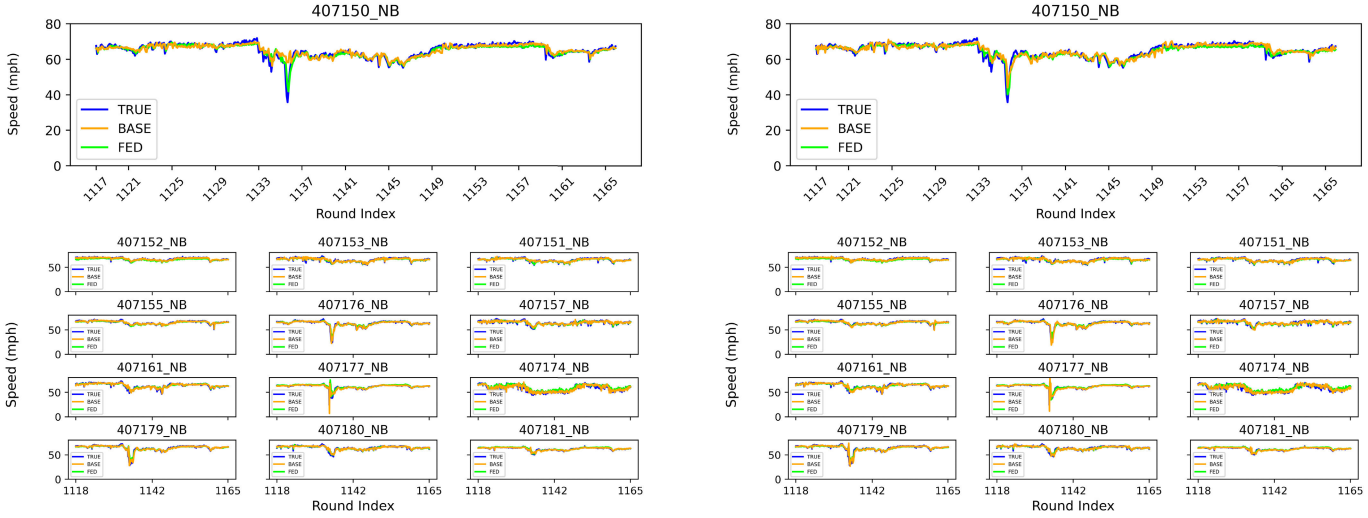
Figs. 4 and 5 illustrate the partial online inference curves of the four models for all clients in the experimental groups, DelDOT and PeMS Bay, with $\beta = 72$. Of the 1,165 rounds, we report the inference values for the final 48 rounds in our simulations, equivalent to two days of online traffic data collection and inference. Algo. 2 describes how we obtained the inference values.

For each subfigure of Fig. 4, sensor *19912_NB* acts as the primary reference detector, while sensor *407150_NB* serves as a reference detector for Fig. 5. For supplementary reference,



(a) LSTM with $\beta = 72$. (b) GRU with $\beta = 72$.

Fig. 4. Online inferences curves of all 7 clients during the last 48 FL rounds using the DelDOT dataset.



(a) LSTM with $\beta = 72$. (b) GRU with $\beta = 72$.

Fig. 5. Online inferences curves of all 13 clients during the last 48 FL rounds using the PeMS Bay dataset.

we provide smaller plots for the remaining sensors in each experimental group. The x-axis represents the FL round index for the final 48 rounds, while the y-axis indicates the traffic variable. Each client collects twelve new data points and produces twelve single-step inferences during each FL round. For example, a round index 1165 represents r_{1165}. The predictions plotted after index 1165 are the ground truth data, the baseline model inferences, and the federated model inferences for FL round 1165, denoted as $TRUE^{r_{1165}}$, $BASE^{r_{1165}}$ and $FED^{r_{1165}}$, respectively. Moreover, the $BASE^{r_{1165}}$ represents the LSTM or the baseline GRU model for a given client, while $FED^{r_{1165}}$ curves represent either *LSTM-Fed* or *GRU-Fed*.

1) Baseline vs. Federated Models: When analyzing the plot for DetectorID *19912_NB* in Fig. 4, we observed that the *LSTM-Fed* and *GRU-Fed* models both performed better overall compared to their respective baseline models with $\beta = 72$. The same trend for detector *19912_NB* is observed in Table I, which provides the average inference error computed for each of the metrics, all models, and detectors with $\beta = 72$ during the last 48 FL rounds. From the table, the FL models had a lower prediction error for the majority of RENs, with the FL models having a lower prediction error for 64.29% and 71.43% of the DelDOT sensors, respectively. For the second experimental group, PeMS Bay, a similar trend is in Table II and Fig. 5, with the FL models having lower online MAE and RMSE inference errors for 53.85% and 65.38% of RENs, respectively. The *LSTM-Fed* model for the detector *407150_NB* in Fig. 4a accurately predicted a sudden drop in speed around r_1135, while the baseline model did not, demonstrating the superiority of the online FL approach compared to the per-device baseline model. These results are also summarized in Table III.

