SUMO User Conference 2023

Conference paper

https://doi.org/10.52825/scp.v4i.221

Published: 29 June 2023

SUMO Simulations for Federated Learning in Communicating Autonomous Vehicles

A Survey on Efficiency and Security

Levente Alekszejenkó[1] [https://orcid.org/0000-0002-3196-1950], and
Tadeusz Dobrowiecki[1] [https://orcid.org/0000-0002-8307-5096]

[1] Department of Measurement and Information Systems,
Budapest University of Technology and Economics,
Budapest, Hungary

Abstract: In transportation, a vehicle's route is one of the most private information. However, to mutually learn some phenomena in a city, for example, parking lot occupancies, we might have to reveal information about it. In this paper, we focus on assessing the privacy loss in a vehicular federated machine learning system. For the analysis, we used the Monaco SUMO Traffic Scenario (MoST). We also used the simulation inputs as statistical data to calculate privacy loss metrics. Results show that a vehicular federated machine learning system may pose a smaller privacy threat than individual learning, but its performance is lower compared to a centralized learning approach.

Due to the vast amount of data and processing time, we also describe a method to build a Docker image of SUMO together with a software client-server architecture for SUMO-based learning systems on multiple computers.

Keywords: federated learning, communicating vehicles, efficiency, security

1. Introduction

Modern vehicles carry an abundance of sensors. Additionally, recent developments in machine learning made it possible to process even more data about the transportation infrastructure than at any time in the past.

As it would be impossible to store all this information, we shall build compact, predictive models to utilize all the knowledge. These models are valuable inputs to optimize the traffic in a city. In the future, these models can also be parts of autonomous driving algorithms.

There are several ways to build machine learning models. In this paper, we consider that vehicles try to learn such phenomena *individually* and as *participants of a federated learning system*. Like in mobile edge networks, federated learning might have many advantages also in vehicular learning, including low latency, privacy, and efficient use of network bandwith [1]. Therefore, with the help of the Monaco SUMO Traffic

Scenario [2] (MoST) simulations in Eclipse SUMO [3], *we analyze the performance of these learning approaches and compare them to a centralized method in which the infrastructure itself collects the data and trains the model.*

Additionally, while sharing data, even in federated learning, we shall be cautious not to share too much personal information, especially route origins, and destinations. As the route is private information, keeping it hidden from unauthorized access is both a data security problem and a legal obligation. In this paper, we also *evaluate the mentioned learning schemes from a data privacy point of view*.

In the following, in Section 2, we review the relevant literature. Section 3 describes the parametrization and properties of the used simulation. The mentioned learning schemes are described in Section 4–6. Finally, Section 7 concludes the paper. Moreover, long-running computations required the dockerization of SUMO and a TraCI script, which we describe in Appendix A. Appendix B, in addition, proposes a method to distribute SUMO and machine learning workers between multiple computers.

For further development, we published our source codes at https://github.com/alelevente/sumo_sec.

2. Related works

Crowdsensing complex traffic phenomena, such as parking place availability [4], seems to be a promising new way of optimizing the traffic flow in cities. However, sharing data on Vehicular Ad-hoc Networks (VANETs) also poses some technical challenges. We have to solve the problem of secure data exchange while respecting the limits of the available network bandwidth [5].

Several papers focus on simulating cyberattacks on Connected Autonomous Vehicles (CAVs) and VANETs, including active [6], [7] or Distributed Denial of Service attacks [8]. These simulations usually use SUMO to evaluate vehicle movements.

Moreover, federated learning [9] also gained importance in machine learning on mobile devices in recent years. Federated learning ensures that its participants shall not exchange their training data. It guarantees a certain level of security and also reduces communication costs. Therefore, a vehicular network can take advantage of the federated scheme as well [10]. Unfortunately, the security level in a federated learning scenario highly depends on identically distributed datasets: with non-IID (not Identically and Independently Distributed) data, it is possible to carry out various attacks against the participants [11]. However, there are numerous countermeasures to such attacks; they might not be applied onboard a vehicle due to the limited amount of power and computational resources.

In this paper, we measure the parking lot occupancy by multiple measurement setups. We evaluate the idea that vehicular crowdsensing schemes can function as *cooperation-based* location privacy-ensuring methods to hide routes, which is a critically privacy-sensitive property [12], of the vehicles. To infer this information, it is enough to suppose a *passive* attack carried out by an honest-but-curious party.

3. Simulation

To obtain measurement results, we used the Eclipse SUMO traffic simulation tool with the MoST scenario. We have changed the simulation timestep from 0.25 s to 1.0 s to reduce the computation time.[1] We parametrized the simulation to have a maximum of 15 minutes of departure offset stochastically for each vehicle. It models that the population follows some daily routine; however, individuals might not depart at the same time each day. That yields an average traffic demand similar to what the MoST defines with a certain perturbance around it.

SUMO is supposed to provide training data on parking lot occupancies measured by moving cars. Hence, during the simulation, we recorded the position of each vehicle with 0.1 Hz frequency through the Traffic Control Interface (TraCI). We also recorded the occupancy of parking lots once each minute. As the MoST originally defined the scenario, the measurement times ranged from 4:00 a.m. to 2:00 p.m.

These measurements were repeated 60 times, corresponding to a measurement series of 3 months of working day data. As running a single simulation takes approximately 1 hour on a PC, we sped it up by running multiple simulations in parallel. To this end, as described in Appendix A, we created a Docker image that runs a TraCI client and a SUMO instance.

