Dynamic Semi-Synchronous Federated Learning for Connected Autonomous Vehicles

Wellington Lobato[1], Joahannes B. D. da Costa[1], Allan M. de Souza[1], Denis Rosário[2], Christoph Sommer[3], Leandro Villas[1]

[1]Institute of Computing, University of Campinas (UNICAMP), Brazil

[2]Federal University of Pará (UFPA), Brazil

[3]TU Dresden, Faculty of Computer Science, Germany

{wellington,joahannes}@lrc.ic.unicamp.br,

denis@ufpa.br, cms-labs.org/people/sommer,leandro@ic.unicamp.br

Abstract. *Due to the increased computational capacity of Connected and Autonomous Vehicles (CAVs) and concerns about transferring private information, storing data locally and moving network computing to the edge is becoming increasingly appealing. This makes Federated Learning (FL) appealing for CAV applications. However, the synchronous protocols used in FL have several limitations, such as low round efficiency. In this context, this work presents FALCON, a semi-synchronous protocol for FL based on the link duration. FALCON leverages data periodically transmitted by CAVs to compute link duration and establish a dynamic temporal synchronization point. Additionally, FALCON includes a client selection mechanism that considers the local model versions and models with higher local loss. FALCON reduces the communication rounds and the number of selected clients while maintaining the same level of accuracy for FL applications.*

1. Introduction

Autonomous Vehicles (AVs) free the driver from the stressful task of driving, providing physical and mental relief [Sun et al. 2022b]. The operation of AVs considers a set of sensors responsible for furnishing the vehicle with knowledge about the surrounding environment, often providing data to Machine Learning (ML) models in order to understand/learn the environment for better driving decisions [Liu and Gaudiot 2022]. In this context, CAVs extend the capabilities of AVs by enabling them to share the data with neighbors to add further aspects to data processing, enabling a cooperative understanding of the environment [Damaj et al. 2021].

CAVs generate a vast amount of sensor data – up to 1 GB per second, and such extensive data sharing in CAVs raises significant communication and privacy concerns [Chellapandi et al. 2024]. For instance, the data collected by CAVs commonly reveals sensitive information about their users, including their home addresses, frequently visited locations, and daily routines [Lv et al. 2023]. In this context, Federated Learning (FL) emerges as a crucial solution to provide a privacy-preserving property for CAV applications while reducing the strain on the network [Chellapandi et al. 2024]. In addition, it is believed that the future of ML and cloud computing will be distributed at the

network edge, such as allowed by FL [Zhang et al. 2023]. Specifically, at each communication round, a set of CAVs is selected to receive the global model, perform the training based on its local data, and share their model parameters instead of their sensing data [Song et al. 2022]. Afterward, a given aggregation policy takes an average of the shared local models at edge servers to produce an accurate global model. Finally, the updated global model is distributed to the CAVs. Hence, FL allows continuous learning by adapting the ML model of CAVs without sharing raw dat a[Wang et al. 2023].

Many FL implementations rely on synchronous protocols for model aggregation, where the server waits for all selected CAVs to return the local training results before proceeding with the aggregation policy [You et al. 2023]. However, synchronous FL protocols have limitations, including unreliable participants, low round efficiency, and underutilization of CAV computation resources [Hao et al. 2020]. Furthermore, the synchronous FL protocols require the server to wait for slow learners (stragglers) in each communication round, reducing the efficiency of FL [Wu et al. 2021]. Otherwise, semi-synchronous FL protocols introduce the synchronization point concept for model aggregation. Specifically, at each communication round, the edge server waits for a given time interval (*i.e.*, temporal synchronization point) for CAVs to upload their local model before applying the aggregation policy. As a result, the edge server no longer needs to wait for stragglers or disconnections [Wu et al. 2021].

In this context, a suitable temporal synchronization point is crucial to improve the efficiency in terms of number of communication rounds, number of selected clients, and the aggregation cost of the semi-synchronous FL protocol [Chen et al. 2020]. Establishing a reliable synchronization point is challenging since there is a trade-off between the temporal synchronization point and the FL efficiency [Ma et al. 2021]. For example, having a large temporal synchronization point results in less communication rounds, reducing computation and communication costs, as each round entails transmitting the model across the network and applying the aggregation policy [Almanifi et al. 2023].

However, this also results in the server waiting for extended periods without receiving local updates [Sun et al. 2022a]. Furthermore, the number of selected clients also influences the communication cost, since more clients lead to more model transmission across the network in each communication round. Hence, defining the dynamic synchronization point based on a given criteria at each round is crucial, which is still an open issue. The dynamic synchronization point also enables to adapt to the dynamic nature of CAVs and the wireless communication channel characteristics, since CAVs frequently lose contact with the cloud or edge server and fail to complete training or upload their local model [Niknam et al. 2020].

In this paper, we propose a dynamic semi-synchronous FL protocol based on connectivity information, which reduces number of communication rounds and aggregation cost while maintaining the accuracy of global model, called **F**eder**A**ted **L**earning based on **CO**n**N**ectivity (FALCON). In its operation, the edge server computes the average link duration of the selected clients (*i.e.*, a set of CAVs) and defines a dynamic temporal synchronization point to aggregate and update the global model. FALCON also includes a client selection mechanism to select the models with higher local loss. Evaluation results show that FALCON reduces the total number of communication rounds by 46.67%, in the high density scenario, and the number of selected clients by 38.46% while maintaining the accuracy of the global model by 83.11%, compared to other approaches.