In summary, the results of Table III show promise for FL methods in online traffic prediction. Often the online federated models outperformed their associated per-device baseline models. However, the performance of the FL model

TABLE I

ONLINE INFERENCE ERRORS OF THE LAST 48 ROUNDS WITH $\beta = 72$ FOR THE DELDOT DATASET AND 7 CLIENTS

Sensors	DelDOT Dataset							
	LSTM				GRU			
	Baseline		FED		Baseline		FED	
	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
19912_NB	22.44	30.40	**18.51**	**24.73**	23.40	32.47	20.50	27.99
19924_NB	**37.29**	**50.68**	41.23	58.75	38.75	53.77	50.37	69.50
19951_NB	26.76	36.18	**26.09**	**35.46**	30.60	41.38	28.43	40.00
19978_NB	**17.17**	23.59	19.27	**23.31**	19.20	25.66	18.53	24.32
19985_NB	14.24	19.00	18.02	23.20	**13.70**	**18.44**	17.01	22.98
19992_NB	18.06	24.14	**17.35**	**22.69**	18.96	25.44	17.48	23.28
19997_NB	24.43	34.80	**20.38**	**28.09**	26.23	36.15	22.78	31.67

TABLE II

ONLINE INFERENCE ERRORS OF THE LAST 48 ROUNDS WITH $\beta = 72$ FOR THE PEMS BAY DATASET AND 13 CLIENTS

Sensors	PeMS Bay ITSC 13 Subset							
	LSTM				GRU			
	Baseline		FED		Baseline		FED	
	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
407150_NB	1.33	2.29	**1.15**	**1.68**	1.34	2.01	1.19	1.69
407152_NB	**0.88**	**1.14**	1.38	1.70	0.97	1.31	1.36	2.70
407153_NB	1.32	1.81	**1.21**	**1.62**	1.38	1.88	1.25	1.63
407151_NB	1.20	1.69	**1.06**	**1.41**	1.24	1.70	1.10	2.10
407155_NB	**0.97**	1.32	0.98	**1.28**	1.09	1.52	1.03	1.35
407176_NB	**1.18**	1.78	1.21	**1.47**	1.25	1.79	1.22	2.11
407157_NB	1.70	2.25	**1.47**	1.97	1.81	2.34	1.48	**1.96**
407161_NB	1.45	2.02	**1.24**	**1.75**	1.49	2.11	1.30	3.30
407177_NB	**1.12**	3.22	1.22	3.02	1.16	3.58	1.22	**2.80**
407174_NB	**2.43**	**3.15**	3.01	3.75	2.58	3.34	3.01	3.77
407179_NB	**1.20**	**2.18**	1.33	2.66	1.34	2.74	1.37	2.73
407180_NB	1.18	1.72	**1.06**	1.60	1.25	1.82	1.08	**1.58**
407181_NB	**0.71**	**1.09**	0.89	1.28	0.77	1.31	0.94	1.30

TABLE III

SUMMARY OF THE MODEL PERFORMANCE ACROSS ALL EXPERIMENTS FROM THE CLIENT PERSPECTIVE. COUNTS THE NUMBER OF CLIENTS THAT HAD THE LOWEST AVERAGE ONLINE INFERENCE ERROR DURING THE LAST 48 FL ROUNDS FOR ALL MODELS AND BOTH DATASETS

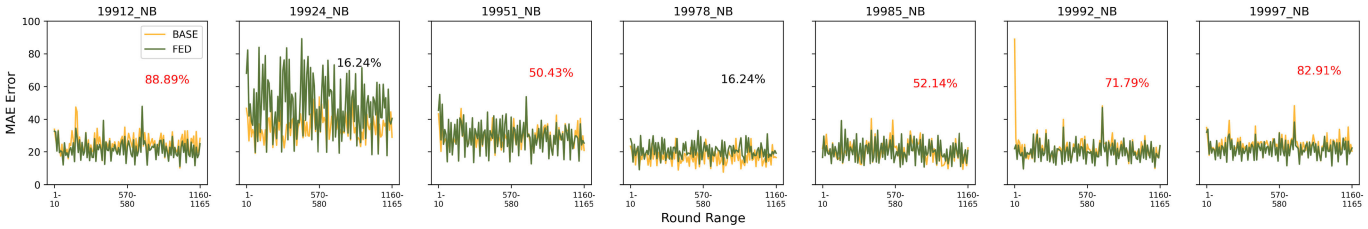
Model	DelDOT (N=7)		PeMS Bay (N=13)	
	MAE	RMSE	MAE	RMSE
GRU-Base	2	2	5	5
GRU-Fed	5	5	8	8
LSTM-Base	3	2	7	4
LSTM-Fed	4	5	6	9
Total (%): Fed	**64.29%**	**71.43%**	**53.85%**	**65.38%**
Total (%): Base	34.71%	28.57%	46.15%	34.62%

exhibited fewer overall improvements, compared to baselines, in the PeMS Bay experiments. Due to the non-stationarity of traffic data and its temporal volatility, the traffic trends of nearby sensors, even along the same roadway and direction of travel, can be noticeably different, introducing client drift at different stages of the online learning process. Traffic trends are dynamic and spatiotemporal, while the impacts of congestion are intense, sudden, and highly localized. The experimental results indicate that a time-dependent and localized aggregation of models, considering the stochastic nature of traffic data, has the potential to minimize the impacts of client drift for online FL in ITS.