We assume that a vehicle *knows* a parking lot's occupancy if it passes through an edge with a center that is not more than 50 m away from the edge of the parking lot. This distance can be understood as the communication or visual range of the cars. After the simulations had terminated, we determined for each vehicle its measured parking lots and the actual occupancy values when the cars were nearby.

4. Centralized learning

We trained a neural network only with the simulated parking lot occupancy data to obtain a baseline model. Figure 1 illustrates the scheme of this learning setup. For large parking garages, it is possible to implement such an approach by collecting each garage's occupation data to train a neural network on a remote server. However, this centralized setup requires an infrastructure that detects whether a parking lot is free or occupied. Hence, real-world implementation would be impractical due to its installation and operational costs.

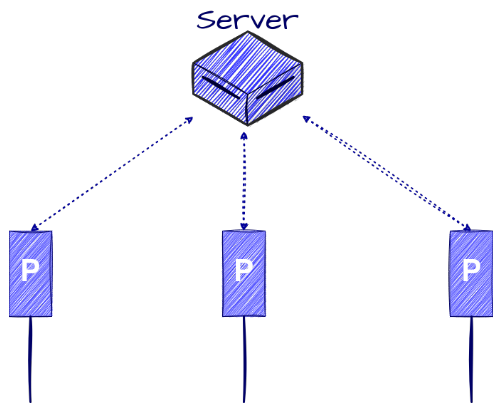


Figure 1. Scheme of the centralized learning approach. The server has connection to each parking lot; hence, it can collect data from them, and use this data for learning a predictive model.

[1]Because of this, accidents might occur in SUMO due to the small τ values. In our case, this is only considered to be a random event without any further investigation.

4.1. Neural network architecture and parameters

The training data consisted of $\{\texttt{ID of parking lot, timestamp, occupancy}\}$ records, from which the first two properties were the training features and the $\texttt{occupancy}$ column was the label to predict. We normalized the timestamps (ranging from 4:00 to 14:00) to the $[0.0, 1.0]$ interval, and we standardized the occupancy data (ranging from 0.0 (empty) to 1.0 (full)) by the mean and standard deviation values measured on the first simulated day. As there are relatively few parking lots in the MoST, we were able to represent them by a one-hot-encoding.

The used neural network is quite simple: it consists of 3 hidden, fully connected layers (with $[200, 100, 20]$ neurons respectively) with rectified linear unit activation (ReLU) introduced as eq. (2) in [13]. As the parking lot occupancy prediction is a regression problem, a mean squared error (MSE) loss function suffices. Finally, we chose the RMSprop optimizer [14] because we empirically found that it results in a slightly faster convergence.

Out of the 60 days of measurement, the first 55 days served as training data, and the rest was the test data. During the training process, 30% of the samples were the validation set. To achieve maximum performance as well as to avoid overfitting, we applied an early stopping mechanism that terminated the training process if there was no significant improvement (0.0001 improvements in MSE loss on the training set within 3 consecutive epochs).

4.2. Performance of the centralized model

In this centralized setup, we can utilize the whole dataset provided by SUMO. Therefore, the expectation might be that this approach performs outstandingly well in the parking lot occupancy prediction.

The results confirm this anticipation: on the last 5 test days, the model produces an MSE loss value smaller than 0.002. As, e.g., Figure 2 shows, the estimation fits the measured value of the parking lot occupancy with minimal error.

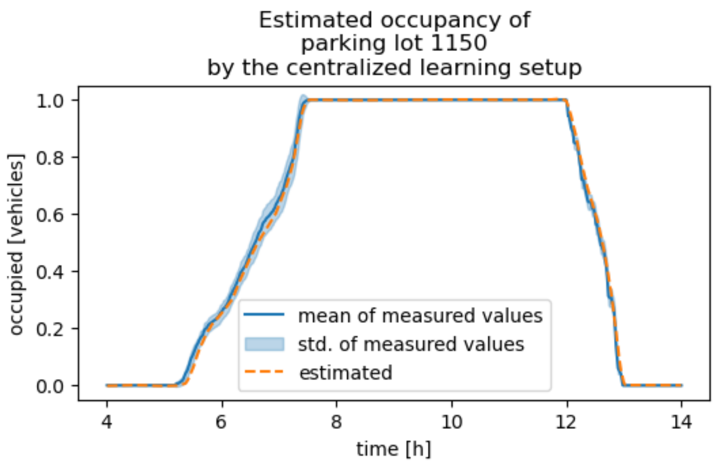


Figure 2. Parking lot occupancy estimation of the centralized approach

5. Individual learning

As we have noted at the beginning of Section 4, a centralized free parking lot prediction system would require many sensors and communication to track the occupancies. Consequently, this results in a high installation and operational cost, especially in curbside parkings.

Fortunately, as modern vehicles will carry more and more sensors, they might be able to measure the occupancy rate of the parking lots on the fly. However, such records can require a vast amount of storage space or have a high communication cost when sending them to a remote server. Therefore, we have to represent this information more compactly. For example, a trained machine-learning model can efficiently encode such data in a predetermined storage space (engineers can determine the number of model parameters at design time). Based on this idea, we tested how an individual vehicle, see Figure 3, can measure and learn the occupancy of the parking lots.