Our contributions can be summarized as follows: *(i)* We propose a protocol for semi-synchronous FL called FALCON, which dynamically adjusts the temporal synchronization point based on the link duration between CAVs and the edge server. *(ii)* We design a client selection scheme by considering CAVs with higher local loss and model version. *(iii)* In a detailed performance evaluation with a realistic mobility trace, we show the benefits of link duration information to define the temporal synchronization point with better reliability compared to other state-of-the-art approaches.

The remainder of this paper is structured as follows. Section 2 introduces related work, highlighting the individual advantages and limitations of earlier approaches. Section 3 presents the FALCON operation as well as the methods used to compute the synchronization point and client selection. Section 4 shows the performance analysis of the proposed protocol compared to state-of-the-art approaches in the CAV scenario. Finally, section 5 describes the conclusion of this paper and present some future work directions.

2. Related Work

[Nishio and Yonetani 2019] proposed FedCS, an FL protocol with a client selection mechanism wherein the server seeks to maximize client inclusion within a fixed time interval. FedCS employs a client selection mechanism based on the information Mobile Edge Computing (MEC) provides. However, FedCS did not consider the impact of the mobility of devices, which can significantly interfere with the performance of FL. Additionally, FedCS is designed with optimistic assumptions in mind: it does not consider clients' unreliability and mainly focuses on leveraging the wireless networks when there is a stable connection and no congestion.

[Stripelis et al. 2022] introduced a semi-synchronous FL protocol (SEMI-SYNC) that aggregates local models in a fixed interval, reducing idle time and achieving fast convergence. The synchronization point is determined based on the maximum time required for any client to complete a single epoch. However, the communication model employed in this work fails to account for network disconnections. Furthermore, the authors neglect to consider the costs associated with model transmission.

[Wu et al. 2021] presented the SAFA protocol for fast and lag-tolerant federated optimization. The authors implement a server cache-based mechanism to mitigate the effects of stragglers, crashes, and local model obsolescence. The SAFA aggregation policy relies on specific criteria, including attaching to fixed time intervals and cache sizes for storing received models. However, the accuracy of FL applications can be influenced by the number of communication rounds and can lead to an increase in selected clients, primarily driven by the cache-based synchronization point. Moreover, the authors did not consider the mobility of devices.

[Liang et al. 2022] proposed a semi-synchronous FL protocol to enhance the performance of ML for CAVs. The authors implemented a dynamic waiting time technique to adjust the server temporal synchronization point at each communication round based on the local computation time. However, the authors assume that the communication of CAVs occurs in an open space, which may not accurately reflect real-world scenarios. Furthermore, the authors do not directly consider the impact of CAVs mobility on the choice of waiting time.

By analyzing the state of the art, we conclude that existing approaches do not

consider the mobility of CAVs, intermittent communication, nor network dynamics to define a dynamic temporal synchronization point for model aggregation, which could jeopardize the efficiency of FL-based applications. Some approaches also employ fixed intervals for synchronization or a cache for model updates, which fails to ensure efficiency in scenarios characterized by high network disconnections and dynamic mobility patterns.

3. Dynamic Semi-synchronous Federated Learning

This section introduces the FALCON protocol, which considers the link duration between the CAVs and edge servers to define the dynamic temporal synchronization point. In its operation, FALCON defines the temporal synchronization point to aggregate and update the global model with better reliability. FALCON also includes a client selection mechanism based on model version and local loss. In the following, we introduce the system model and FALCON operations.

3.1. Network and System Model

We employ a scenario involving a set of n CAVs (referred to as clients or learners in FL) navigating an urban area. Each CAV is uniquely identified by an index i within the range of $[1, n]$, represented as $V = v_1, \cdots, v_n$. Every CAV, denoted as v_i, is traveling in a specific direction indicated by $\overrightarrow{dir_i}$ and has a speed, s_i, that falls within the range of a minimum speed (s_{min}) and a maximum speed (s_{max}) limit. Each vehicle v_i has a position $P_i = (X_i, Y_i)$ at a given time instant t assigned by the Global Navigation Satellite System (GNSS). Furthermore, each CAV v_i is equipped with a Vehicle-to-Infrastructure (V2I) communication interface, such as Dedicated Short Range Communication (DSRC) or 5G. Additionally, the CAVs are equipped with onboard sensors capable of collecting data, which is subsequently utilized to feed ML models for tasks such as recognition or image classification within the context of FL-based applications.

The scenario comprises a collection of m edge servers, strategically deployed at arbitrary locations, where each edge server is uniquely identified by an index j within the range of $[1, m]$ and is denoted as $ES = ES_1, \cdots, ES_m$. In this context, the edge server ES_j plays a crucial role in the distribution of ML parameters for the initial or updated global model ω to all CAVs during each communication round μ. Furthermore, the edge server also assumes the responsibility of collecting and analyzing connectivity data, aiding in the determination of a dynamic temporal synchronization point for model aggregation, and applying a client selection mechanism.