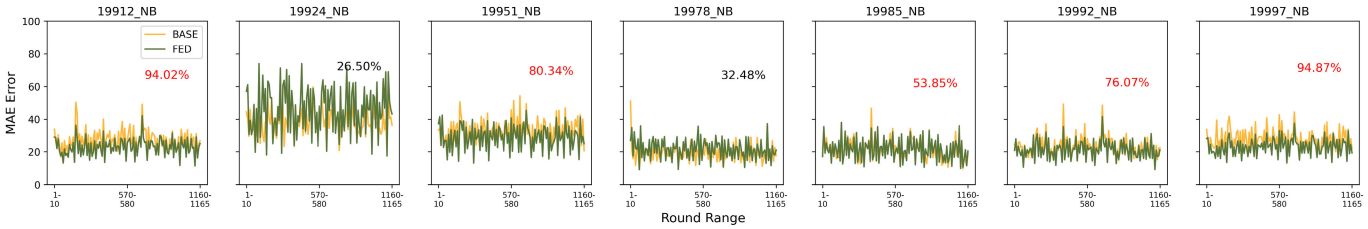
2) Long Short-Term Memory (LSTM) vs. Gated Recurrent Unit (GRU) Models: Tables I and II compare the performance of *LSTM-Fed* and *GRU-Fed* models. During the last 48 FL rounds, the *LSTM-Fed* model prediction error using MAE was 7.39% and 2.61% lower on average for all REN clients compared to *GRU-Fed*, for the DelDOT and PeMS Bay experiments, respectively. Generally, traffic flow data is more stochastic than speed, leading to a more challenging modeling problem. The difference in performance between the federated LSTM and GRU models is prominent for the DelDOT experimental group. Since the GRU cell is less complex than the LSTM cell, the resulting model has significantly fewer learnable parameters [58] and did not capture the traffic trends with the same precision. However, when working with speed data, the trade-off in performance for the GRU model is less severe while offering faster training times due to the reduced complexity. Summarily, the choice of RNN cell will depend on the specific application requirements and traffic variable to be forecasted.

3) Real-Time Mean Absolute Error (MAE) Across All 1165 Rounds: During the experiments, we collected the model inference errors during each FL round to analyze the evolution of the models and their performance as online learning progressed. Fig. 6 plots the normalized MAE errors for all 1,165 FL rounds and sensors, averaged in 10-round intervals, for both the federated models. The round range is on the x-axis, and the y-axis indicates the MAE error. A percentage value on each graph indicates the 10-round intervals in which the MAE inference error of the FL models was lower than the associated baseline models for a given experiment. The percentages in red indicate that the value is greater than 50%, highlighting a case where the federated model, most of the time, outperformed the baseline model. The results indicate that, for GRU and LSTM, the federated models produced the most accurate predictions in most 10-round groups for five of seven clients. Sensors *19912_NB* and *19997_NB* experienced the greatest improvement with their FL models, almost always producing predictions with lower error than their respective baseline models. However, sensor *19924_NB* experienced significant client drift throughout the FL process, degrading inference performance.

4) Feed-Forward (FF) Inference Performance: Fig. 7 displays the FF prediction curves for all seven sensors and the look-ahead times of 15 and 30-minutes, respectively. For the FF experiments, we selected the LSTM model architecture and DelDOT dataset, with $\beta = 72$, and performed the FF inference process alongside the real-time inference process described in Sec. IV-B, using a single global model, G^{r_i}. Moreover, out of all 1,165 training rounds, we graph the FF inferences for the most recent 48 rounds (i.e., the two most recent days) and compare the FF predictions with the ground truth traffic flow



(a) MAE of the LSTM models with $\beta = 72$.



(b) MAE of the GRU models with $\beta = 72$.

Fig. 6. Online MAE inference error for the GRU and LSTM models during all 1165 rounds with $\beta = 72$.

TABLE IV
ONLINE FEED-FORWARD INFERENCE ERRORS AVERAGED OVER THE LAST 48 ROUNDS USING THE *LSTM-Fed* MODEL WITH $\beta = 72$

Sensor	Look-ahead Time			
	15-minutes		30-minutes	
	MAE	RMSE	MAE	RMSE
19912_NB	23.02	30.14	31.83	41.34
19924_NB	59.68	78.92	91.06	117.09
19951_NB	33.59	49.51	52.19	69.54
19978_NB	25.21	30.24	39.77	46.87
19985_NB	25.07	31.13	40.02	50.89
19992_NB	22.96	31.19	31.51	40.65
19997_NB	24.27	32.35	33.21	43.06

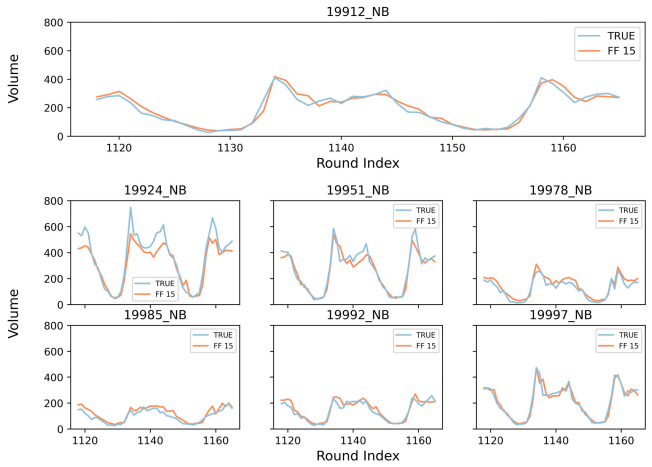
data. In each subfigure of Fig. 7, traffic volume is on the y-axis with the FL round index as the x-axis with detector 19912_NB as the reference detector. The error metrics for FF predictions are in Table IV, which computes the prediction error across the last 120 rounds of online training and inference. We select 120 rounds as the plotting range because the FF approach generates one prediction for each look-ahead time during any round r_i.