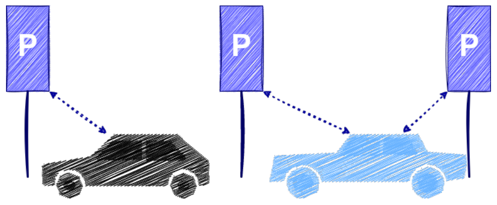


Figure 3. Scheme of the individual learning approach. Individual vehicles try to learn the occupancy of the parking lots.

In the following, for illustration, we will see how vehicle `commercial_3-1_98` from the MoST learns and what performance it achieves.

5.1. Performance of the individual training process

To ensure that only the access to the data influences the training process, we trained an identical neural network to the one proposed in Section 4.1. Hence, the only difference is, in this case, that the neural network utilized only the measurement data of the parking lots which lay along the route of the `commercial_3-1_98` vehicle. The edges from which these parking lots are observable according to Section 3 are shown in Figure 4.

Naturally, we shall not expect that the parking lot occupancies for the whole time range can be accurately estimated. We can only assume that for the measured parking lots at the observation time, the vehicle will be able to approximate the occupancy values. Figure 5 confirms this hypothesis: Figure 5a illustrates how the `commercial_3-1_98` vehicle can predict the occupancy of parking lot no. 1140, which lays along its path. Around the observation time (depicted as a red line), the vehicle can more or less accurately estimate that this specific parking is full. On the other hand, parking lot no. 1101 is not in the knowledge base of the `commercial_3-1_98` vehicle; therefore, it cannot accurately estimate its occupancy, see Figure 5b.

As a vehicle often follows identical routes at the same time of the day, corresponding to the daily routine of its owner, even this model might be helpful to recommend parking lots that are free with high probability at a given time.

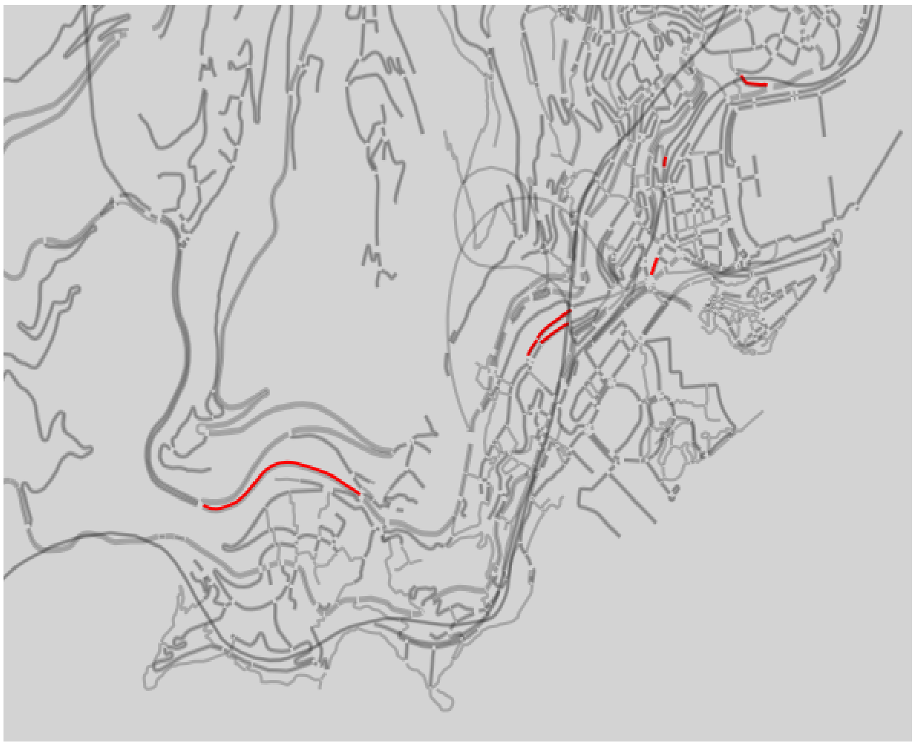


Figure 4. A section of the map being simulated by the MoST. Edges on which vehicle `commercial_3-1_98` measured the occupancy of the parking lots are depicted in red. By connecting these points, we might obtain information about the vehicle's route.

5.2. Vulnerabilities of the individual learning scheme

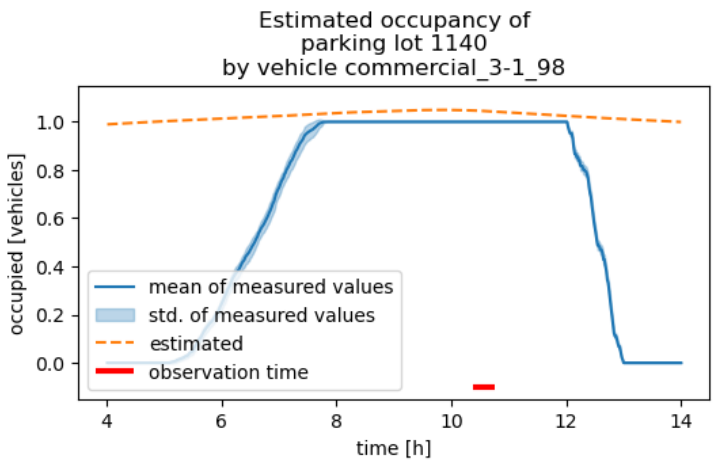
The trained neural network efficiently encodes the measured occupancy data; therefore, it can also be helpful to another vehicle unfamiliar with the environment (at a given time in a district). Therefore, cars might share their trained models.