3.2. FALCON Operations

The operation of FALCON protocol consists of five steps performed by server or client side during each communication round μ. The initial step (Step 1) occurs when the global model $\omega^{\mu-1}$ is disseminated to the CAVs by the edge server ES_j. Concurrently, in the Step 2, CAVs periodically broadcast beacon messages through the network, and thus the edge server ES_j could collect such beacons to underhand crucial CAV information, such as, position P_i, speed s_i, local loss $f_i(\omega_i^\mu; x_i^k, y_i^k)$, and model version ($vr(\omega_i^\mu)$). Based on such information, the edge server ES_j computes the link duration to determine a dynamic temporal synchronization point, guaranteeing that CAVs upload their local model parameters within the specified link duration. Hence, the edge server ES_j sets a synchronization point T_{lim}^μ at the start of each communication round μ, and waits for CAVs to upload their local model parameters ω_i^μ before applying the aggregation policy (step 4).

In Step 3, each CAV undertakes the local model training using its respective local data. In this way, each CAV v_i has a local dataset $D_i = \{x_i^k, y_i^k\}$, where x_i^k and y_i^k denotes the input data and output label of the k-th data sample, respectively, and $|D_i|$ is the number of data samples collected by a given vehicle v_i. The total number of data samples is expressed by $D \triangleq \sum_{i=1}^{|V|} |D_i|$. Hence, each CAV v_i trains the local models ω_i^μ based on the local data D_i with the local loss function $f_i(\omega_i^\mu; x_i^k, y_i^k)$ of the k-th data sample. The local loss function of CAV v_i can be computed based on Eq. 1, where the local loss must be computed and minimized for better convergence with a minimum accuracy value across users.

$$F_i(\omega_i^\mu) \triangleq \frac{1}{D_i} \sum_{k=1}^{D_i} f_i(\omega_i^\mu; x_i^k, y_i^k). \tag{1}$$

Once the CAV v_i completes the local training, it uploads the resulting local model to the edge server ES_j. It is important to mention that the edge server ES_j waits a temporal synchronization point T_{lim}^μ to receive the uploaded models and apply the aggregation policy to produce a new global model $\omega^{\mu+1}$, as denoted by Eq. 2.

$$\omega^{\mu+1} = \min_{\omega^\mu} F(\omega^\mu) \triangleq \sum_{i=1}^{|V|} \frac{D_i}{D} F_i(\omega_i^\mu) \tag{2}$$

In addition, during the step 4, FALCON defines the number of K selected clients based on a client selection mechanism, where the FALCON protocol sorts the models based on the higher loss value and selects K clients to participate in the upcoming learning rounds. Finally, the edge server generates the updated global model by aggregating the selected local models (Step 3 and 4) and transmits the updated global model (Step 5). In the following sections, we introduce how FALCON performs the computation of dynamic temporal synchronization point and the client selection mechanism.

3.3. Dynamic Temporal Synchronization Point

In semi-synchronous FL protocols, the edge server ES_j sets a synchronization point T_{lim}^μ at the start of each communication round μ and waits for CAVs to upload their local model parameters ω_i^μ before applying the aggregation policy. However, the dynamic mobility patterns of CAVs and the inherent variability of wireless communication channels often result in frequent disruptions in connectivity between CAVs and the edge server, leading to failures in completing training or uploading their local models within the designated link duration. As a result, establishing an appropriate temporal synchronization point based on connectivity information is critical for enhancing the efficiency of semi-synchronous FL protocols, since it allows fine-tuning the synchronization frequency, aggregation cost, and maintaining control over the number of communication round.

FALCON protocol effectively addresses such issues by integrating the link duration value to establish a temporal synchronization point T_{lim}^μ, which guarantee that CAVs upload their local model parameters within a valid link duration. Based on this, we assume that edge server is aware of CAV information in real-time to build the knowledge necessary to determine the temporal synchronization point, which is possible by edge

server receiving the beacons already exchanged by CAVs without extra overhead. Specifically, each CAV transmits periodic beacons containing its index i, position P_i, and speed s_i information.

Given the history of associations of a given CAV v_i with the edge server ES_j, it is possible to decompose the link duration information into time series representing the connection status between a given CAV v_i in the communication range of the edge server ES_j, which is stored in the edge server ES_j. In this way, FALCON calculates the time difference between consecutive beacons by analyzing the timestamp of each received beacon, where Δt_i denotes the link duration of a given CAV v_i maintains contact with the edge server ES_j and it is modeled as follows.

$$\Delta t_i = \frac{R_i - \mathrm{dist}(v_i, ES_j)}{\mathrm{abs}(s_i)}, \tag{3}$$

where R_i denotes the estimated maximum communication range of CAV v_i, $\mathrm{dist}(v_i, ES_j)$ denotes the Euclidean distance between CAV v_i and the edge server ES_j, and s_i represent the speed of a CAV v_i. Hence, the analytical model on link duration Δt_i incorporates the influence of mobility characteristics and channel range, offering a mathematical representation of the intermittent CAV network.