Examining Fig. 7, it is observed that the proposed FF inference approach for the federated LSTM model exhibits mixed results across the detector clients in the experiments. While all clients experienced increased prediction error as the look-ahead time increased, the impacts are most severe for detectors having significant differences in traffic trends between peak and off-peak periods, such as detector *19924_NB*, when compared to the single-step inferences. We believe the spikes in error are most likely a result of the stochastic and noisy nature of the traffic data, which exhibits many sharp peaks and fluctuations during the morning and evening rush hours. When feeding forward the prior predictions around the peak periods, the prediction error magnifies and compounds as the prediction horizon increases. A similar trend occurs when comparing the MAE prediction error for the 15-minute look-ahead predictions in Table IV with the 5-minute inferences in Table I. For example, detector *19912_NB* had a 5-minute MAE prediction error of 22.44, while the 15-minute look-ahead prediction error was only 23.02. In contrast, detector *19924_NB* had MAE errors of 37.29 and 59.68 for the 5-minute and 15-minute predictions, respectively. The traffic volume curve for detector *19912_NB* is significantly smoother and less noisy than detector *19924_NB*.

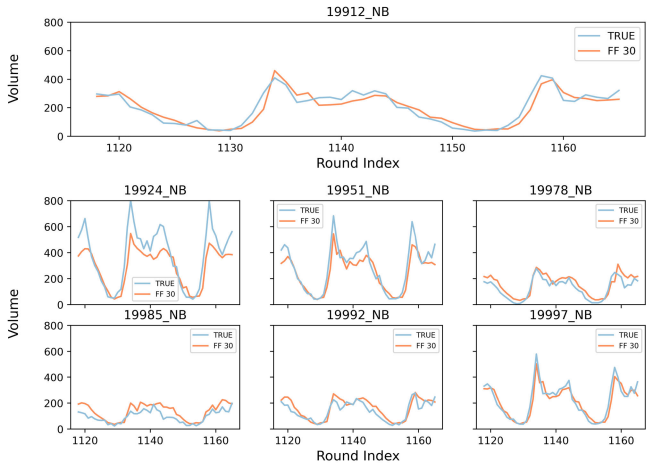
C. Performance of Blockchain Network

Experiments benchmark and analyze the blockchain network performance under transaction latency and throughput metrics. Transaction latency measures the lifetime of a single transaction, from the initial construction on the client side until the updates are committed to the ledger and usable by the clients; including consensus processing and communication between the clients, peers, and orderers. Transaction throughput denotes the number of transactions per second (TPS) processed by the blockchain network during a testing cycle with a fixed input rate. The input rate indicates the clients' number of TPS sent during each experiment. We analyzed changes to both metrics for all experiments under varying send rates and differentiated the READ and WRITE operations, as each operation's associated cost is noticeably different.

1) Transaction Throughput: Figs. 8a and 8b plot the average transaction throughput of our blockchain network for the READ and WRITE operations, respectively, across multiple testing cycles. For the READ operation, Fig. 8a illustrates that the blockchain network can efficiently process READ transactions in real-time as the network load increases to about 1750 TPS; however, as the load exceeds 1750 TPS, the throughput levels stabilize. The results indicate that our

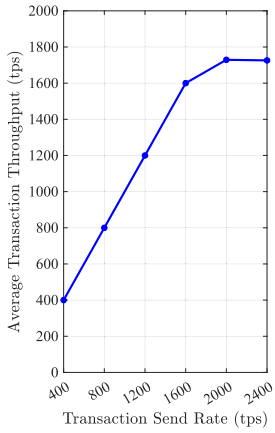


(a) Feed-forward 15-minutes ahead inference with $\beta = 72$.

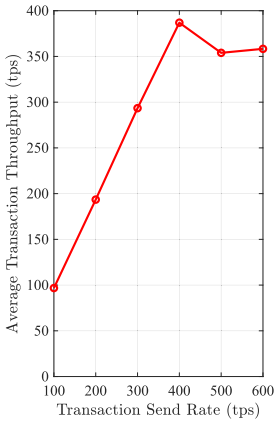


(b) Feed-forward 30-minutes ahead inference with $\beta = 72$.

Fig. 7. Online feed-forward inference curves of DelDOT clients during the last 48 rounds using the *LSTM-Fed* model.



(a) READ operations



(b) WRITE operations

Fig. 8. Average transaction throughputs for READ and WRITE operations.

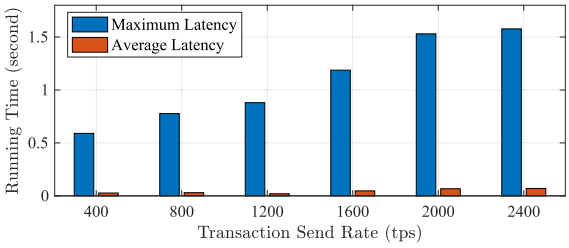


Fig. 9. Maximum and average transaction latencies for READ operations.

Hyperledger Fabric blockchain network is scalable and can support multiple concurrent FL processes with many clients.

Moreover, Fig. 8b highlights that the WRITE operation is more expensive than the READ operation, as the blockchain network performance peaked at a processing speed of 390 TPS. As the send rate increases past 400 TPS, the performance levels degrade slightly and stabilize around 350 TPS. One explanation for this observation is the increased processing time of the WRITE operation compared to the READ operation. When writing to the blockchain, transactions will queue in the pending transaction pool as the send rate exceeds the maximum network throughput, causing a bottleneck. This issue is less visible in the READ results, as pending transactions are processed faster.

2) Transaction Latency: Figs. 9 and 10 report both the average and minimum transaction latency for READ and WRITE operations, respectively. In Fig. 9 there is a clear uptrend in the latency when increasing send rate because higher transaction loads on each peer and orderer will increase confirmation times. However, even at the highest send rate of 2400 TPS, the average latency value is only 0.07 seconds, demonstrating that the blockchain network can perform well for retrieval operations even under considerable transaction loads. On the other hand, Fig. 10 shows the severely degraded performance when the WRITE load exceeds 400 TPS, with the average latency value rapidly converging to the maximum latency, confirming that the blockchain will not perform well in the architecture if the WRITE load exceeds 400 TPS. Comparing both operations, it is clear that even at lower send rates, the WRITE operation's average and maximum latency values are significantly higher than that of the READ operation.