Supposing that the receiver might be malicious, we shall evaluate the possible vulnerabilities of such a data-sharing technique. To this end, we assume that the receiver is honest-but-curious, trying to infer the route, i.e., the measured parking lots and the observation time of the sender vehicle. The receiver is also an oracle possessing all the occupancy data of the parking lots.

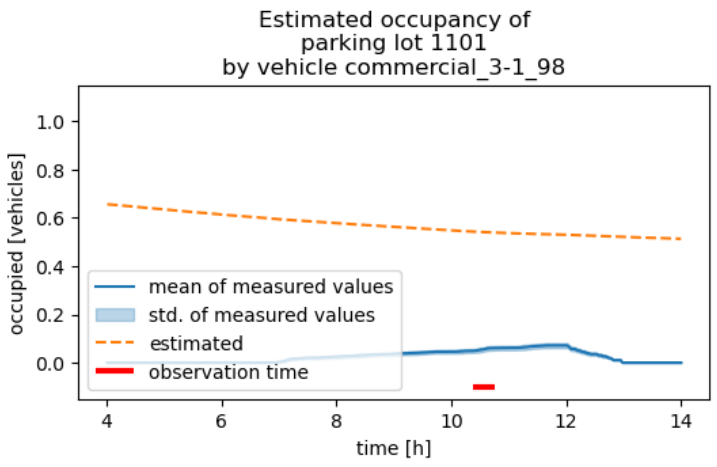
By calculating the prediction accuracy values per parking lot, the attacker can successfully identify some measured parking lots, see Figure 6. We estimated that a vehicle while following its route measures 5.22 parking lots on average in the MoST. Therefore, an attacker can select 5 parking lots having the lowest prediction loss values as the inferred route of the sender. That gives the malicious receiver a map similar to Figure 4, on which it might be able to approximate the path of the sender by connecting the edges with rational and legal routes.

To approximate the observation time, the attacker can apply the following heuristic: first, it selects 5 parking lots, of which predicted occupations are the most accurate. After that, it computes the prediction accuracy $l_p(t)$ of these parking lots per timestep. To smoothen the achieved curve, it can apply a moving average with a window size of e.g. 60 minutes. Let the smoothened curve be $\hat{l}_p(t)$. Then the $\hat{t}_m$ observation time estimate can be defined as:

$$\hat{t}_m = \arg\min_t \hat{l}_p(t). \tag{1}$$



(a) Estimation of the occupancy of a known parking lot



(b) Estimation of the occupancy of an unknown parking lot

Figure 5. Parking lot occupancy estimation of an individual vehicle

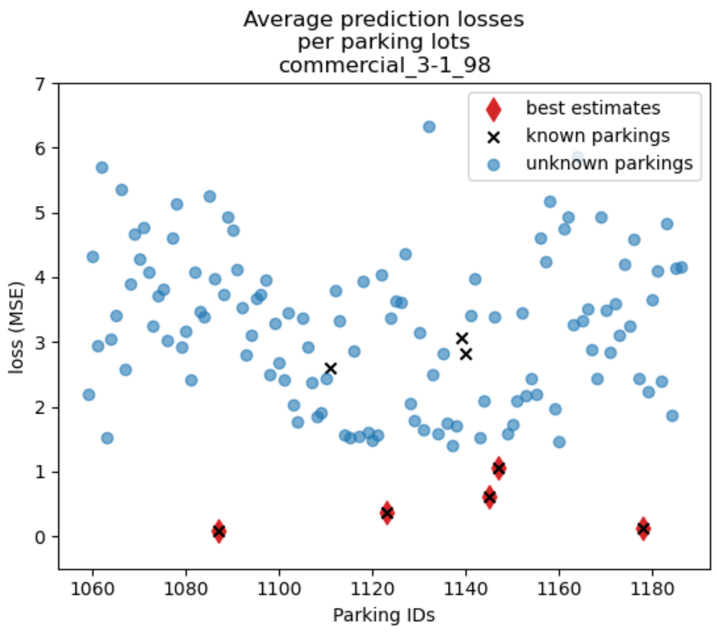


Figure 6. Average prediction accuracies for known and unknown parking lots.

As Figure 7 illustrates, the heuristic of (1) can approximate the observation time of `commercial_3-1_98` with 8 minutes difference. Considering that the observation time offset follows a uniform distribution in the $[0, 15]$ minutes range, this difference is smaller than the offset range.

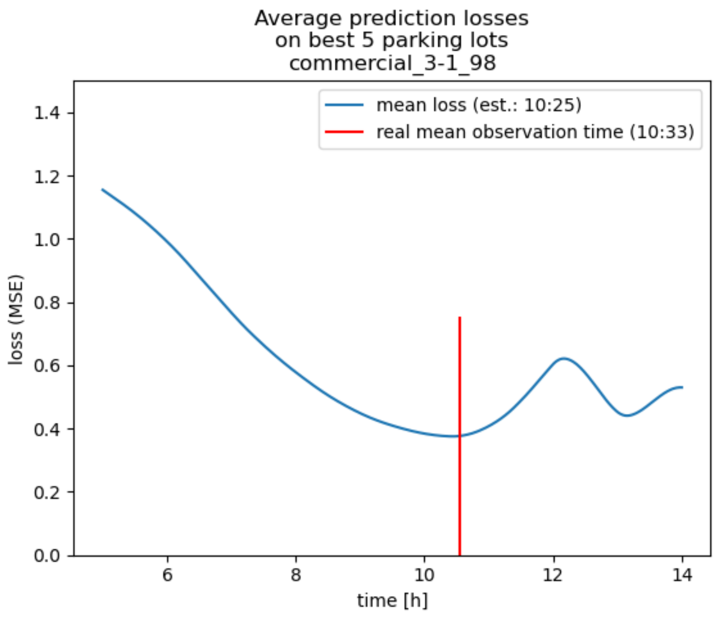


Figure 7. Average prediction accuracies for known parking lots during the simulation time. The minimum of the blue curve is at 10:25 which is the estimate of the observation time according to (1).