However, Eq. 3 only works for situations in which the CAV speed (s_i) is different from 0. Thus, the FALCON protocol filters speed data in instances where stationary or parked CAVs may influence the link duration. Specifically, when computing the average link duration, the protocol distinguishes parked CAVs, establishing an assumption of an initial synchronization time T^0_{lim}, which subsequently results in a predictable fixed behavior. Hence, if the collection of connection or speed data is not precise, it would not adversely affect the efficiency of the protocol. Our speed filter can be formulated as follows:

$$\Delta t_i = \begin{cases} \Delta t_i & \text{if } s_i > 0 \text{ and } \frac{R_i - \mathrm{dist}(v_i, ES_j)}{\mathrm{abs}(s_i)} > T^0_{lim}, \\ T^0_{lim} & \text{otherwise.} \end{cases} \tag{4}$$

Finally, at the begin of the dynamic temporal synchronization point, FALCON computes the average of link duration of all CAVs Δt stored by the edge server, as shown in Eq. 5. Precisely, FALCON sets a dynamic temporal synchronization point T^{μ}_{lim} at the start of each communication round μ by taking into account the average link duration from the previous temporal synchronization point.

$$T^{\mu}_{lim} = \frac{\sum_{i=1}^{|V|} \Delta t_i}{|V|}. \tag{5}$$

3.4. Client Selection

The client selection mechanism is responsible for determining specific subsets of CAVs eligible to participate in the upcoming learning rounds. In this way, the edge server ES_j distributes the global model parameters ω^{μ}_i to the selected CAVs, which train the ML model over their local data D_i. However, the mechanism must guarantee that the participating CAVs have valuable samples, which reduces the waste of computation resources by removing the learning whose data are no longer critical for the model training.

In this context, the efficiency of the client selection mechanism is closely associated to the fraction of picked clients. In this sense, FALCON incorporates a hyper-parameter C to regulate the maximum fraction ($0 < C < 1$) of CAVs suitable to participate in a single round of training. For instance, we can set C to a large value (*i.e.*, closer to 1) in order to pick as many clients as possible in each round. However, this is neither realistic nor beneficial, since allowing more CAVs to participate increases the potential risk of uplink congestion and the communication cost as well. In addition, the server may have to wait for more CAVs, which may never respond due to crash midway.

In order to select the best models, based on the values obtained after training, FALCON uses beacon transmission to collect the local loss values from each CAV v_i and model version $vr(\omega_i^\mu)$ without additional communication overhead. In this context, we integrate a lag tolerance mechanism to guarantee performance stability in the models of CAVs and prevent outdated CAVs with stale models from participating in the aggregation policy. Specifically, FALCON verifies the version of the local model $vr(\omega_i^\mu)$ received by the edge server and compares it with the current global round ω^μ. The lag tolerance mechanism defines a limit τ for stale model versions, allowing stragglers to still contribute to the model aggregation policy. Hence, we compute the difference between the local model version for a given CAV v_i and the global model version at the edge server, and our lag-tolerant distribution principle can be formulated as follows:

$$\omega_i^\mu = \begin{cases} \omega_i^\mu & \text{if } vr(\omega_i^\mu) \geq (\mu - \tau), \\ \omega^\mu & \text{otherwise.} \end{cases} \tag{6}$$

We define a lag tolerance limit $\tau = 1$ model version, which guarantee performance stability in the models of CAVs and prevent outdated CAVs with stale models from participating in the aggregation policy. The lag-tolerant model distribution policy compels the CAVs outdated models to utilize the most recent global model as their starting point for the subsequent round of training. Conversely, for clients deemed as having tolerable lag, they are permitted to build upon their prior local results. In summary, FALCON mandates synchronization only for CAVs categorized as deprecated, allowing tolerable clients to maintain their asynchronous status with the server. Deprecated CAVs are compelled to synchronize to prevent severely outdated local models from adversely affecting the global model.

Furthermore, previous research has emphasized that selecting clients with higher local loss computed at each communication round μ facilitates faster convergence [Jee Cho et al. 2020]. In this context, the FALCON protocol creates a list comprising K representing the number of CAVs to be selected, where K is calculated as $\lceil C * |V| \rceil$, as determined by the hyper-parameter C. This list of potential candidates undergoes an arrangement based on descending order of loss values. Furthermore, the FALCON protocol identifies and selects the CAVs that exhibit the highest loss function value, denoted as $f_i(\omega_i^\mu; x_i^k, y_i^k)$. The FALCON protocol also employs a list of previously chosen clients (Q_K) to prevent the repetition of a limited group of clients. By using this approach, clients who have already been picked in the prior round will be excluded from being selected in the subsequent round.

For example, let us examine a scenario involving two clients with different datasets, one experiencing a notably high loss rate and the other with significantly lower local

loss. In a specific training round μ, the global model acquires more substantial new knowledge, prioritizing its training emphasis on the local data from the client with the highest loss, and this occurs because the client with the highest local loss needs to contribute more time to the model to make it more generic. In this sense, each new communication round repeats the client selection process.

4. Evaluation

In this section, we describe the CAV scenario, including framework, database and simulation details. We evaluate of the FALCON compared to FedCS [Nishio and Yonetani 2019], SEMI-SYNC [Stripelis et al. 2022], and SAFA [Wu et al. 2021]. We also discuss the obtained results in terms of the number of selected clients, the total number of communication rounds, the average aggregation cost, and the global model's accuracy.