D. System Execution Time Analysis

To demonstrate the feasibility of the architecture for real-world deployment, we perform extensive experiments to analyze the running time of different operations on a resource-constrained edge device. Specifically, we analyze the training and inference time for the DL models and *FedAvg* computation time under various settings and for a single device. Devices perform the operations in parallel during each FL round.

Fig. 11 plots the execution time for training and testing of the *GRU-Fed* and *LSTM-Fed* models for various values of β during a single FL round. Increasing β has a significant impact on the training time, however, it is observed that *GRU-Fed* scales significantly better than *LSTM-Fed* due to its more

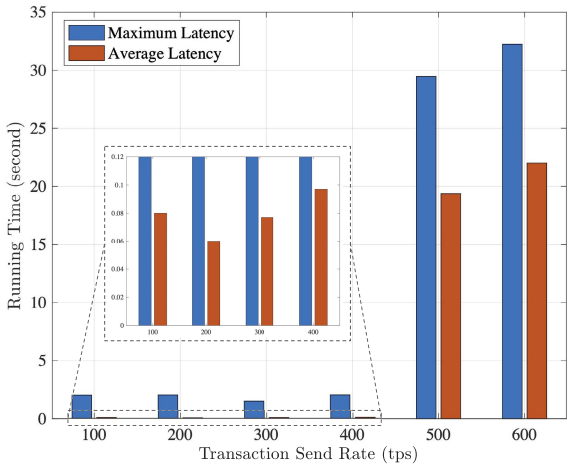


Fig. 10. Maximum and average transaction latencies for WRITE operations.

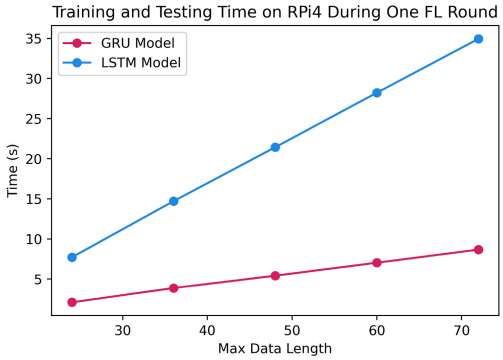


Fig. 11. Training and inference execution time on Raspberry Pi v4 for the GRU and LSTM models with varying β.

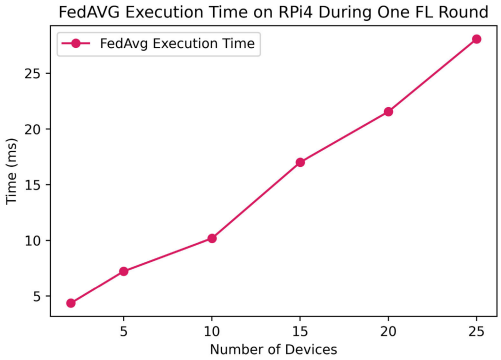


Fig. 12. *FedAvg* execution time from the perspective of a single device for varying client counts.

simple internal structure. Moreover, even at $\beta = 72$, it is observed that training and testing take a combined maximum time of around 35 seconds for the LSTM model, highlighting the practicality of the proposed approach. In Fig. 12, we measure the execution time of *FedAvg* as a function of the number of participating devices, which is independent of the β. Due to the simplicity of the computation, *FedAvg* executes on the order of milliseconds even when the participant count increases to 25.

In addition to the model training, inference, and *FedAvg* computation times, the end-to-end execution time for a single FL round includes: (1) the transaction processing time to deposit the local model on the blockchain (WRITE); (2) the transaction processing time to retrieve the global model update from the blockchain (READ); and (3) data preparation and scaling. We measure and discuss the blockchain READ and WRITE latency in Sec. IV-C, demonstrating that the average transaction latency for the blockchain READ and WRITE operations is on the order of milliseconds when the send rate is under 400 TPS. Notably, in all the experiments, the data preprocessing and scaling time is always less than 2 milliseconds, regardless of β.

Examining the Figures, it is evident that scaling the number of client devices has little impact on the overall execution time because the client count only impacts the execution time of the *FedAvg* operation, which occurs on the order of milliseconds. While the client count can indirectly affect blockchain performance by increasing the transaction load, it is not a scalability concern in the architecture because a client only sends two transactions per FL round. However, the choice of the model architecture and β significantly impact the overall execution time during a single FL round. Summarily, the experimental results highlight the efficiency and scalability of the proposed real-time learning approach for predicting traffic flow in a real-world setting.

V. Conclusion

This paper proposes an FL approach for training online traffic prediction models at the ITS network edge using live traffic data streams, in contrast to a historical database. We incorporated blockchain and CEC to decentralize the FL process. The FL process is prototyped using Python, and we simulated blockchain operations on Hyperledger Fabric to conduct extensive experiments in various settings. We federated the LSTM and GRU models in the FL experiments and measured the performance of online training and inference using location-specific shards of arterial traffic data for each participant. Furthermore, we quantify the execution time of the FL process using the Raspberry Pi as a lightweight edge device. The experimental results demonstrate that the proposed online FL approach outperforms the baseline cases without federation for the majority of sensors and has suitable execution time and latency for real-world deployment. For future work, we plan to research online, personalized, and time-dependent model aggregation methods for improving the performance of online FL in ITS. This work will help lay the foundation for future research into online and decentralized traffic prediction models.