6. Federated learning

As both the centralized and individual learning schemes have their drawbacks, it might be fruitful to combine them. It would result in cheap measurements done by the vehicles and an accurate model on the server side. It is the idea of the federated learning scheme, see Figure 8.

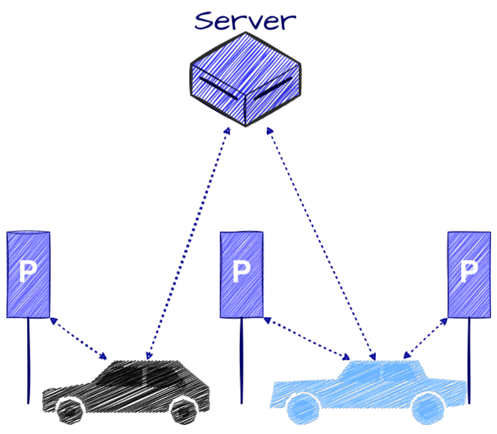


Figure 8. Scheme of the federated learning approach. Individual vehicles train their own models which they send to the server. The server aggregates the models and shares this federated model with the participating vehicles.

Considering that the data provided by the SUMO simulations require approximately 240 GiB of storage space, its processing even on a high-end PC is not feasible. Therefore, we sampled 10% of the vehicles. In this way, we had measurements of approximately 4000 vehicles. This sampling can also represent the situation when only a portion of the cars can measure complex phenomena such as the parking lot occupancy.

After each simulated day, the server randomly chose 50 participant vehicles. These participants received the actual federated model (in the beginning, it was initialized randomly) and trained their local update models, identical to the one described in Section 4.1, with all their data from previous measurements until the time of the training. The participants send the update to the remote server which aggregates them by the `FedAvg` algorithm [9] to obtain the next iteration of the federated model.

Unfortunately, operating this learning scheme requires either plenty of time or the parallelization of the learning tasks. To this end, we developed a software architecture that distributes the computation among multiple PCs. Appendix B highlights the main components and design concepts of the architecture.

6.1. Performance of the federated learning scheme

We expect that the accuracy of the federated learning scheme shall converge to the performance of the baseline, centralized approach. However, there might be some rarely visited parking lots. Therefore, the training data of the federated learning scheme might be sparser than in the centralized approach. That can reduce the numeric performance of the federated system, which achieves an average MSE loss of a little bit above of 1.0 on the test data.

Although this loss value is 3 magnitudes higher than the baseline, a significant part of the imprecision comes from the prediction error of such remote parking places. In real life, we may tolerate this kind of mistake because rarely visited parking lots are usually empty. Hence, it seems more important to predict accurately the occupancy of frequently used parking facilities. Figure 9 illustrates the performance of the federated system on an often-used parking lot. As we can see, the shape of the prediction curve is similar to the real one, but it does not fit as well as that in Figure 2. Due to the limited datasets of the participating vehicles, this process requires either more communication rounds or more participants to achieve the convergent state.

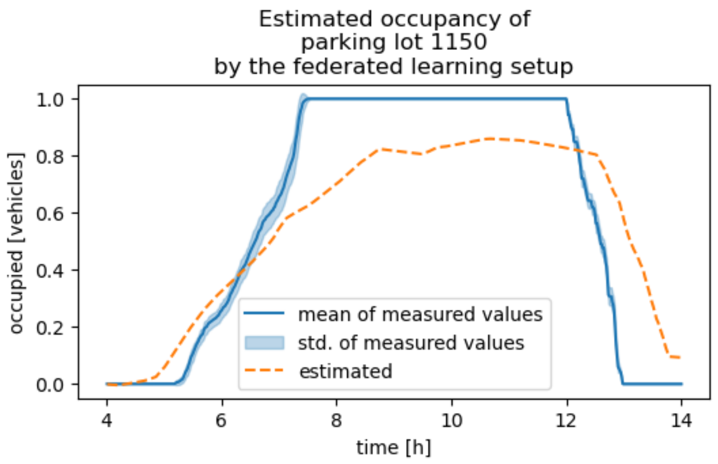


Figure 9. Parking lot occupancy estimation of the federated learning approach

6.2. Vulnerabilities of the federated learning scheme

Considering that the federated learning scheme also incorporates data sharing, we shall investigate the possible security vulnerabilities. Again, we are interested in the routes of the vehicles, i.e., the measured parking lots and the observation times.

The whole federated system slowly converges; therefore, we shall concentrate on the participants' updates. The updates reflect the gradients of the loss function at the participants; consequently, they contain implicit information from the participants' training datasets [9]. Hence, if we compare the performance of the received updates with the sent federated model, we might extract the route of the participant.