4.1. Simulation Description

We conducted simulations using OMNeT++ 5.7.1 as the simulator and FLEXE[Lobato et al. 2022] as the simulation framework. FLEXE is an open-source extension to the vehicular network simulation framework *Veins* which offers researchers a simulation environment to run FL experiments in realistic vehicular scenarios. FLEXE uses the TensorFlow's training interface to split the data for each CAV and train the local model. FLEXE also uses SUMO 1.4.0 as a traffic simulator to simulate the CAVs moving on the road at a random cruise speed.

We employ the Luxembourg SUMO Traffic (LuST) scenario to evaluate the impacts of different semi-synchronous protocols with realistic vehicular mobility trace [Codeca et al. 2017]. LuST is an open-source realistic mobility trace scenario containing 24 hours of mobility data for multiple CAVs, routes, and road lengths. The LuST scenario predetermines vehicle routes, and all CAVs share identical properties, such as size, mean speed, and acceleration. We chose an arterial area from the Luxembourg city center (*i.e.*, a highway scenario), where several city routes converge on this region. We conducted simulations at three different times of the day to represent three network densities in the vehicular environment (*i.e.*, low, medium, and high density), which enables to comprehensively examine the effects of different traffic levels on the efficiency of FL. We conducted 48 simulation runs with different randomly generated seeds, and the results include a 95% confidence interval.

The simulation scenario includes an edge server with a single global model placed in the scenario's center. We assume that all participating CAVs have similar computational capabilities and are willing to contribute data for training the FL model. We set the transmission power to 20 mW for the edge server. We also use a bit rate of 6 Mbit/s at the MAC layer. For vehicles, we changed the transmission power to 15 mW to give a communication range equal to 300 m. The beacon's frequency was 1 Hz. Table 1 summarizes the main simulation parameters used in our evaluation.

Each CAV possesses an ML model tailored to perform an image classification task, such as, required in task for autonomous or self-driving cars. After recognizing the type of objects, the CAV can take appropriate action. For example, if the detected object is classified as a cyclist or pedestrian then the CAV can overtake the detected object at the safe distance. Therefore, recognition or image classification is an important task

for smooth driving in CAVs. In this sense, we employ a publicly available large-scale image databases, namely, Fashion-MNIST (FMNIST)[1]. The FMNIST database consists of 28x28 grayscale images showcasing fashion products across ten categories, sourced from a Zalando article image database. Besides it is a fashion products database, it encompasses a training set of 60 000 examples and a test set of 10 000 samples.

We implemented a Convolutional Neural Network (CNN) composed of two convolution layers with a 5x5 kernel size and 32 filters. A 2x2 max pooling operation follows each convolution layer. The model includes a fully connected layer with a ReLU activation function and a final softmax output layer. Stochastic Gradient Descent updates the model weights, while Sparse Categorical Cross-entropy is the loss function during training. During each communication round, the CAVs train their local models with five epochs for both models.

Tabela 1. Simulation Parameters

	Parameter	Value
communication	Simulation area	1 km^2
	Scenario	LuST
	Traffic density	58 (Low), 157 (Medium), 235 (High)
	Vehicle speed	13.84 m/s (49.82 km/h), St.Dev: 5.27
	Beacon transmission rate	1 Hz
	Transmission power	15 mW
	Reception sensitivity	−110.0 dB
	Bitrate	6 Mbit/s
	Transmission range	300 m
fed. learning	Database	FMNIST
	Database size	50k
	Local training	5 epochs
	Batch size	64
	Loss function	Sparse Categorical Cross-entropy
	Optimizer	Adam

We consider data across CAVs exhibit non-Independent and Non-Identically Distributed (non-IID) characteristics in our evaluation scenario, due to data heterogeneity across various CAVs and ML applications. In this sense, numerous distributed ML training approaches encounter substantial accuracy degradation due to the disparities in data quantity and category distribution within non-IID data. In the non-IID context, we use label distribution skew to characterize the local data distribution among the CAVs, resulting in varying label proportion [Hao et al. 2020]. Specifically, we sample a proportion $p_{k,i} \sim Dir(\beta)$ that represent the instances of class k to the CAV v_i, where $Dir(\beta)$ is a Dirichlet distribution with a concentration parameter $\beta = 0.1$ [Li et al. 2021].

4.2. Simulation results

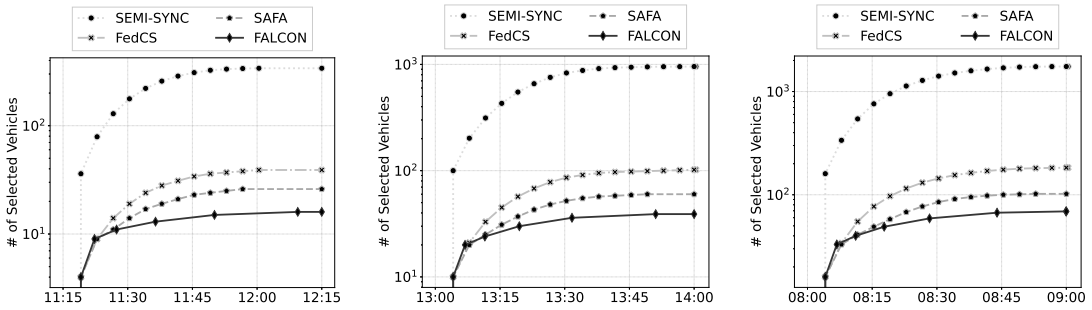
Figure 1 shows the cumulative number of selected clients on each communication rounds for the analyzed semi-synchronous FL protocols under different network density levels. By analyzing the results, it is possible to conclude that FALCON reduces the number

[1]`https://github.com/zalandoresearch/fashion-mnist`

of selected clients by 95.28%, 58.97%, and 38.46% for low density compared to SEMI-SYNC, FedCS, and SAFA, respectively. It is important to note that the number of selected clients increase as the network density increases as well, due to the increase in the number of vehicles available to share the model with the edge server. For instance, FALCON increases the number of selected clients by 76.81% for high density compared to low density.