Acknowledgment

The authors appreciate the support from Gene Donaldson, DelDOT TMCs Operations Manager.

References

[1] M. Akhtar and S. Moridpour, "A review of traffic congestion prediction using artificial intelligence," *J. Adv. Transp.*, vol. 2021, pp. 1–18, Jan. 2021.

[2] D. Schrank, T. Lomax, and B. Eisele. (2019). *2019 Urban Mobility Report*. Texas Transportation Institute [Online]. Available: https://static.tti.tamu.edu/tti.tamu.edu/documents/mobility-report-2019.pdf

[3] UD Energy. *Annual Vehicle Miles Traveled in the United States*. Accessed: Nov. 10, 2022. [Online]. Available: https://afdc.energy.gov/data/10315

[4] R. Jiang et al., "Spatio-temporal meta-graph learning for traffic forecasting," in *Proc. AAAI Conf. Artif. Intell.*, vol. 37, no. 7, 2023, pp. 8078–8086.

[5] Z. Shao, Z. Zhang, F. Wang, and Y. Xu, "Pre-training enhanced spatial–temporal graph neural network for multivariate time series forecasting," in *Proc. 28th ACM SIGKDD Conf. Knowl. Discovery Data Mining*, Aug. 2022, pp. 1567–1577.

[6] Z. Shao et al., "Decoupled dynamic spatial–temporal graph neural network for traffic forecasting," *Proc. VLDB Endowment*, vol. 15, no. 11, pp. 2733–2746, Jul. 2022, doi: 10.14778/3551793.3551827.

[7] W. Zhang, Z. Zhang, and H.-C. Chao, "Cooperative fog computing for dealing with big data in the Internet of Vehicles: Architecture and hierarchical resource management," *IEEE Commun. Mag.*, vol. 55, no. 12, pp. 60–67, Dec. 2017.

[8] M. A. Azad, S. Bag, S. Parkinson, and F. Hao, "TrustVote: privacy-preserving node ranking in vehicular networks," *IEEE Internet Things J.*, vol. 6, no. 4, pp. 5878–5891, Aug. 2019.

[9] K. Bonawitz et al., "Towards federated learning at scale: System design," in *Proc. Mach. Learn. Syst.*, vol. 1, 2019, pp. 374–388.

[10] 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.

[11] F. Ayaz, Z. Sheng, D. Tian, and Y. L. Guan, "A blockchain based federated learning for message dissemination in vehicular networks," *IEEE Trans. Veh. Technol.*, vol. 71, no. 2, pp. 1927–1940, Feb. 2022.

[12] M. Aloqaily, I. A. Ridhawi, and M. Guizani, "Energy-aware blockchain and federated learning-supported vehicular networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 11, pp. 22641–22652, Nov. 2022.

[13] W. Li, C. Meese, H. Guo, and M. Nejad, "Blockchain-enabled identity verification for safe ridesharing leveraging zero-knowledge proof," in *Proc. 3rd Int. Conf. Hot Inf.-Centric Netw. (HotICN)*, Dec. 2020, pp. 18–24.

[14] W. Li, C. Meese, Z. G. Zhong, H. Guo, and M. Nejad, "Location-aware verification for autonomous truck platooning based on blockchain and zero-knowledge proof," in *Proc. IEEE Int. Conf. Blockchain Cryptocurrency (ICBC)*, May 2021, pp. 1–5.

[15] H. Chai, S. Leng, Y. Chen, and K. Zhang, "A hierarchical blockchain-enabled federated learning algorithm for knowledge sharing in Internet of Vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 7, pp. 3975–3986, Jul. 2021.

[16] G. Rathee, F. Ahmad, F. Kurugollu, M. A. Azad, R. Iqbal, and M. Imran, "CRT-BIoV: A cognitive radio technique for blockchain-enabled Internet of Vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 7, pp. 4005–4015, Jul. 2021.

[17] H. Liu et al., "Blockchain and federated learning for collaborative intrusion detection in vehicular edge computing," *IEEE Trans. Veh. Technol.*, vol. 70, no. 6, pp. 6073–6084, Jun. 2021.

[18] Y. Lu, X. Huang, K. Zhang, S. Maharjan, and Y. Zhang, "Blockchain empowered asynchronous federated learning for secure data sharing in Internet of Vehicles," *IEEE Trans. Veh. Technol.*, vol. 69, no. 4, pp. 4298–4311, Apr. 2020.

[19] *Blockchain*. Accessed: Dec. 2, 2021. [Online]. Available: https://en.wikipedia.org/wiki/Blockchain

[20] W. Li, C. Meese, M. Nejad, and H. Guo, "P-CFT: A privacy-preserving and crash fault tolerant consensus algorithm for permissioned blockchains," in *Proc. 4th Int. Conf. Hot Inf.-Centric Netw. (HotICN)*, Nov. 2021, pp. 26–31.

[21] J. Kang, Z. Xiong, D. Niyato, Y. Zou, Y. Zhang, and M. Guizani, "Reliable federated learning for mobile networks," *IEEE Wireless Commun.*, vol. 27, no. 2, pp. 72–80, Apr. 2020.

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

[23] Y. Qi, M. S. Hossain, J. Nie, and X. Li, "Privacy-preserving blockchain-based federated learning for traffic flow prediction," *Future Gener. Comput. Syst.*, vol. 117, pp. 328–337, Apr. 2021.

[24] *Hyperledger Caliper*. Accessed: Nov. 15, 2020. [Online]. Available: https://www.hyperledger.org/use/caliper

[25] P. Kairouz et al., "Advances and open problems in federated learning," *Found. Trends Mach. Learn.*, vol. 14, nos. 1–2, pp. 1–210, Jun. 2021.