To evaluate the success rate of such a comparison-based honest-but-curious attacker, we defined its accuracy as follows. For each participant, the attacker carries out an inference fundamentally similar to the methods described in Section 5.2. The difference is that, in this case, the attacker evaluates the performance of both the federated and the participant models per parking lot. Let us denote the MSE prediction loss of the federated model on the i^{th} parking lot as $l_t^{(f)}(i)$, and the MSE prediction loss of a participant's model on this parking lot as $l_t^{(p)}(i)$. We suppose that the models perform better on the range of the training data ($l_t^{(p)}(i) < l_t^{(f)}(i)$ if the participant vehicle measured parking lot no. i). Therefore, (2) gives a heuristic that the participant is likely measured parking lot no. j:

$$j = \arg\min_i \left(l_t^{(p)}(i) - l_t^{(f)}(i)\right). \tag{2}$$

Let us collect the 5 best parking lots by the above heuristic. Then, we can check how many of these collected parking lots are in the measured parking lot list of the given participants. This ratio will be the accuracy of the attacker: e.g., the accuracy will be 1.0 if all 5 selected parking lots are in the list, up to 0.0 in case these two sets are disjunct. Figure 10 illustrates the curve of this accuracy value. The heuristic performs surprisingly well which proves that in the beginning of the training process, an honest-but-curious attacker can succesfully infer which parking lots were measured by a participating vehicle. Moreover, as the linear trend estimate indicates, the success

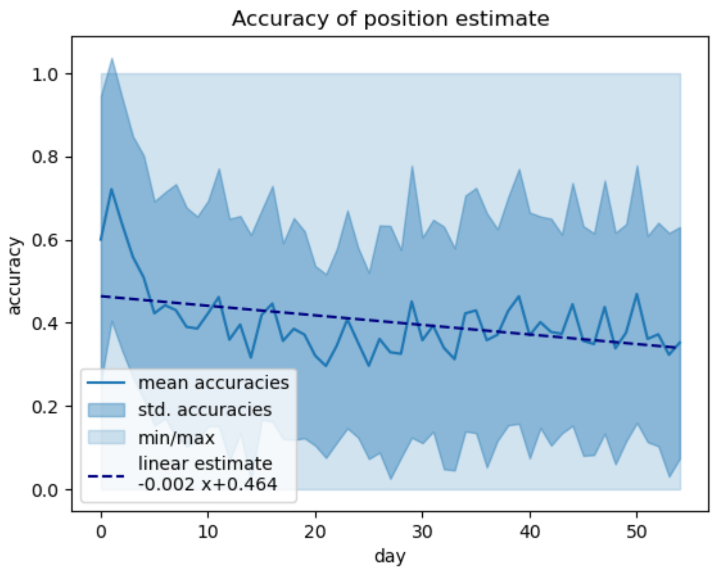


Figure 10. Position inference success rates

rate of the attacker decreases as more and more training rounds are performed. This is due to the convergence of the federated system: a participant in a well-trained scheme would not be distinguished from the other participants. However, we shall note that as a real-world traffic infrastructure constantly changes, we shall not assume that such a convergent state exists.

Moreover, we also tried to estimate the observation times of the participating vehicles. For this approximation, we calculated the $\hat{l}_p^{(f)}(t)$ and $\hat{l}_p^{(p)}(t)$ smoothened MSE loss values on the identified 5 best parking lots per timestep of the federated and the participant models respectively, similarly as in Section 5.2. Then, the observation time estimation $\hat{t}_m^{(p)}$ can be defined as:

$$\hat{t}_m^{(p)} = \arg\min_t \left(\hat{l}_p^{(p)}(t) - \hat{l}_p^{(f)}(t)\right). \tag{3}$$

Figure 11 illustrates the prediction power of the heuristic based on (3). As depicted, the average prediction accuracy oscillates around 2 hours. Consequently, in half of the measurement cases, the approximation error is even smaller than that. As the 1σ standard deviation range shows, the estimate often might be accurate, leaving only a marginal error. It concludes that an honest-but-curious attacker in a federated system may successfully infer both observation time and position.