The performance of FALCON protocol is due to it employing the shared local model's hyper-parameter C and τ to prioritize models with the lag tolerance and highest local loss, which reduces the number of selected clients on each round while keeps the accuracy level similar to other approaches as shown in Figure 4. In this context, FedCS, SEMI-SYNC, and SAFA require more model transmission across the network in each communication round, consuming more network resources. For instance, compared to FALCON, the bit transmission of the SEMI-SYNC, FedCS, and SAFA is about 95.92%, 61.76%, and 35% higher in the medium-density scenario.

Moreover, the SEMI-SYNC protocol outperformed the other methods, mainly attributable to its strategy of choosing all available models within the communication range of the edge server. Conversely, the FedCS protocol employs a hyper-parameter, denoted as C, to select a percentage of the models received by the edge server for aggregation purposes. The number of clients chosen is directly linked to the value of the hyper-parameter C. Lastly, the SAFA protocol adopts a selection mechanism similar to that of FedCS and FALCON, which combines a defined percentage of clients with the lag-tolerance variable to select clients.



(a) Low Density (b) Medium Density (c) High Density

Figura 1. Impact of CAVs density on the number of selected vehicles

Figure 2 demonstrates the impact of the temporal synchronization point on the number of communication rounds conducted by each evaluated protocol. By analyzing the results, it is possible to observe that FALCON achieves 46.15%, 46.15%, and 41.67% reductions, in low density, compared to the SEMI-SYNC, FedCS, and SAFA protocols, respectively. This reduction is attributed to FALCON's utilization of the link duration to determine the next temporal synchronization point, effectively minimizing communication rounds and significantly decreasing the number of transmitted packets by edge server.

It is important to highlight that the protocols SEMI-SYNC, FedCS, and SAFA have a fixed synchronization point, where FedCS and SEMI-SYNC select the synchronization based on a fixed time interval, while SAFA chooses based on the local updates stored in the cache. We refer to the same values established by [Wu et al. 2021] to ensure a

fair definition of this fixed temporal synchronization point in the FedCS and SEMI-SYNC protocols. Hence, FALCON provides lower number of communication rounds, which reduces the model training and usage of computation and communication resource. In the high-density scenario, as shown in Figure 2, FALCON reduces the communication rounds approximately by 52.94%, 52.94%, and 46.66% compared to the FedCS, SEMI-SYNC, and SAFA protocols, respectively.

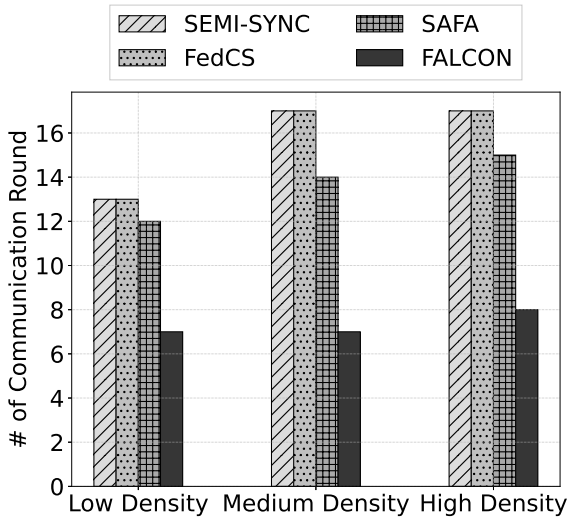


Figura 2. Total number of communication rounds

Figure 3 presents the average aggregation cost for each communication round in milliseconds (ms), offering valuable insights into the direct relationship between the number of selected clients and the total number of communication rounds. This result highlights the significant reduction that the FALCON protocol achieved in terms of minimizing aggregation costs, where FALCON protocol outperforms SEMI-SYNC, FedCS, and SAFA by impressive margins of 96.30%, 64.11%, and 35.96%, respectively. Moreover, it becomes evident that the aggregation cost intricately relates to the number of rounds conducted by FALCON, as demonstrated in Figure 2. Figure 3 effectively illustrates the concurrent reduction in communication and computation costs throughout the FL process, thereby highlighting the overall efficiency of FALCON approach. An intriguing observation emerges when we compare the FedCS and SEMI-SYNC protocols, which share the same number of communication rounds but exhibit an 88% difference in the number of selected CAVs, as indicated in Figures 1 and 2. This disparity is reflected in the aggregation cost, where the FALCON protocol strategically reduces client selection values and the total number of communication rounds.