[26] R. S. Antunes, C. A. D. Costa, A. Küderle, I. A. Yari, and B. Eskofier, "Federated learning for healthcare: Systematic review and architecture proposal," *ACM Trans. Intell. Syst. Technol.*, vol. 13, no. 4, pp. 1–23, Aug. 2022.

[27] S. Jamil, M. Rahman, and Fawad, "A comprehensive survey of digital twins and federated learning for industrial Internet of Things (IIoT), Internet of Vehicles (IoV) and Internet of Drones (IoD)," *Appl. Syst. Innov.*, vol. 5, no. 3, p. 56, Jun. 2022.

[28] F. Sun, Z. Zhang, S. Zeadally, G. Han, and S. Tong, "Edge computing-enabled Internet of Vehicles: Towards federated learning empowered scheduling," *IEEE Trans. Veh. Technol.*, vol. 71, no. 9, pp. 10088–10103, Sep. 2022.

[29] R. Kanagavelu et al., "Two-phase multi-party computation enabled privacy-preserving federated learning," in *Proc. 20th IEEE/ACM Int. Symp. Cluster, Cloud Internet Comput. (CCGRID)*, 2020, pp. 410–419.

[30] J.-H. Chen, M.-R. Chen, G.-Q. Zeng, and J.-S. Weng, "BDFL: A Byzantine-fault-tolerance decentralized federated learning method for autonomous vehicle," *IEEE Trans. Veh. Technol.*, vol. 70, no. 9, pp. 8639–8652, Sep. 2021.

[31] J. Zhang et al., "Adaptive vertical federated learning on unbalanced features," *IEEE Trans. Parallel Distrib. Syst.*, vol. 33, no. 12, pp. 4006–4018, Dec. 2022.

[32] X. Kong, H. Gao, G. Shen, G. Duan, and S. K. Das, "FedVCP: A federated-learning-based cooperative positioning scheme for social Internet of Vehicles," *IEEE Trans. Computat. Social Syst.*, vol. 9, no. 1, pp. 197–206, Feb. 2022.

[33] 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. 2019.

[34] P. Ramanan and K. Nakayama, "BAFFLE: Blockchain based Aggregator free federated learning," in *Proc. IEEE Int. Conf. Blockchain (Blockchain)*, 2020, pp. 72–81.

[35] C. Ma et al., "When federated learning meets blockchain: A new distributed learning paradigm," *IEEE Comput. Intell. Mag.*, vol. 17, no. 3, pp. 26–33, Aug. 2022.

[36] M. Shaygan, C. Meese, W. Li, X. Zhao, and M. Nejad, "Traffic prediction using artificial intelligence: Review of recent advances and emerging opportunities," *Transp. Res. C, Emerg. Technol.*, vol. 145, Dec. 2022, Art. no. 103921.

[37] A. M. Elbir, B. Soner, S. Coleri, D. Gunduz, and M. Bennis, "Federated learning in vehicular networks," in *Proc. IEEE Int. Medit. Conf. Commun. Netw. (MeditCom)*, Sep. 2022, pp. 72–77.

[38] W. Y. B. Lim et al., "Towards federated learning in UAV-enabled Internet of Vehicles: A multi-dimensional contract-matching approach," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 8, pp. 5140–5154, Aug. 2021.

[39] Q. Zhang, P. Palacharla, M. Sekiya, J. Suga, and T. Katagiri, "Blockchain-based secure aggregation for federated learning with a traffic prediction use case," in *Proc. IEEE 7th Int. Conf. Netw. Softwarization (NetSoft)*, 2021, pp. 372–374.

[40] T. Zeng et al., "Multi-task federated learning for traffic prediction and its application to route planning," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jul. 2021, pp. 451–457.

[41] Y. Chen, Y. Ning, M. Slawski, and H. Rangwala, "Asynchronous online federated learning for edge devices with non-IID data," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, 2020, pp. 15–24.

[42] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," in *Proc. Mach. Learn. Syst.*, vol. 2, 2020, pp. 429–450.

[43] C. Xie, S. Koyejo, and I. Gupta, "Asynchronous federated optimization," 2019, *arXiv:1903.03934*.

[44] *Public, Private, Permissioned Blockchains Compared*. Accessed: Dec. 2, 2021. [Online]. Available: https://www.investopedia.com/news/public-private-permissioned-blockchains-compared/

[45] P. Swathi, C. Modi, and D. Patel, "Preventing Sybil attack in blockchain using distributed behavior monitoring of miners," in *Proc. 10th Int. Conf. Comput., Commun. Netw. Technol. (ICCCNT)*, 2019, pp. 1–6.

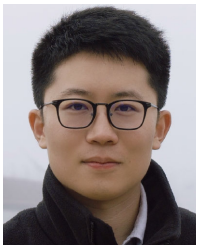
[46] C. Lin, D. He, X. Huang, K.-K. R. Choo, and A. V. Vasilakos, "BSeIn: A blockchain-based secure mutual authentication with fine-grained access control system for industry 4.0," *J. Netw. Comput. Appl.*, vol. 116, pp. 42–52, Aug. 2018.

[47] D. Said, "A decentralized electricity trading framework (DETF) for connected EVs: A blockchain and machine learning for profit margin optimization," *IEEE Trans. Ind. Informat.*, vol. 17, no. 10, pp. 6594–6602, Oct. 2021.