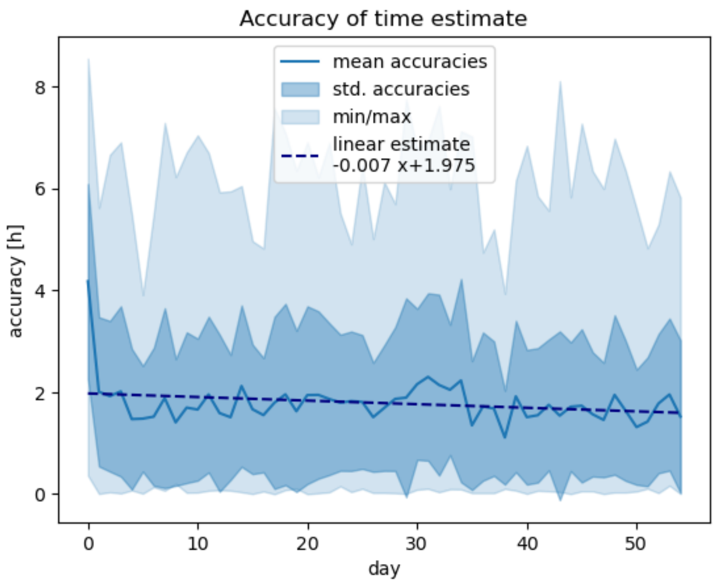


Figure 11. Observation time inference accuracies

As the linear trend indicates it in Figure 11, the observation time estimate gets more accurate as the system trains. That can have various explanations. The first one is that the federated system is yet to converge; therefore, if we perform more training rounds, the probability of a successful observation time inference will decrease. The second possible explanation relies on the execution order: first, a participant trains its local model, then sends back its updates. The malicious server evaluates this update and finally aggregates the received models into the federated one. In this order, a participant can more accurately predict a specific parking lot at a given time than the federated system, explaining why the trendline does not increase. Lastly, it is also possible that we do not have enough data points to achieve a stable, constant value. However, it is possible to proceed with the measurements; operating the federated learning system is computationally really demanding.

7. Conclusion

In this paper, we presented how communicating vehicles can measure and learn a complex phenomenon, i.e., the parking lot occupation of the city. To obtain the input data, we ran Eclipse SUMO multiple times with the MoST scenario. Unfortunately, running the simulations requires plenty of time. To mitigate the time demand, we created a Docker image containing Eclipse SUMO and a TraCI script that can run in parallel in several instances on a PC. As connecting this system with machine learning tools is also challenging, we present a possible software architecture that allows us to distribute the SUMO-based learning system among various computers.

We conclude that a *centralized learning scheme can outperform a federated one*, but its real-world implementation is not feasible due to the need for thousands of sensors in a city. On the other hand, vehicles might learn the parking lot occupancies in a specific district and time of the day; such individually trained models cannot perform well outside the range of their training datasets. Assuming that this neural network model is the compressed version of the collected data, it might be worth sharing it with other vehicles. But it should be emphasized that this data sharing poses a potential privacy risk as an honest-but-curious partner can infer the vehicle's route and observation time.

We also tested a federated learning scheme to combine the benefits of cheap data collection and acceptable model performance. However, such *a federated system is neither entirely secure, especially not at the beginning of the training process*. It is a challenging task to make this federated measurement and learning system protected and well-performing. To this end, our future research focuses on solving that problem.

A. Creating a Docker image with SUMO and TraCI

As running even one instance of the MoST scenario in Eclipse SUMO takes approximately one hour, running it in multiple instances in parallel is quite beneficial. Unfortunately, we ran into a problem[2] when we tried to execute our TraCI script either multithreaded or in separate processes.

As a workaround, we created a Docker image containing SUMO and our TraCI script to collect measurement data. In our GitHub repository, one can find a Dockerfile that creates an Ubuntu-based SUMO installation. Besides pulling the latest SUMO version, the resulting image will incorporate the measurement script too.

B. A multi-computer client-server architecture for SUMO-based learning systems

The operation of the described federated system requires training multiple neural network agents. Unfortunately, it also consumes a significant amount of time even on a high-end PC (with AMD Ryzen 7 3800X CPU, 32 GiB of RAM, and an Nvidia RTX3060 GPU). Moreover, we used Tensorflow to implement the neural network, which, for some unknown reason, does not support multithreaded or process-based parallel training. Therefore, we designed a framework to distribute the workload among various computers, see Figure 12. This approach is fundamentally similar to the parallel training described in [15]. However, a reinforcement learning process heavily depends on the

[2]As of Eclipse SUMO version 1.15.0, the problem may be somehow related to multiprocessing, as the circumstances of getting a TraCI error were not deterministic.

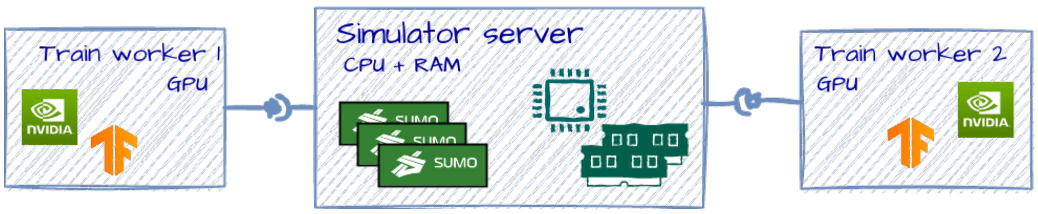


Figure 12. Schematics of a distributed learning system based on SUMO simulations

simulation speed; in our case, the performance bottleneck is the training time of a neural network.

One can deploy multiple SUMO containers, described in Appendix A, on a computer. As SUMO efficiently uses the system memory and runs most of the time on a single CPU core, the `simulator server` might not necessarily be a high-performance computer.

Moreover, training neural networks can be much faster on computers with GPU. To take advantage of it, we can create several `train workers`. These train workers shall operate a simple HTTP server, e.g., implemented by Flask, and provide services such as training a neural network. Or use the neural network to predict a value. These services can be accessed by calling the corresponding HTTP requests. As the interface is through HTTP protocol, we can deploy `train workers` to multiple computers. For more complex tasks and setups, one might also place the `train workers` into Docker containers and build up a Kubernetes-based system.

Data availability statement

As input data, we have used the Monaco SUMO Traffic Scenario [2] in our simulations which one can access at `https://github.com/lcodeca/MoSTScenario`.

Underlying and related material

The source codes producing the presented results are available at: `https://github.com/alelevente/sumo_sec`.