Finally, Figure 4 presents the learning accuracy results using the FMNIST database to examine the performance of the protocols in a specific AV application. It is crucial to highlight that higher accuracy values within these figures signify superior classification outcomes. Upon analyzing the results in Figure 4(a), we conclude that all protocols exhibit similar accuracy levels, hovering around 83%, in the final round of communication for the classification task using the FMNIST database, regardless of the CAV density. What is particularly noteworthy is the sustained high accuracy values across all protocols. In a standout fashion, FALCON demonstrates remarkable consistency by maintaining an accuracy level of 85.11% in Figure 4(b) while simultaneously reducing the required number

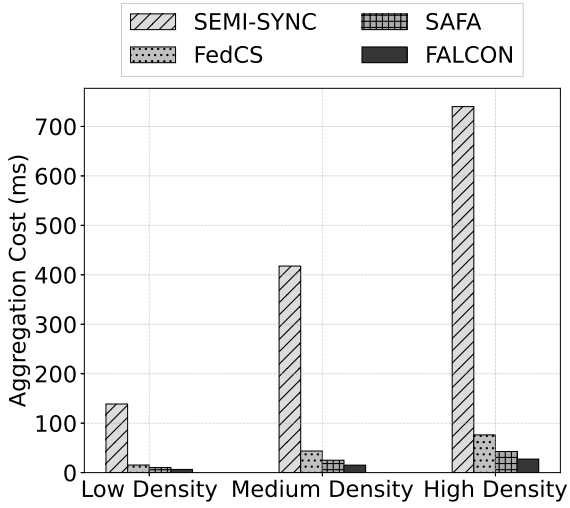
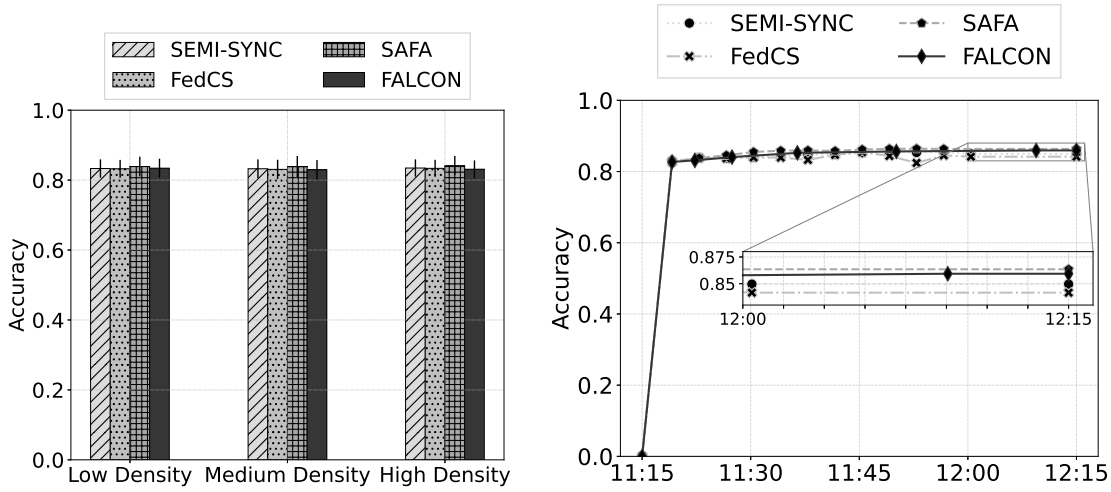


Figura 3. Average aggregation cost of different protocols

of communication rounds.



(a) Average accuracy per CAVs density

(b) Accuracy per round in high density

Figura 4. Impact of CAVs density on global model accuracy in the FMNIST

5. Conclusions and Future Work

In this paper, we have outlined the operation of FALCON, a semi-synchronous FL protocol that considers the temporal link duration between CAVs and the edge server. FALCON considers data periodically transmitted by CAVs operating within the edge server's communication range to calculate the link duration and dynamically set up a temporal synchronization point. Additionally, FALCON relies on a client selection mechanism that considers lag tolerance and prioritizes clients based on their highest local loss. Simulation results show that the dynamic temporal synchronization point affects the number of global model aggregations, the total number of selected clients, and the aggregation cost applied in the edge server. The results indicate FALCON's effectiveness in reducing the number of communication rounds and significantly decreases the number of selected clients, while maintaining the same classification task accuracy.

In future work, we plan to undertake comprehensive assessments of more intricate applications tailored for CAVs and explore various aggregation policies to gain deeper insights into their effects and associated costs. We also intend to incorporate the historical mobility patterns of CAVs when determining the temporal synchronization point and a detailed discussion on scalability to prove the scalability, acknowledging the significance of accounting for diverse mobility scenarios to assess our solutions comprehensively.

Acknowledgment

The authors would like to thank the São Paulo Research Foundation (FAPESP), grants #2015/24494-8 and #2019/19105-3. We also would like to thank to PPI-Softex with support from the MCTI [01245.003479/2024-10].

Referências

Almanifi, O. R. A., Chow, C.-O., Tham, M.-L., Chuah, J. H., and Kanesan, J. (2023). Communication and computation efficiency in federated learning: A survey. *Internet of Things*, 22:100742.

Chellapandi, V. P., Yuan, L., Brinton, C. G., Żak, S. H., and Wang, Z. (2024). Federated learning for connected and automated vehicles: A survey of existing approaches and challenges. *IEEE Transactions on Intelligent Vehicles*, 9(1):119–137.