[48] Y. Gao, Y. Chen, X. Hu, H. Lin, Y. Liu, and L. Nie, "Blockchain based IIoT data sharing framework for SDN-enabled pervasive edge computing," *IEEE Trans. Ind. Informat.*, vol. 17, no. 7, pp. 5041–5049, Jul. 2021.
[49] H. Guo, W. Li, M. Nejad, and C.-C. Shen, "Access control for electronic health records with hybrid blockchain-edge architecture," in *Proc. IEEE Int. Conf. Blockchain (Blockchain)*, 2019, pp. 44–51, doi: 10.1109/Blockchain.2019.00015.
[50] J. Cui, F. Ouyang, Z. Ying, L. Wei, and H. Zhong, "Secure and efficient data sharing among vehicles based on consortium blockchain," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 7, pp. 8857–8867, Jul. 2022.
[51] W. Li, M. Nejad, and R. Zhang, "A blockchain-based architecture for traffic signal control systems," in *Proc. IEEE Int. Congr. Internet Things (ICIOT)*, Jul. 2019, pp. 33–40.
[52] X. Wang, S. Garg, H. Lin, G. Kaddoum, J. Hu, and M. M. Hassan, "Heterogeneous blockchain and AI-driven hierarchical trust evaluation for 5G-enabled intelligent transportation systems," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 2, pp. 2074–2083, Feb. 2023.
[53] A. Vangala, B. Bera, S. Saha, A. K. Das, N. Kumar, and Y. Park, "Blockchain-enabled certificate-based authentication for vehicle accident detection and notification in intelligent transportation systems," *IEEE Sensors J.*, vol. 21, no. 14, pp. 15824–15838, Jul. 2021.
[54] *Hyperledger Fabric*. Accessed: Oct. 21, 2020. [Online]. Available: https://www.hyperledger.org/projects/fabric
[55] D. Ongaro and J. Ousterhout, "The raft consensus algorithm," in *Proc. USENIX Annu. Tech. Conf. (USENIX ATC)*. Philadelphia, PA, USA: USENIX Association, Jun. 2014, pp. 305–319.
[56] C. Meese, H. Chen, S. A. Asif, W. Li, C.-C. Shen, and M. Nejad, "BFRT: Blockchained federated learning for real-time traffic flow prediction," in *Proc. 22nd IEEE Int. Symp. Cluster, Cloud Internet Comput. (CCGrid)*, May 2022, pp. 317–326.
[57] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," 2017, *arXiv:1707.01926*.
[58] R. Cahuantzi, X. Chen, and S. Güttel, "A comparison of LSTM and GRU networks for learning symbolic sequences," 2021, *arXiv:2107.02248*.

Collin Meese (Graduate Student Member, IEEE) received the B.S. degree in computer science from the University of Delaware in 2020, where he is currently pursuing the Ph.D. degree. His research interests include blockchain, distributed machine learning, intelligent transportation systems, connected and autonomous vehicles, and intelligent civil systems. He is an NSF GRFP Fellow.

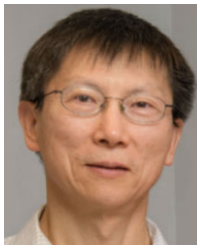
Hang Chen received the B.S. degree in computer science from the University of Delaware, USA, in 2015, where he is currently pursuing the Ph.D. degree. His research interests include distributed ledger technology, federated learning consensus design, deep networks cross-validation, and smart transportation.



Wanxin Li (Member, IEEE) received the bachelor's degree from Chongqing University and the master's and Ph.D. degrees from the University of Delaware. He is currently an Assistant Professor with the Department of Communications and Networking, Xi'an Jiaotong-Liverpool University. His research interests include blockchain, cryptography, distributed AI, and smart transportation. He is a member of ACM and IET. He was a recipient of the IEEE ITSS Best Dissertation Award and the IEEE TEMS Outstanding Ph.D. Dissertation Award in 2022.

Danielle Lee received their B.S. degree in environmental engineering and the B.S. degree in civil engineering in 2021 and the M.S. degree from the University of Delaware in 2022, where she is currently pursuing the Ph.D. degree. Her research interests include multi-modal intelligent transportation systems and equitable intelligent infrastructure systems. She is a member of the ASEE.

Hao Guo (Member, IEEE) received the B.S. degree in computer science from Northwest University, Xi'an, China, in 2012, the M.S. degree in computer science from Illinois Institute of Technology, Chicago, USA, in 2014, and the Ph.D. degree in computer science from the University of Delaware, Delaware, NE, USA, in 2020. He is currently an Assistant Professor with the School of Software, Northwestern Polytechnical University. His research interests include blockchain and distributed ledger technology, privacy-preserving computing, applied cryptography, and the Internet of Things (IoT). He is a member of ACM and CCF.



Chien-Chung Shen (Member, IEEE) received the B.S. and M.S. degrees in computer science from National Chiao Tung University, Taiwan, and the Ph.D. degree in computer science from UCLA. He was a Research Scientist with Bellcore Applied Research, working on the control and management of broadband networks. He is currently a Professor with the Department of Computer and Information Sciences, University of Delaware. His research interests include blockchain, Wi-Fi, SDN and NFV, ad hoc and sensor networks, dynamic spectrum management, cybersecurity, distributed computing, and simulation. He is a member of the ACM. He was a recipient of the NSF CAREER Award.

Mark Nejad (Senior Member, IEEE) is currently an Associate Professor with the University of Delaware. His research is funded by the National Science Foundation and the Department of Transportation. He has published more than 50 peer-reviewed articles. His research interests include network optimization, distributed systems, blockchain, game theory, and intelligent transportation systems. He is a member of INFORMS. He received several publication awards, including the 2016 IISE Pritsker Best Doctoral Dissertation Award and the 2019 CAVS Best Paper Award from the IEEE VTS.