Author contributions

Levente Alekszejenkó is a Ph.D. Student under the supervision of Tadeusz Dobrowiecki. As a part of Mr. Alekszejenkó's Ph.D. research, he conducted the investigation and created the visualization presented in this paper. Both authors are responsible for the presented concepts and methodologies. Prof. Dobrowiecki also had many constructive commentaries on the manuscript written by Mr. Alekszejenkó in the pre-publication stage.

Competing interests

The authors declare that they have no competing interests.

Funding

This research was funded by the National Research, Development, and Innovation Fund of Hungary under Grant TKP2021-EGA-02.

This study was also supported by OTKA 139330 and the European Union project RRF-2.3.1-21-2022-00004 within the framework of the Artificial Intelligence National Laboratory.

References

[1] W. Y. B. Lim, N. C. Luong, D. T. Hoang, *et al.*, "Federated learning in mobile edge networks: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 3, pp. 2031–2063, 2020. DOI: 10.1109/COMST.2020.2986024.

[2] L. Codeca and J. Härri, "Monaco SUMO Traffic (MoST) Scenario: A 3D Mobility Scenario for Cooperative ITS," in *SUMO 2018, SUMO User Conference, Simulating Autonomous and Intermodal Transport Systems, May 14-16, 2018, Berlin, Germany*, B, May 2018.

[3] P. A. Lopez, M. Behrisch, L. Bieker-Walz, *et al.*, "Microscopic traffic simulation using sumo," in *The 21st IEEE International Conference on Intelligent Transportation Systems*, IEEE, 2018. [Online]. Available: https://elib.dlr.de/124092/.

[4] F. Bock, S. Di Martino, and A. Origlia, "Smart parking: Using a crowd of taxis to sense on-street parking space availability," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 2, pp. 496–508, 2020. DOI: 10.1109/TITS.2019.2899149.

[5] H. Khayyam, B. Javadi, M. Jalili, and R. N. Jazar, "Artificial intelligence and internet of things for autonomous vehicles," R. N. Jazar and L. Dai, Eds., pp. 39–68, 2020. DOI: 10.1007/978-3-030-18963-1_2.

[6] S. Iqbal, P. Ball, M. H. Kamarudin, and A. Bradley, "Simulating malicious attacks on VANETs for connected and autonomous vehicle cybersecurity: A machine learning dataset," in *2022 13th International Symposium on Communication Systems, Networks and Digital Signal Processing (CSNDSP), Porto, Portugal, July 20-22. 2022.*, 2022, pp. 332–337. DOI: 10.1109/CSNDSP54353.2022.9908023.

[7] P. Sharma, D. Austin, and H. Liu, "Attacks on machine learning: Adversarial examples in connected and autonomous vehicles," in *2019 IEEE International Symposium on Technologies for Homeland Security (HST)*, 2019, pp. 1–7. DOI: 10.1109/HST47167.2019.9032989.

[8] M. Türkoğlu, H. Polat, C. Koçak, and O. Polat, "Recognition of DDoS attacks on SD-VANET based on combination of hyperparameter optimization and feature selection," *Expert Systems with Applications*, vol. 203, p. 117 500, 2022, ISSN: 0957-4174. DOI: 10.1016/j.eswa.2022.117500.

[9] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, A. Singh and J. Zhu, Eds., ser. Proceedings of Machine Learning Research, vol. 54, PMLR, Apr. 2017, pp. 1273–1282. [Online]. Available: https://proceedings.mlr.press/v54/mcmahan17a.html.

[10] J. Posner, L. Tseng, M. Aloqaily, and Y. Jararweh, "Federated learning in vehicular networks: Opportunities and solutions," *IEEE Network*, vol. 35, no. 2, pp. 152–159, 2021. DOI: 10.1109/MNET.011.2000430.

[11] A. Blanco-Justicia, J. Domingo-Ferrer, S. Martínez, D. Sánchez, A. Flanagan, and K. E. Tan, "Achieving security and privacy in federated learning systems: Survey, research challenges and future directions," *Engineering Applications of Artificial Intelligence*, vol. 106, p. 104 468, 2021, ISSN: 0952-1976. DOI: 10.1016/j.engappai.2021.104468.

[12] H. Jiang, J. Li, P. Zhao, F. Zeng, Z. Xiao, and A. Iyengar, "Location privacy-preserving mechanisms in location-based services: A comprehensive survey," *ACM Comput. Surv.*, vol. 54, no. 1, Jan. 2021, ISSN: 0360-0300. DOI: 10.1145/3423165.

[13] K. Fukushima, "Cognitron: A self-organizing multilayered neural network," *Biological Cybernetics*, vol. 20, pp. 121–136, 1975. DOI: 10.1007/BF00342633.

[14] T. Tieleman and G. Hinton, "Lecture 6e – rmsprop: Divide the gradient by a running average of its recent magnitude," *Neural Networks for Machine Learning*, 2012. [Online]. Available: https://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf.

[15] N. Kheterpal, K. Parvate, C. Wu, A. Kreidieh, E. Vinitsky, and A. Bayen, "Flow: Deep reinforcement learning for control in SUMO," in *SUMO 2018 – Simulating Autonomous and Intermodal Transport Systems*, E. Wießner, L. Lücken, R. Hilbrich, *et al.*, Eds., ser. EPiC Series in Engineering, vol. 2, EasyChair, 2018, pp. 134–151. DOI: 10.29007/dkzb.