Chen, Y., Sun, X., and Jin, Y. (2020). Communication-efficient federated deep learning with layerwise asynchronous model update and temporally weighted aggregation. *IEEE Transactions on Neural Networks and Learning Systems*, 31(10):4229–4238.

Codeca, L., Frank, R., Faye, S., and Engel, T. (2017). Luxembourg SUMO Traffic (LuST) Scenario: Traffic Demand Evaluation. *IEEE Intelligent Transportation Systems Magazine*, 9(2):52–63.

Damaj, I. W., Serhal, D. K., Hamandi, L. A., Zantout, R. N., and Mouftah, H. T. (2021). Connected and Autonomous Electric Vehicles: Quality of Experience survey and taxonomy. *Vehicular Communications*, 28:100312.

Hao, J., Zhao, Y., and Zhang, J. (2020). Time Efficient Federated Learning with Semi-asynchronous Communication. In *IEEE 26th International Conference on Parallel and Distributed Systems (ICPADS)*, pages 156–163.

Jee Cho, Y., Gupta, S., Joshi, G., and Yağan, O. (2020). Bandit-based Communication-Efficient Client Selection Strategies for Federated Learning. In *2020 54th Asilomar Conference on Signals, Systems, and Computers*. IEEE.

Li, Q., He, B., and Song, D. (2021). Model-contrastive federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10713–10722.

Liang, F., Yang, Q., Liu, R., Wang, J., Sato, K., and Guo, J. (2022). Semi-Synchronous Federated Learning Protocol With Dynamic Aggregation in Internet of Vehicles. *IEEE Transactions on Vehicular Technology*, 71(5):4677–4691.

Liu, S. and Gaudiot, J.-L. (2022). Rise of the Autonomous Machines. *Computer*, 55(1):64–73.

Lobato, W., Costa, J. B. D. D., Souza, A. M. d., Rosário, D., Sommer, C., and Villas, L. A. (2022). FLEXE: Investigating Federated Learning in Connected Autonomous Vehicle Simulations. In *IEEE 96th Vehicular Technology Conference (VTC-Fall)*. IEEE.

Lv, P., Xu, W., Nie, J., Yuan, Y., Cai, C., Chen, Z., and Xu, J. (2023). Edge Computing Task Offloading for Environmental Perception of Autonomous Vehicles in 6G Networks. *IEEE Transactions on Network Science and Engineering*, 10(3):1228–1245.

Ma, Q., Xu, Y., Xu, H., Jiang, Z., Huang, L., and Huang, H. (2021). FedSA: A Semi-Asynchronous Federated Learning Mechanism in Heterogeneous Edge Computing. *IEEE Journal on Selected Areas in Communications*, 39(12):3654–3672.

Niknam, S., Dhillon, H. S., and Reed, J. H. (2020). Federated learning for wireless communications: Motivation, opportunities, and challenges. *IEEE Communications Magazine*, 58(6):46–51.

Nishio, T. and Yonetani, R. (2019). Client Selection for Federated Learning with Heterogeneous Resources in Mobile Edge. In *IEEE International Conference on Communications (ICC)*. IEEE.

Song, R., Zhou, L., Lakshminarasimhan, V., Festag, A., and Knoll, A. (2022). Federated Learning Framework Coping with Hierarchical Heterogeneity in Cooperative ITS. In *IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE.

Stripelis, D., Thompson, P. M., and Ambite, J. L. (2022). Semi-Synchronous Federated Learning for Energy-Efficient Training and Accelerated Convergence in Cross-Silo Settings. *ACM Transactions on Intelligent Systems and Technology*, 13(5):1–29.

Sun, J., Li, A., Duan, L., Alam, S., Deng, X., Guo, X., Wang, H., Gorlatova, M., Zhang, M., Li, H., and Chen, Y. (2022a). FedSEA: A Semi-Asynchronous Federated Learning Framework for Extremely Heterogeneous Devices. In *Proceedings of the 20th ACM Conference on Embedded Networked Sensor Systems (SenSys)*. ACM.

Sun, X., Yu, F. R., and Zhang, P. (2022b). A Survey on Cyber-Security of Connected and Autonomous Vehicles (CAVs). *IEEE Transactions on Intelligent Transportation Systems*, 23(7):6240–6259.

Wang, S., Li, C., Ng, D. W. K., Eldar, Y. C., Poor, H. V., Hao, Q., and Xu, C. (2023). Federated deep learning meets autonomous vehicle perception: Design and verification. *IEEE Network*, 37(3):16–25.

Wu, W., He, L., Lin, W., Mao, R., Maple, C., and Jarvis, S. (2021). SAFA: A Semi-Asynchronous Protocol for Fast Federated Learning With Low Overhead. *IEEE Transactions on Computers*, 70(5):655–668.

You, C., Feng, D., Guo, K., Yang, H. H., Feng, C., and Quek, T. Q. S. (2023). Semi-Synchronous Personalized Federated Learning Over Mobile Edge Networks. *IEEE Transactions on Wireless Communications*, 22(4).

Zhang, X., Liu, J., Hu, T., Chang, Z., Zhang, Y., and Min, G. (2023). Federated learning-assisted vehicular edge computing: Architecture and research directions. *IEEE Vehicular Technology Magazine*, pages 2–11.